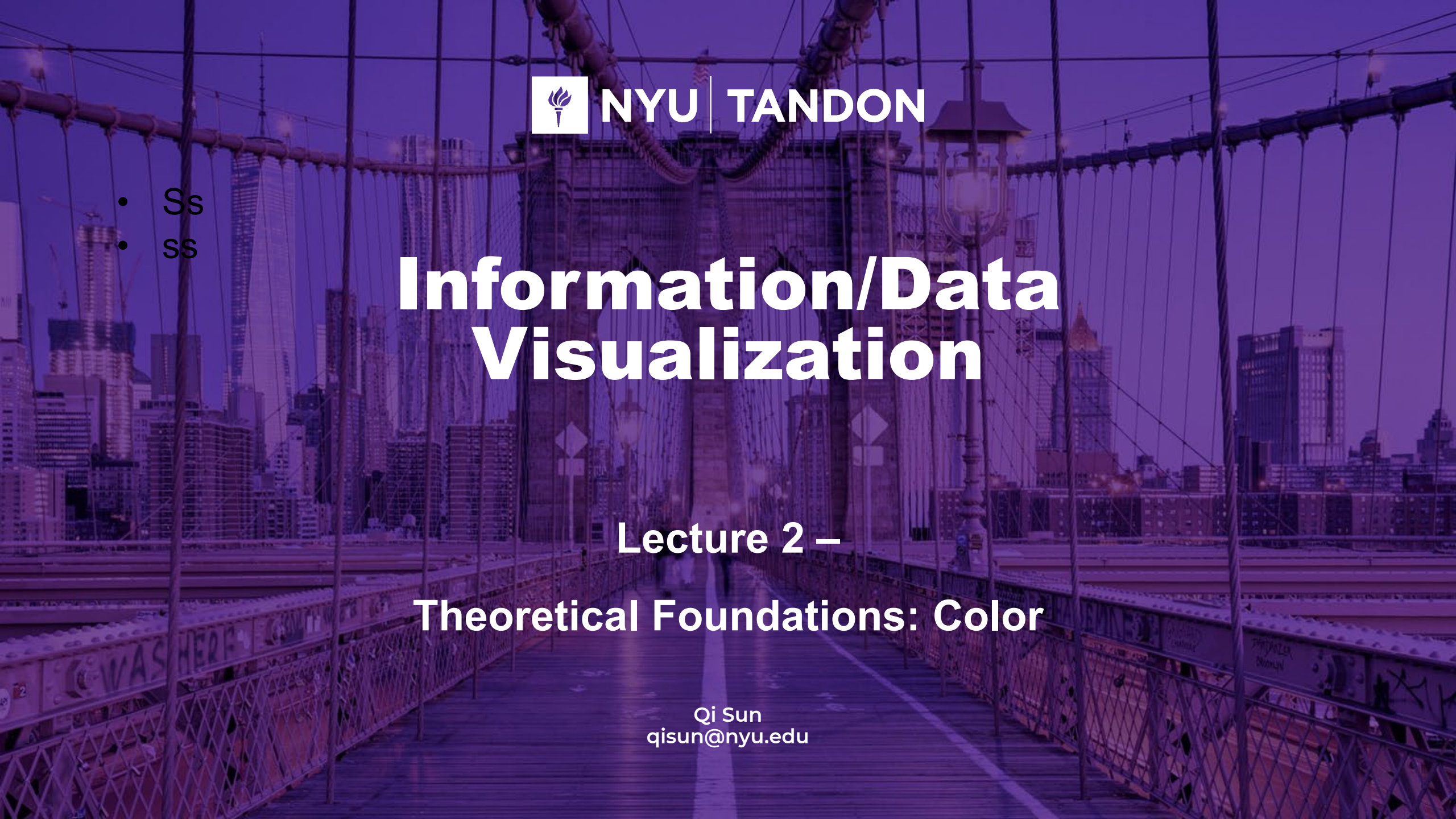


NYU TANDON

- Ss
- ss

Information/Data Visualization

Lecture 2 –

Theoretical Foundations: Color

Qi Sun
qisun@nyu.edu

Where We Are

- Basic mathematical tools

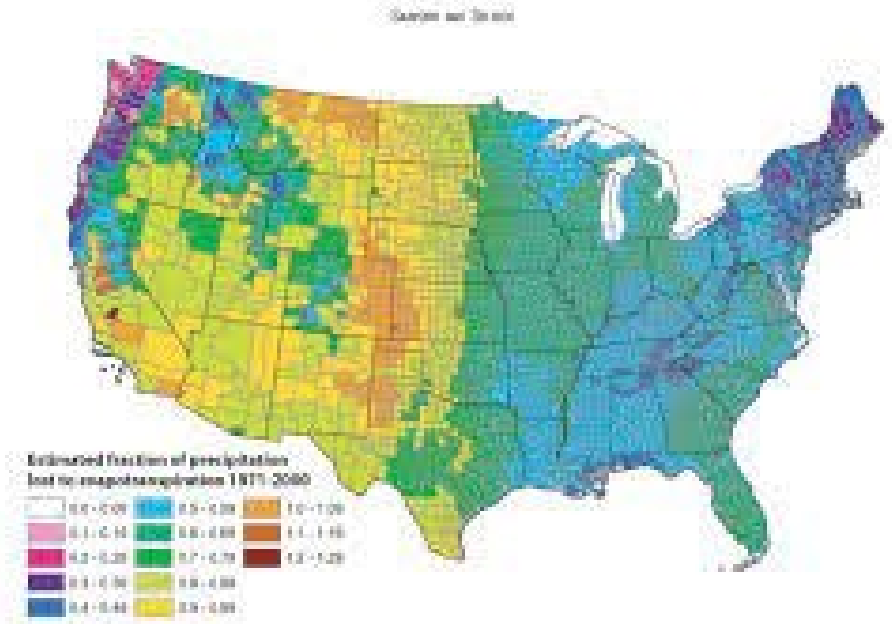
- Color: definition, space, and color map
- 2D visualization

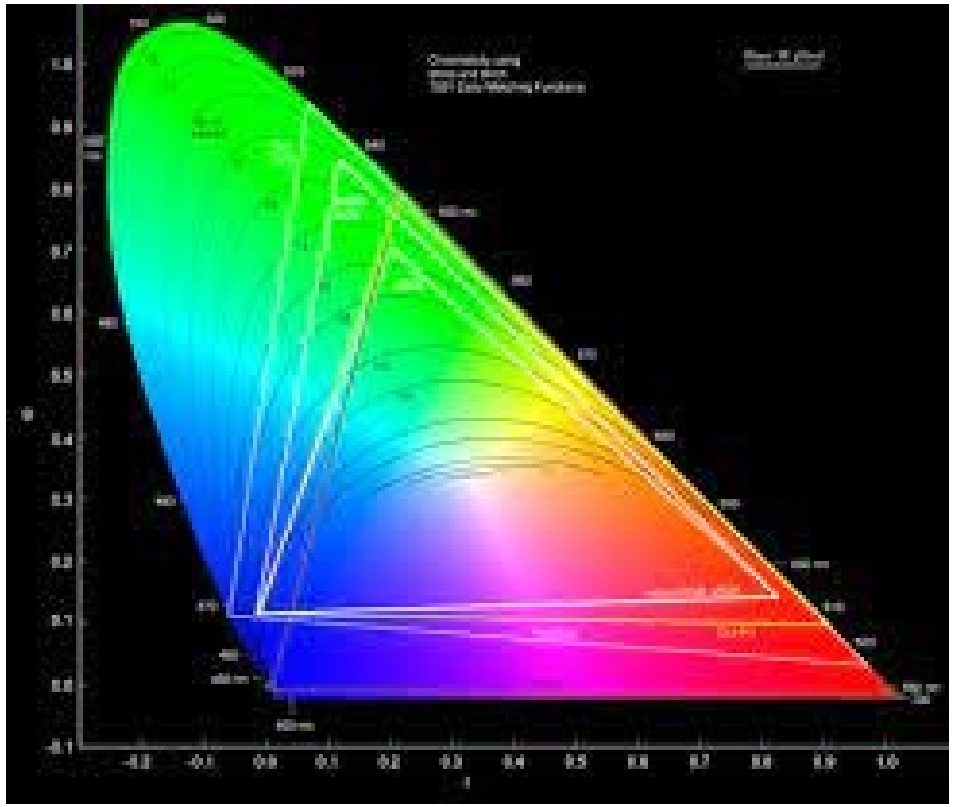
- Human perception of 3D
- 3D visualization

- Application case studies

Color in Visualization

- What is Color?
- Human/optical definitions of colors
- Why RGB?
- Color space
- Contrast and contrast sensitivity
- Color in visualization
- Color Issues

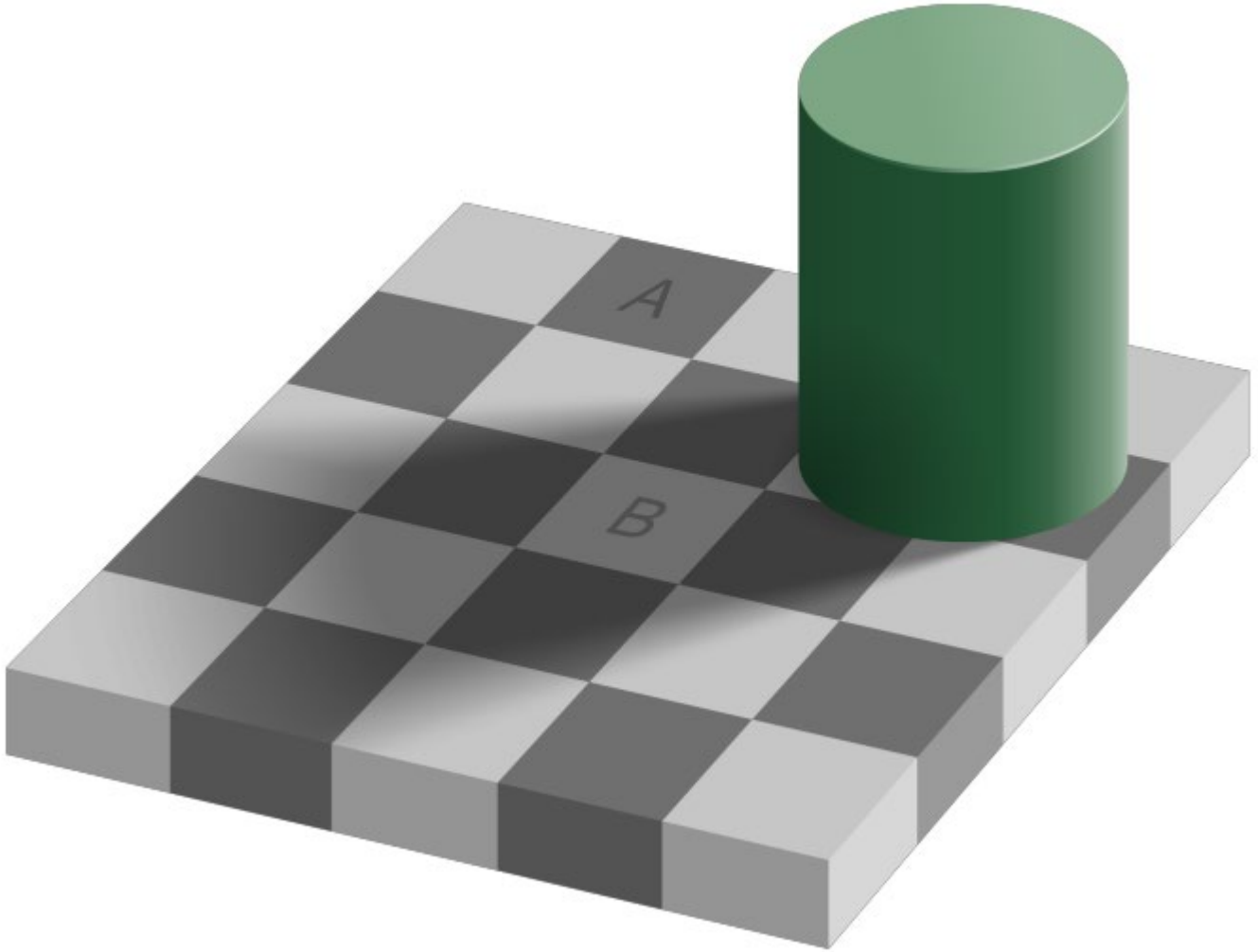

Estimated fraction of precipitation
lost to evapotranspiration 1971-2000



1. Color --

Definition, Metamerism, Perception

What is Color?



What is Color?

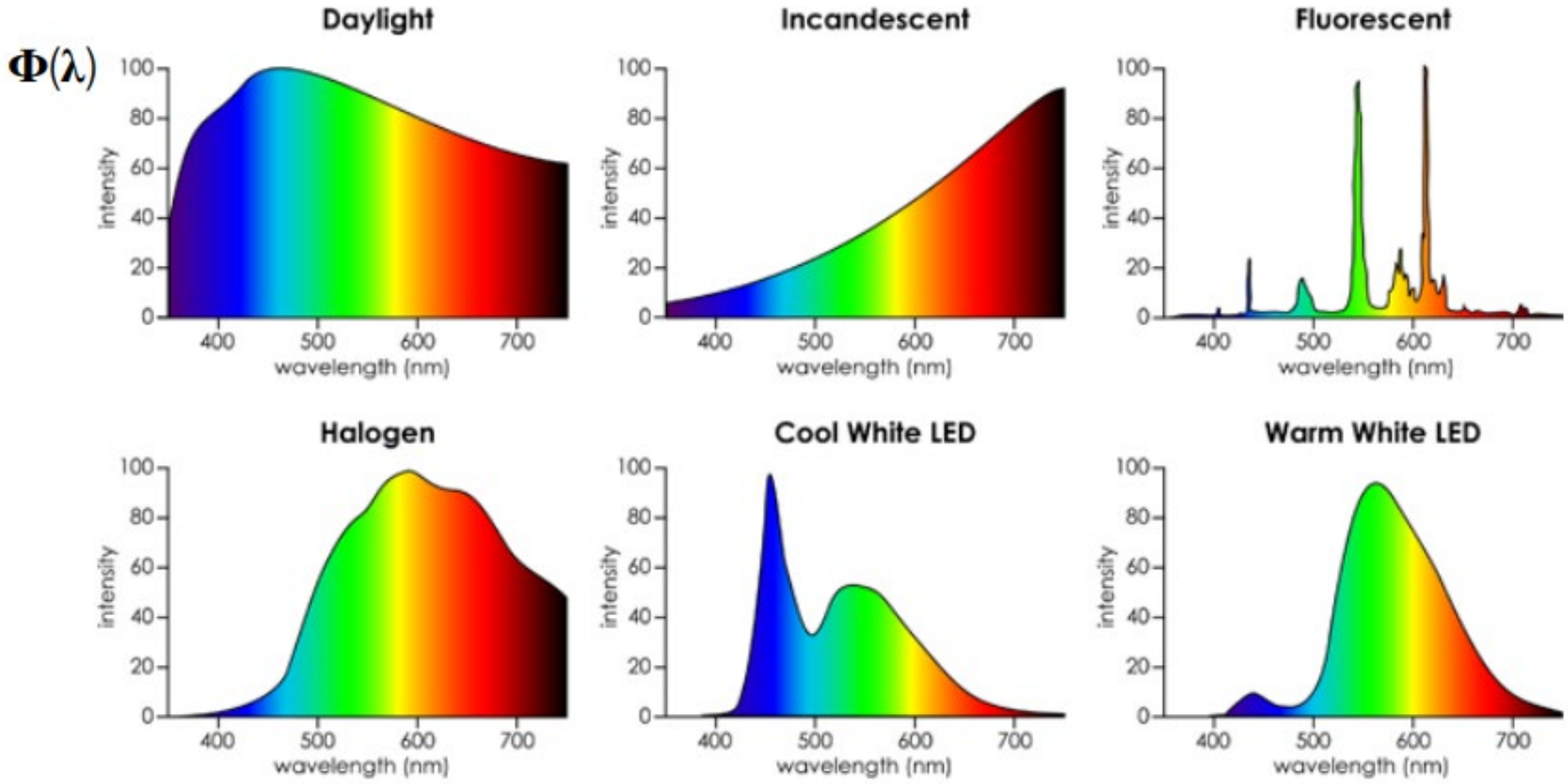
A

B

Color is Light!

https://www.shutterstock.com/video/clip-369937-prism---light-dispersion-seamless-loop-hd1080p
https://www.usatoday.com/story/opinion/2013/06/23/pink-floyd-royalties-pandora-column/2447445

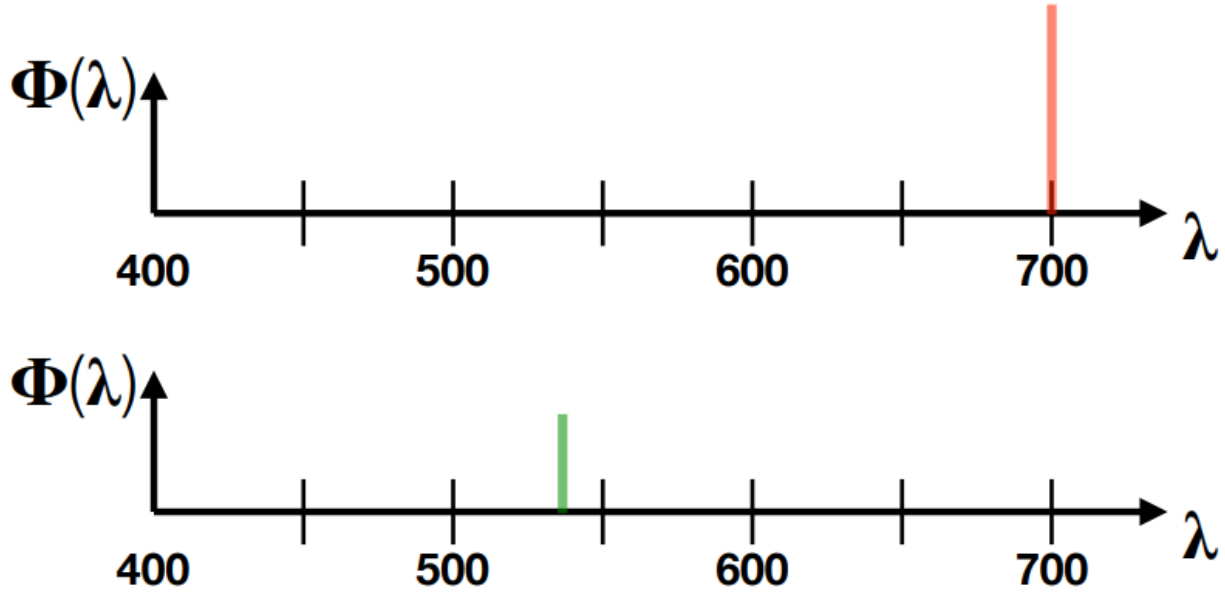
Color is Light!



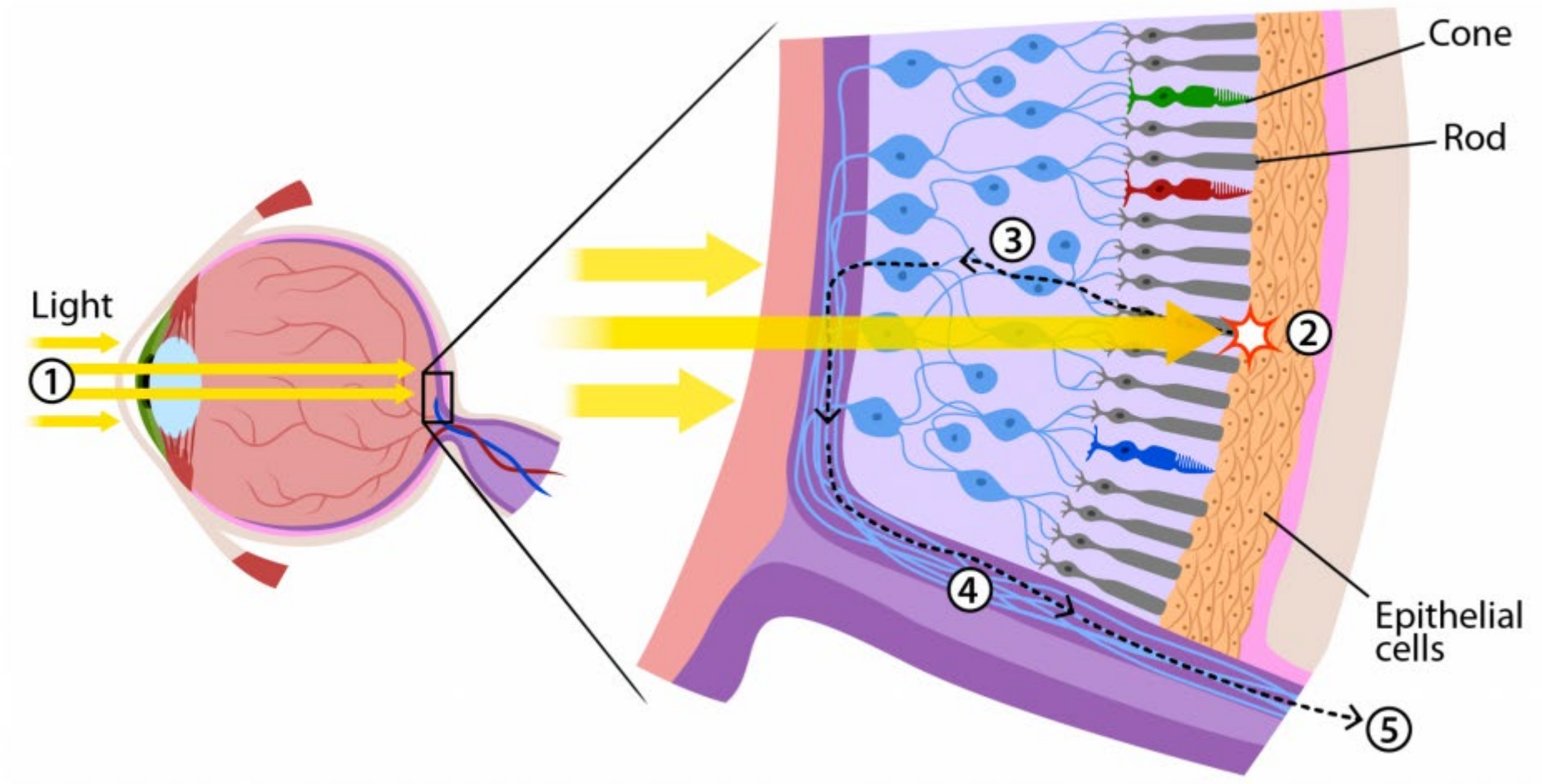
[Yuhao Zhu, U of Rochester

Color is Light!

Single-wavelength (monochromatic) light are called spectral light.

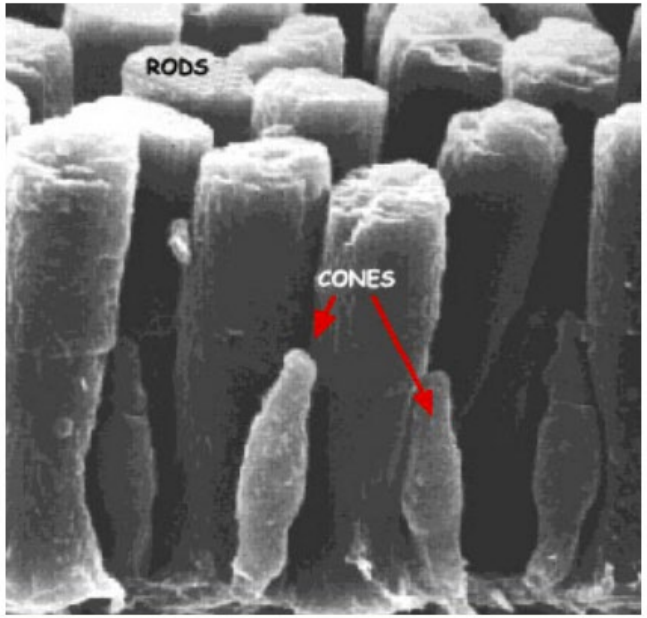


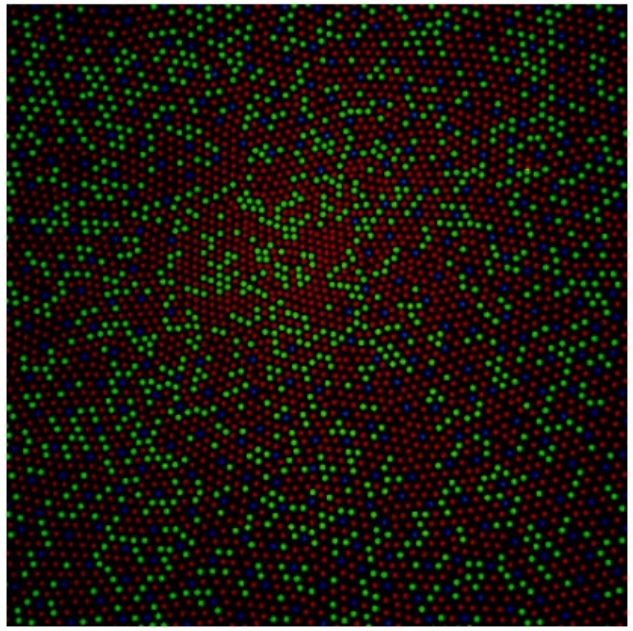
Color in Human Vision – Revisiting the Light



Credit: Prof. Michelle Borkin

Color in Human Vision – Revisiting the Light



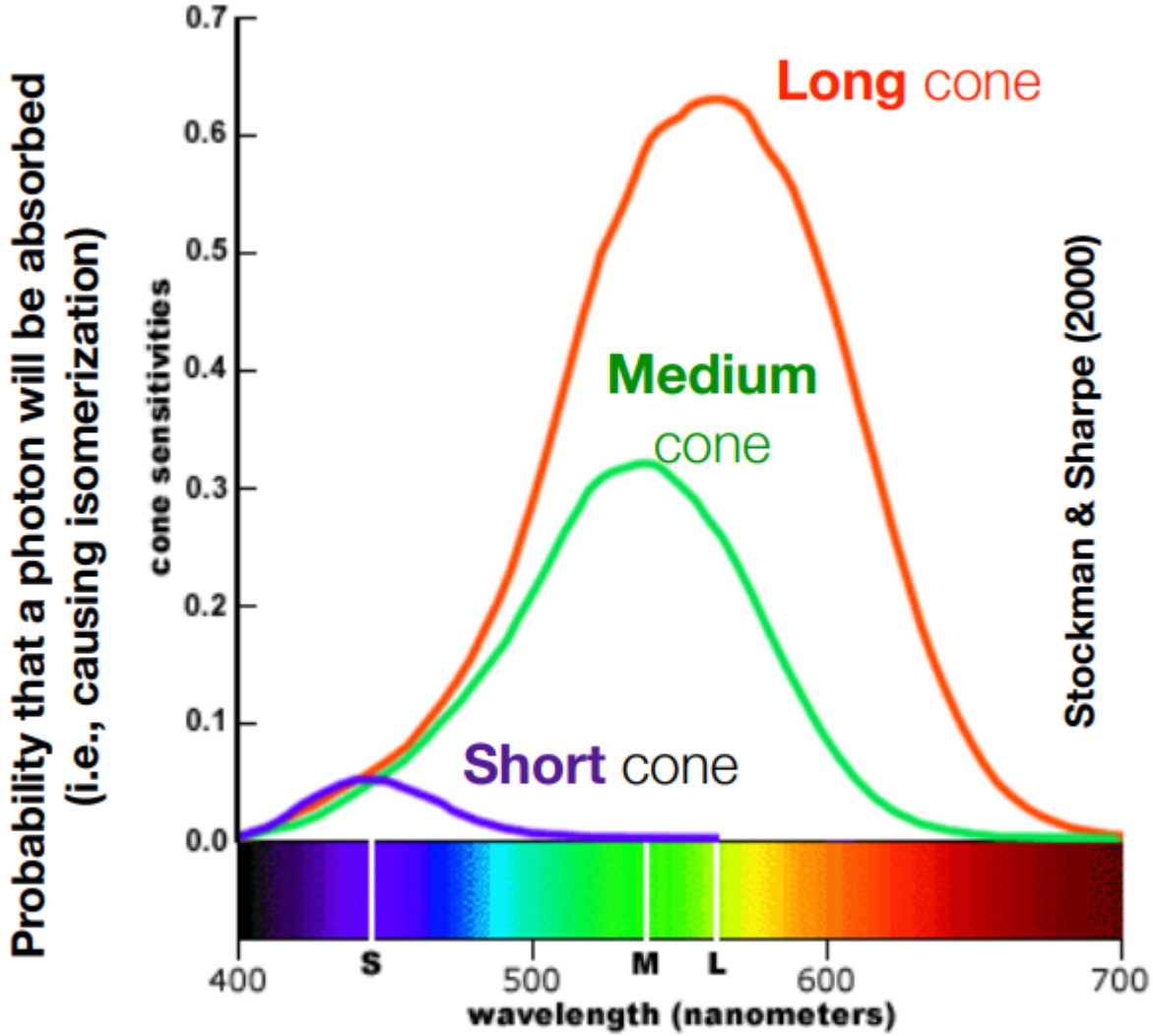


Rod: 120 million
Cone: 5-6 million

Three types of cone exist
at 10° fovea:
~60% L(ong),
~30% M(edium),
~10% S(hort)

Credit: Prof. Michelle Borkin

Color in Human Vision – Revisiting the Light

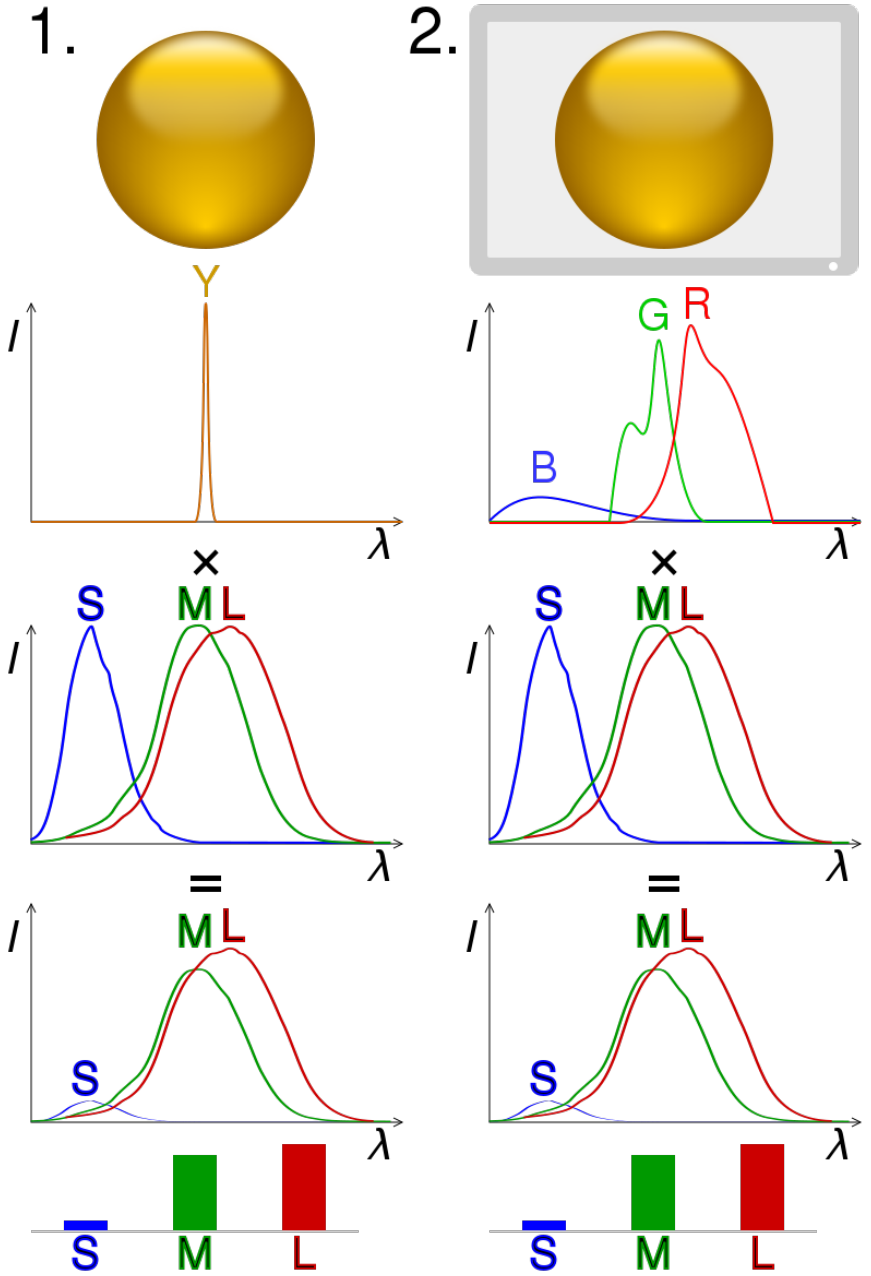


https://www.handprint.com/HP/WCL/color1.html

Color Metamerism

Metamerism is a perceived matching of colors with different (nonmatching) spectral distributions.

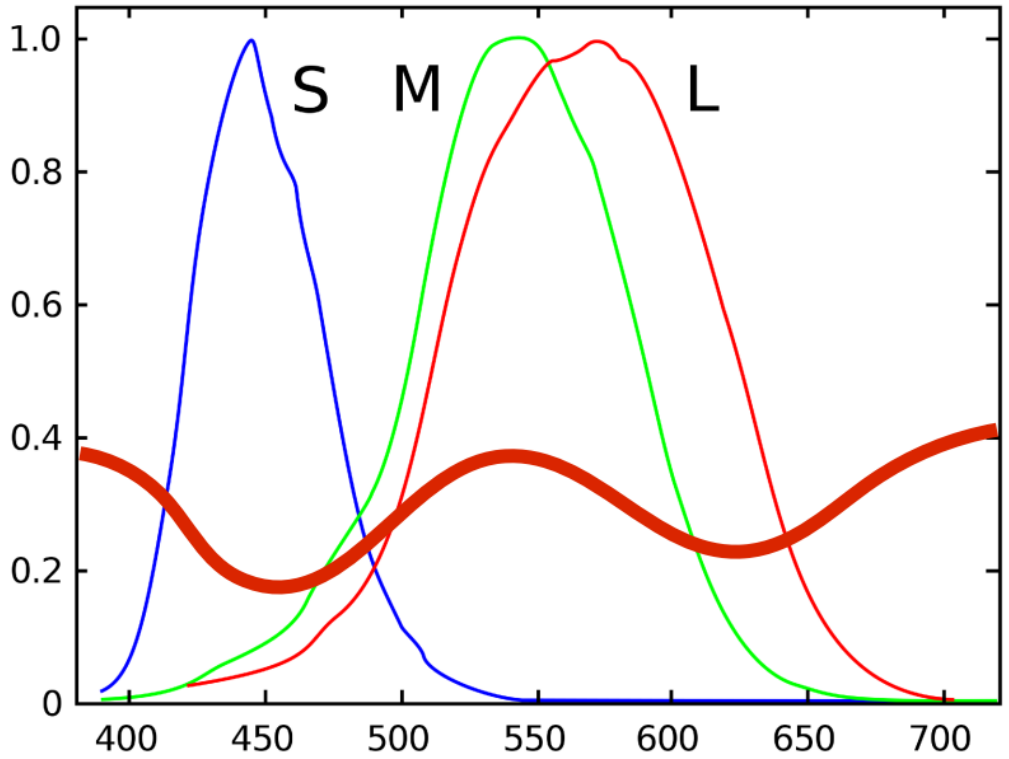
Colors that match this way are called **Metamers**.



Color Metamerism

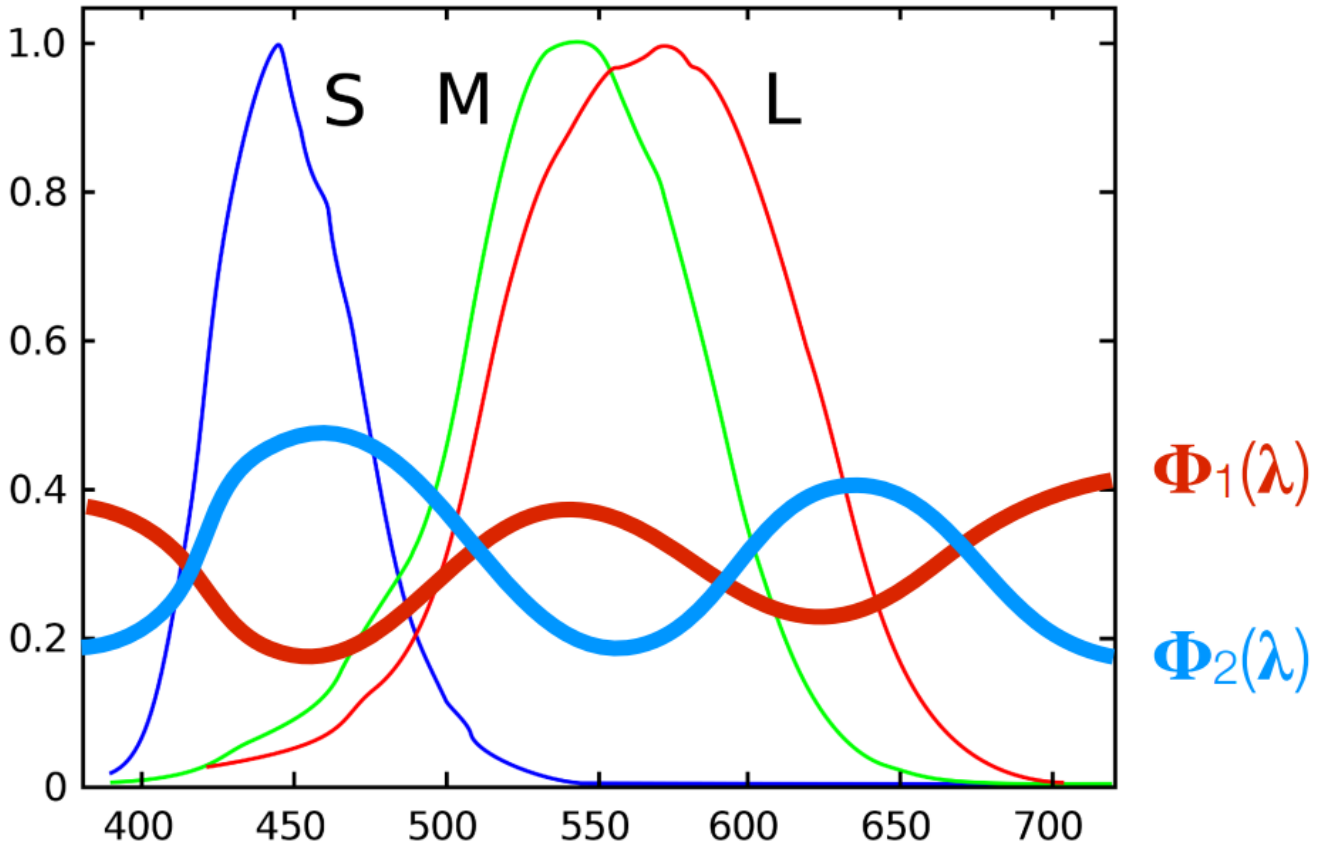
Given a light spectrum we can calculate the total responses of the L, M, and S cone cells.

$$L = \int_{\lambda} \Phi(\lambda) L(\lambda) d\lambda$$

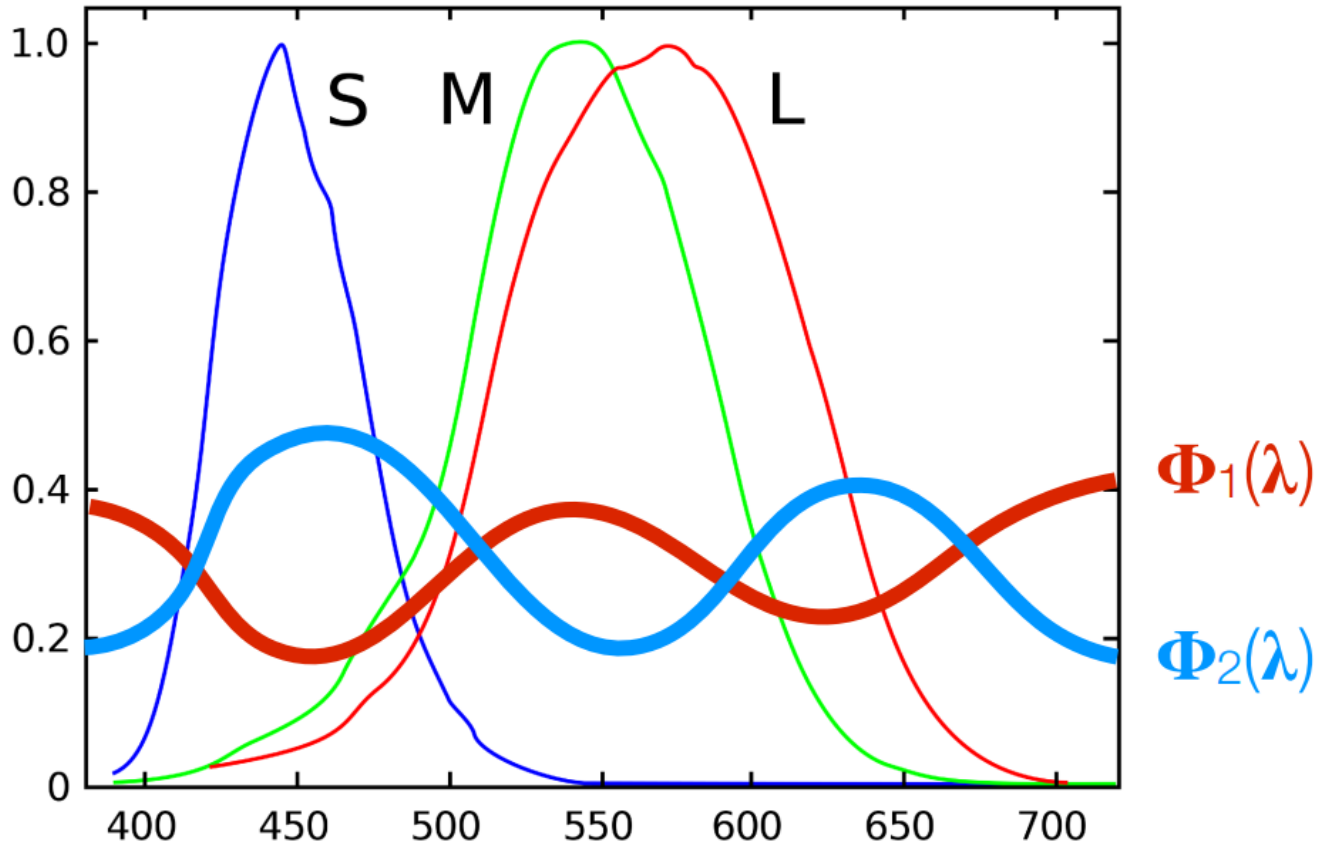


Color Metamerism

Given a light spectrum we can calculate the total responses of the L, M, and S cone cells.

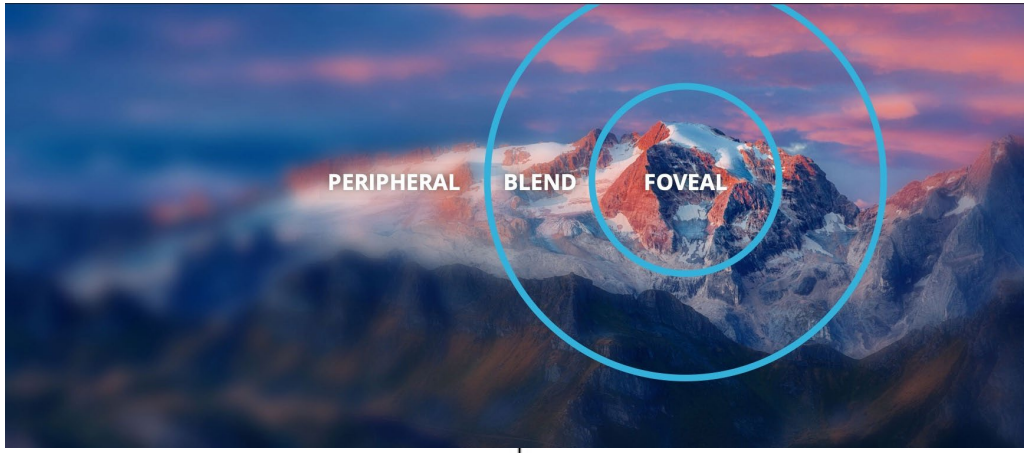


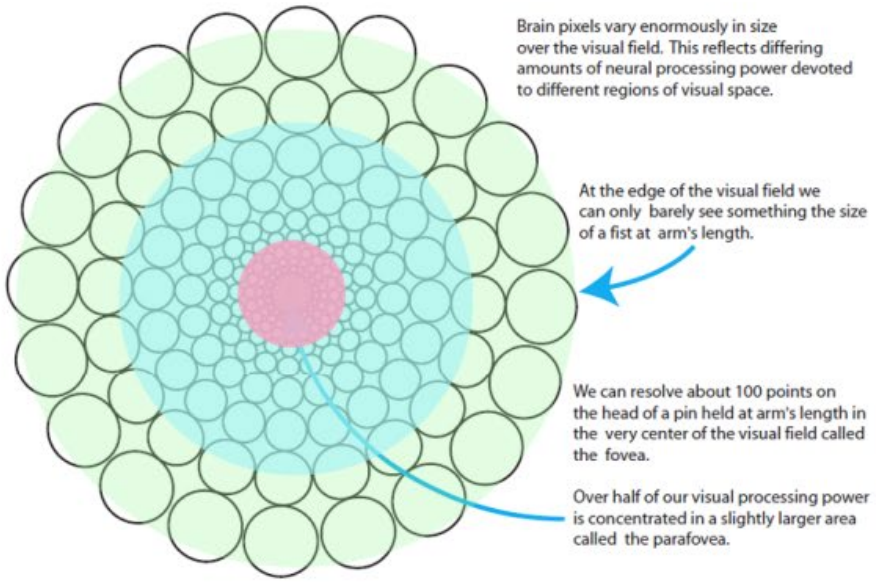
Color Metamerism

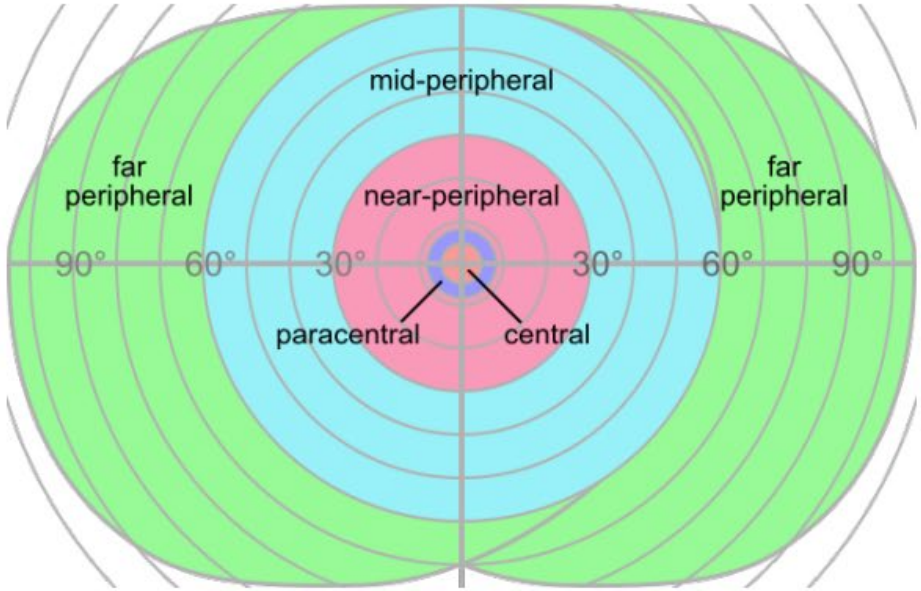


[Yuhao Zhu, U of Rochester

Color in Human Vision – Foveated Vision



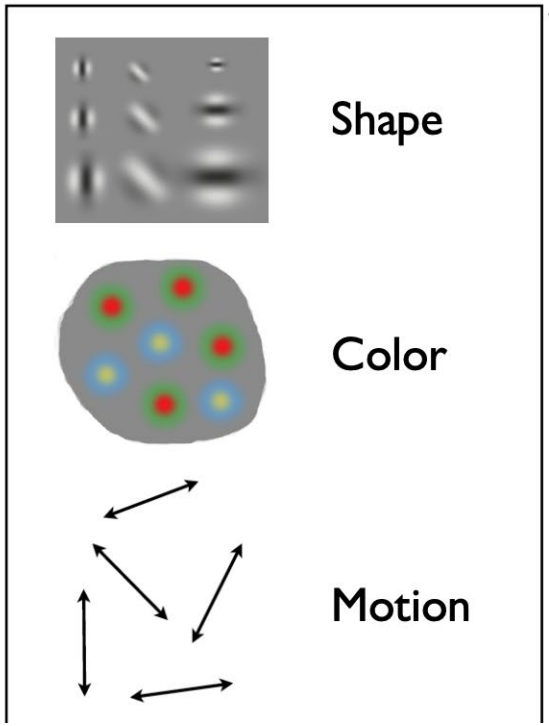


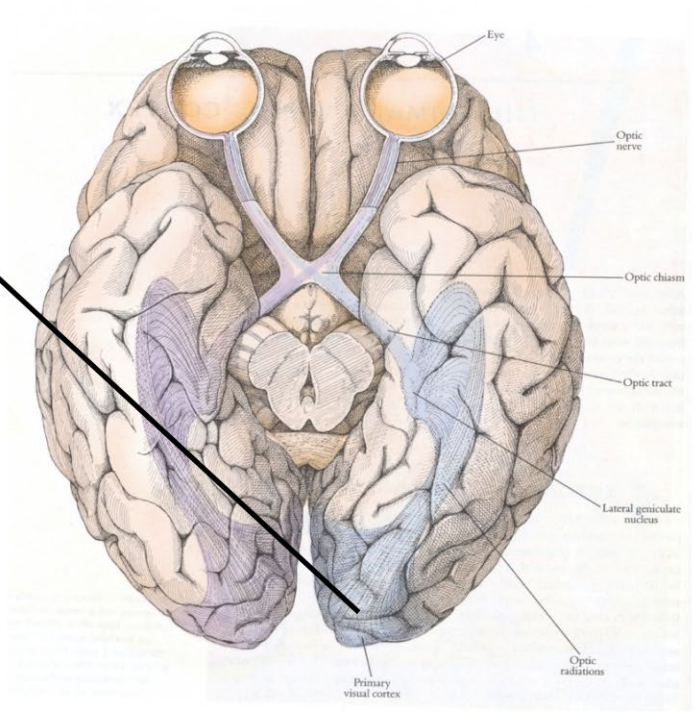


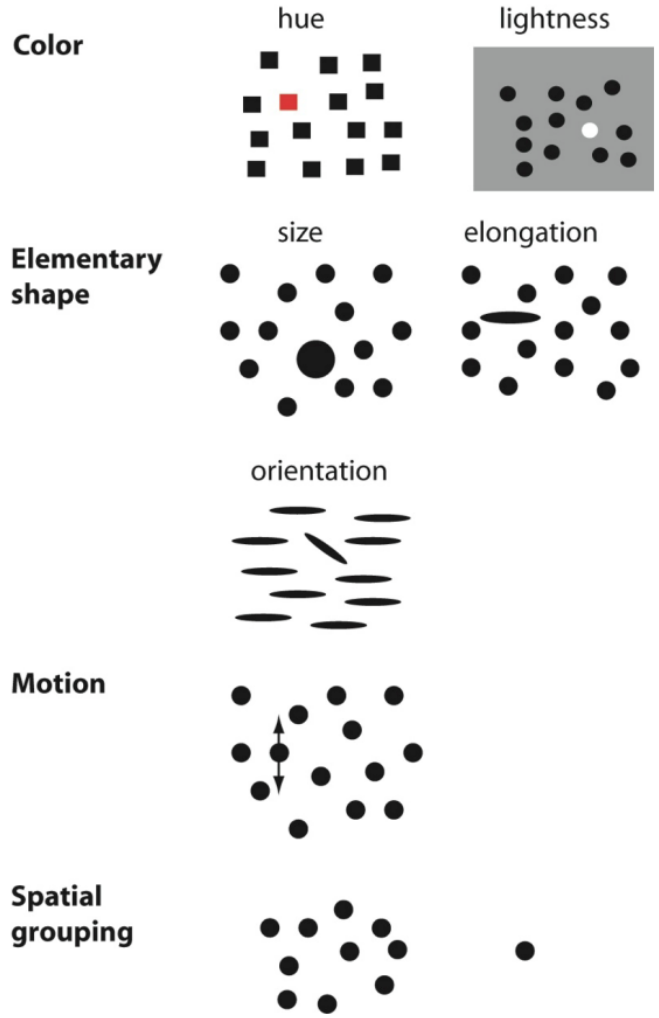
https://en.wikipedia.org/wiki/Peripheral_vision

Color in Human Vision – Feature Analysis

Use these "popout" effects to help design effective visualizations:
i.e., draw viewers' attentions

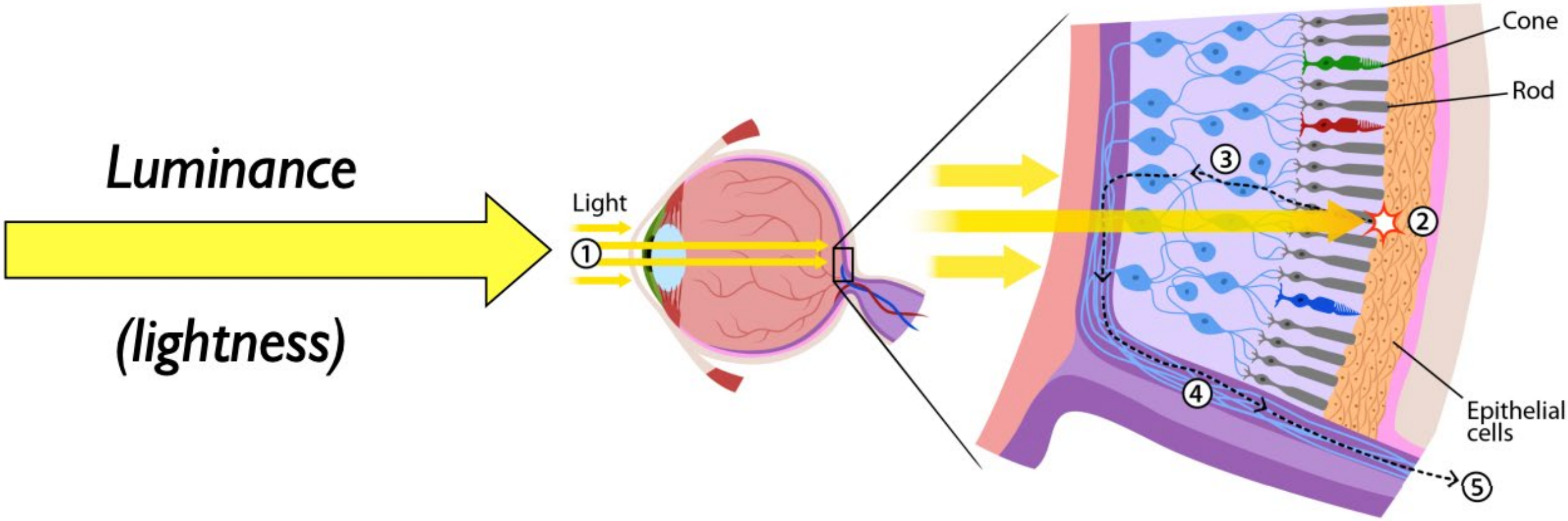






Credit: Prof. Michelle Borkin

Color in Human Vision – Luminance



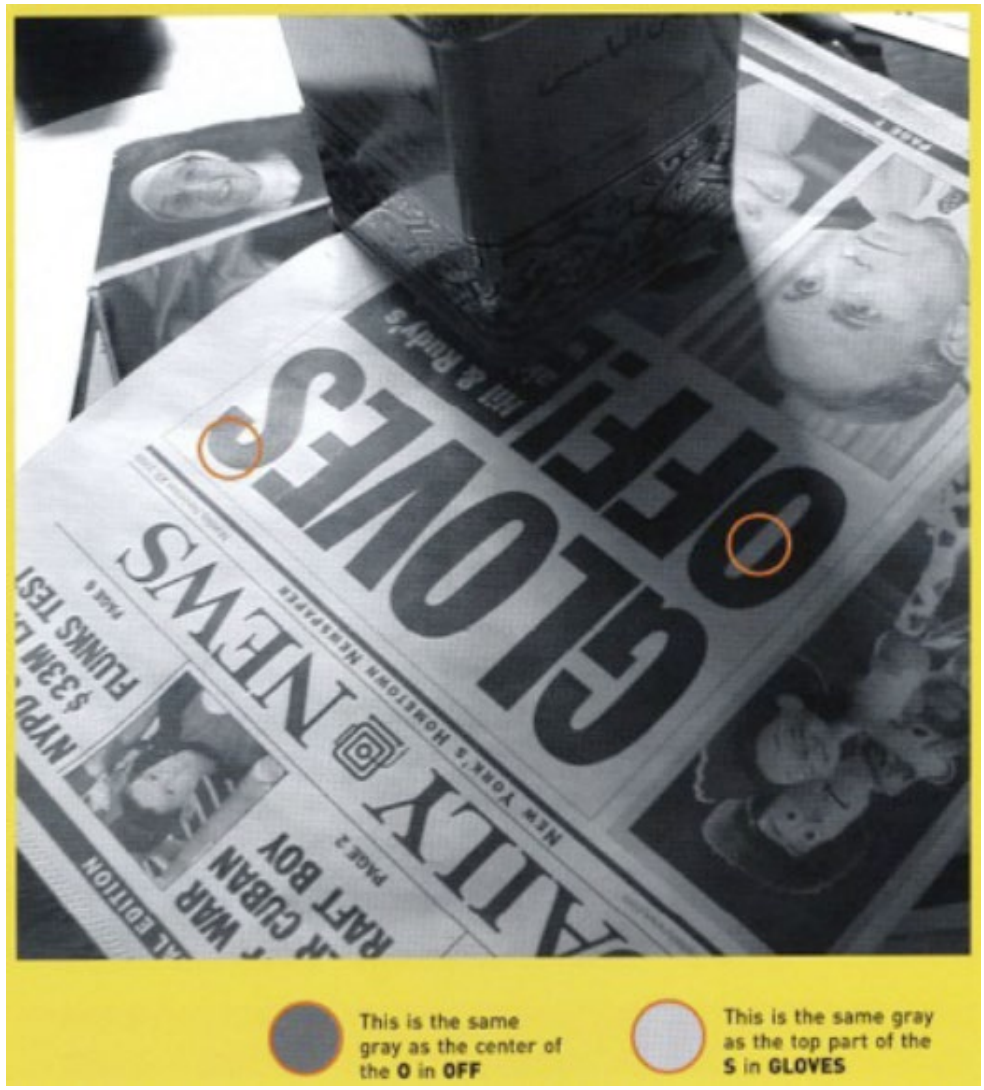
Lightness = the perceived intensity of reflected light (reflectance) from a surface

Brightness = the perceived intensity of emitted light

Credit: Prof. Michelle Borkin

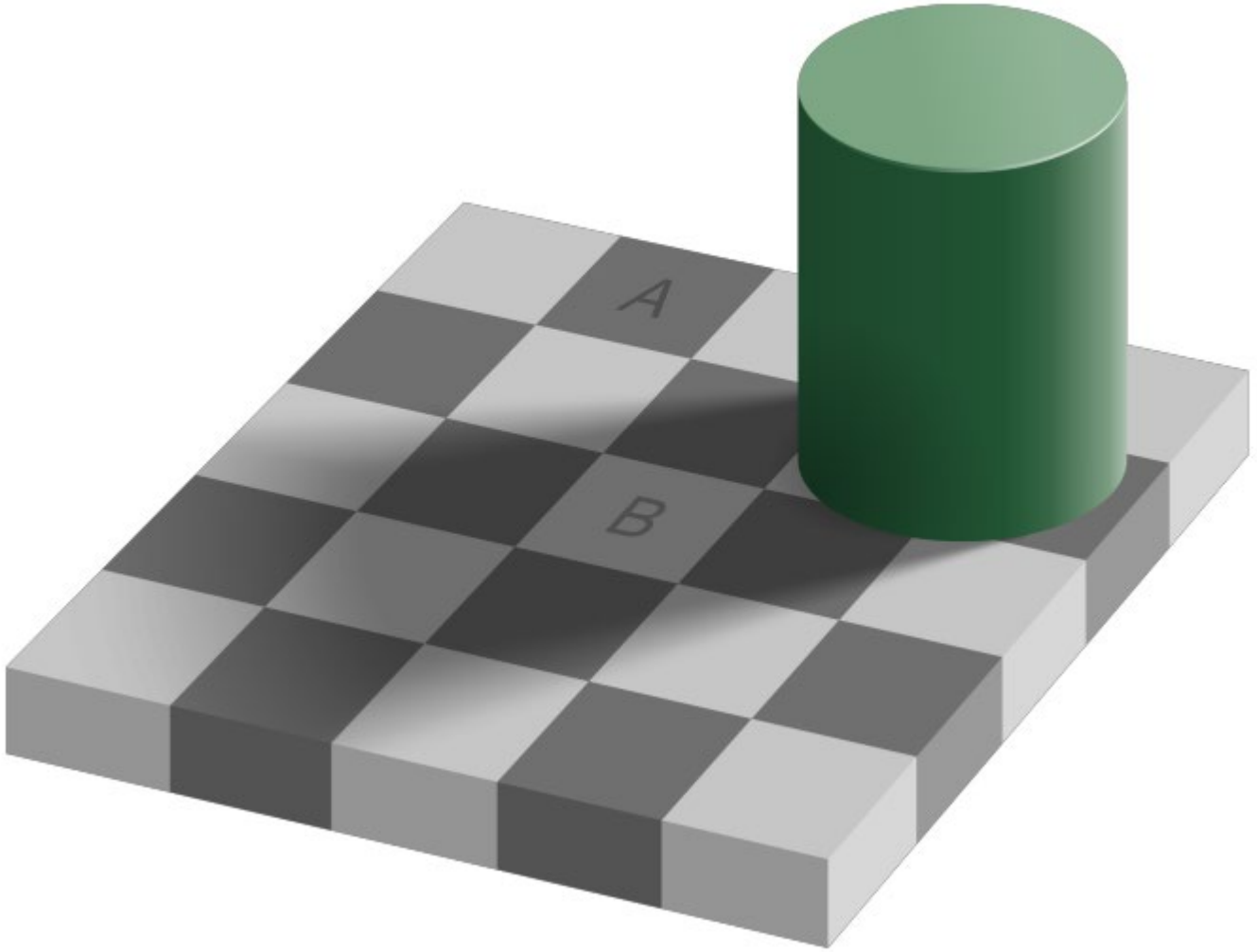
Color in Human Vision – Lightness Constancy

Lightness constancy refers to our ability to perceive the relative reflectance of objects despite changes in illumination.

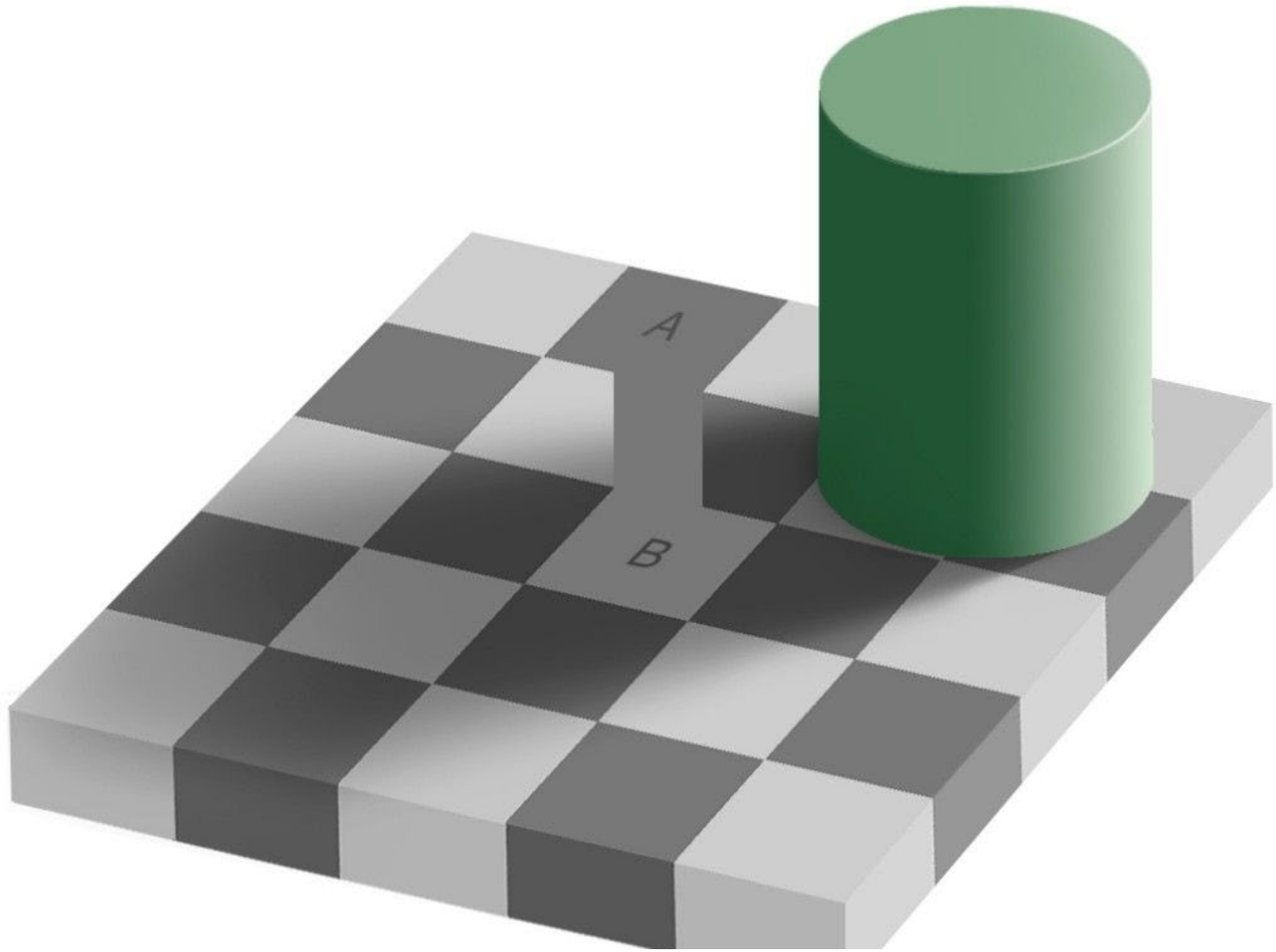


Credit: Prof. Michelle Borkin

Color in Human Vision – Lightness Constancy



Color in Human Vision – Lightness Constancy


A
B

Color in Human Vision – Lightness Constancy

Credit: Prof. Michelle Borkin

Color in Human Vision – Lightness Constancy

Credit: Prof. Michelle Borkin

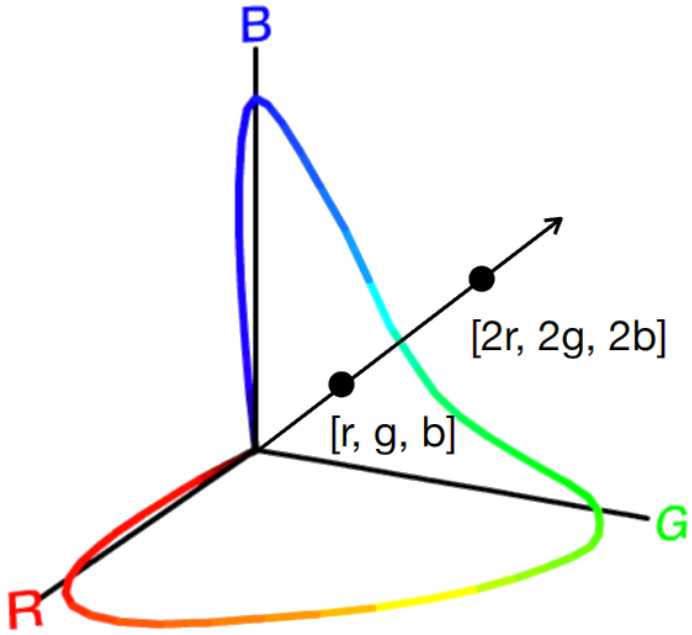
2. Color Space --

Chromaticity and *Gamut*

Color in Visualization - Chromaticity

$$C = \frac{R}{R+G+B}(R) + \frac{G}{R+G+B}(G) + \frac{B}{R+G+B}(B)$$

separate the objective quality and the quantity of a color.



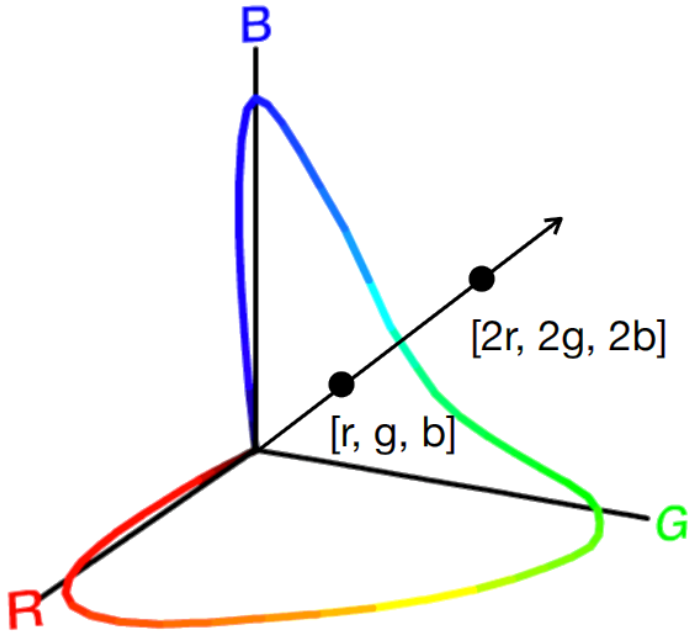
Color in Visualization - Chromaticity

$$C = \frac{R}{R+G+B}(R) + \frac{G}{R+G+B}(G) + \frac{B}{R+G+B}(B)$$

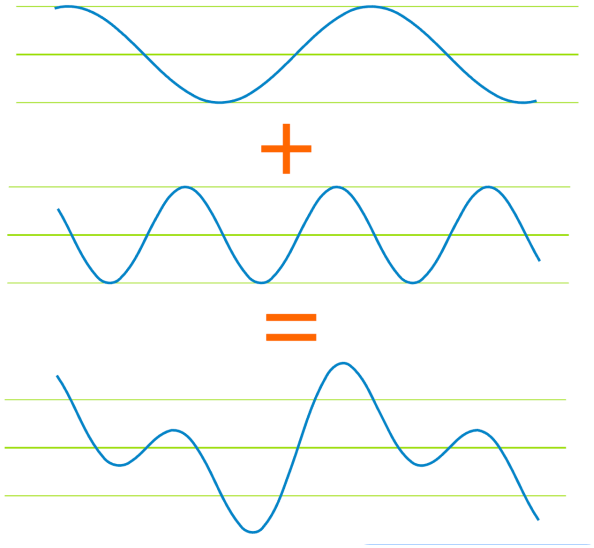
RGB to rgb.

rgb are called the **chromaticity**

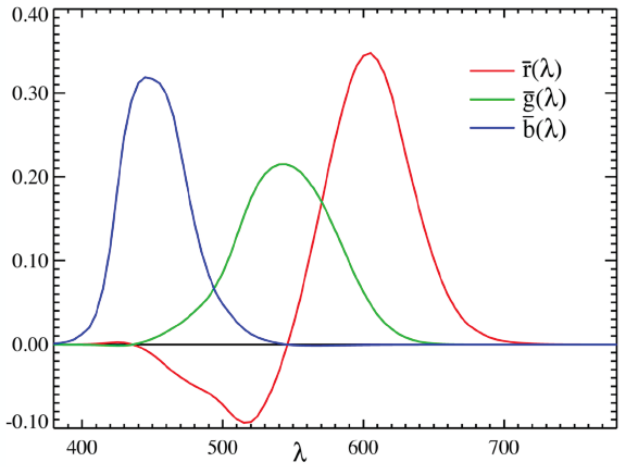
i.e., the “objective quality” of the color

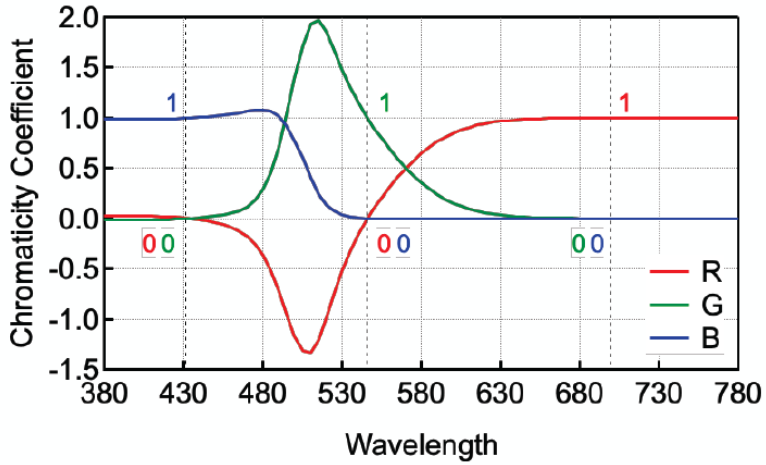


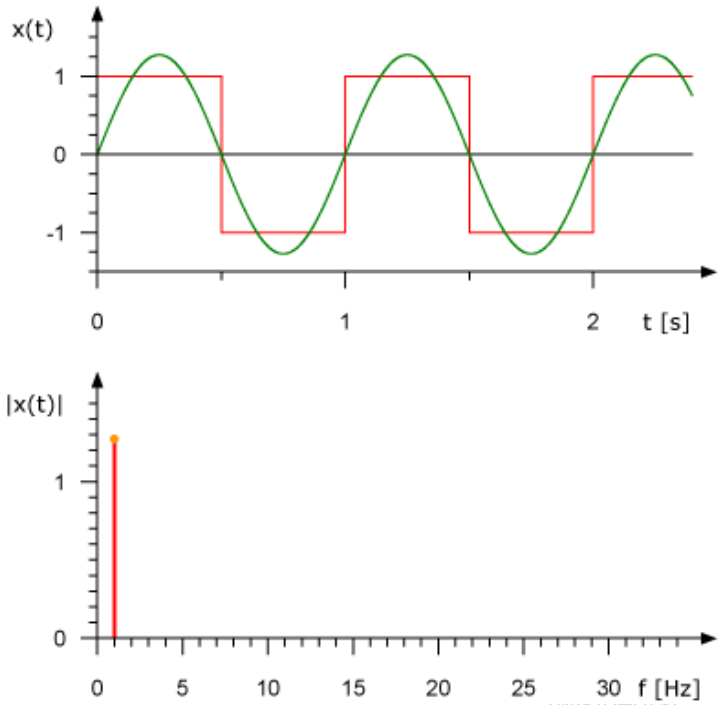
Thinking in Frequency – Fourier Theory



ANY signal can be
decomposed to spectral waves!







How to parameterize a signal?

- Option 1: Taylor series represents any function using polynomials.

$$\begin{aligned} f(x) &= f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 \\ &\quad + \frac{f'''(x_0)}{3!}(x - x_0)^3 + \frac{f''''(x_0)}{4!}(x - x_0)^4 + \cdots \\ &= \sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n. \end{aligned}$$

- Polynomials are not the best - unstable and not very semantically meaningful.

- Easier to talk about “data” in terms of its “frequencies”
  (how fast/often signals change, etc).

How to parameterize a signal?

- Intuition: **Any** periodic data can be rewritten as a weighted sum of Sines and Cosines of different frequencies.

- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson
  - Not translated into English until 1878!

- But it's true!
  - called Fourier Series
  - Possibly the greatest tool used in Engineering

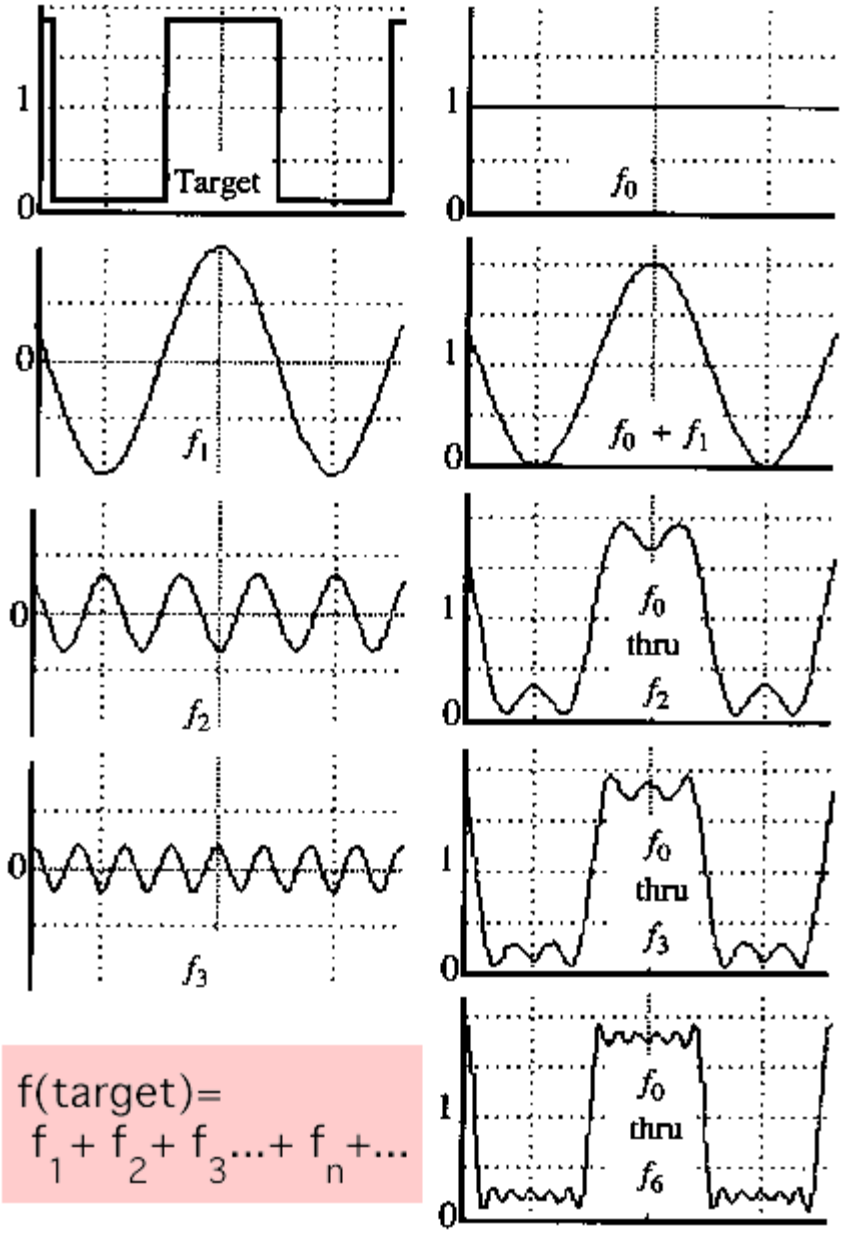
Jean Baptiste Joseph Fourier
(1768-1830)

A Sum of Sinusoids

- Our building block:

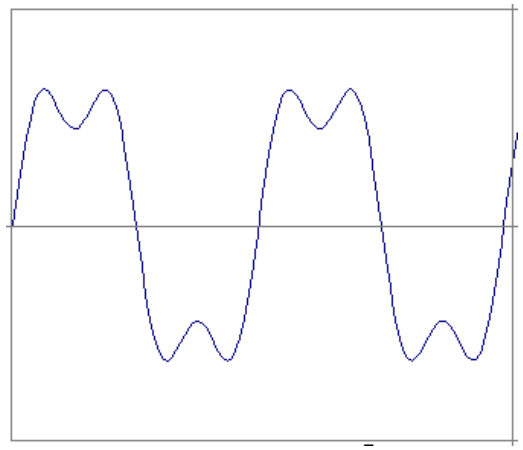
$$A\sin(\omega x + \phi)$$

- Add enough of them to get any signal *f(x)* you want!
- What does each control?
- Which one encodes the **coarse vs. fine** structure of the data?



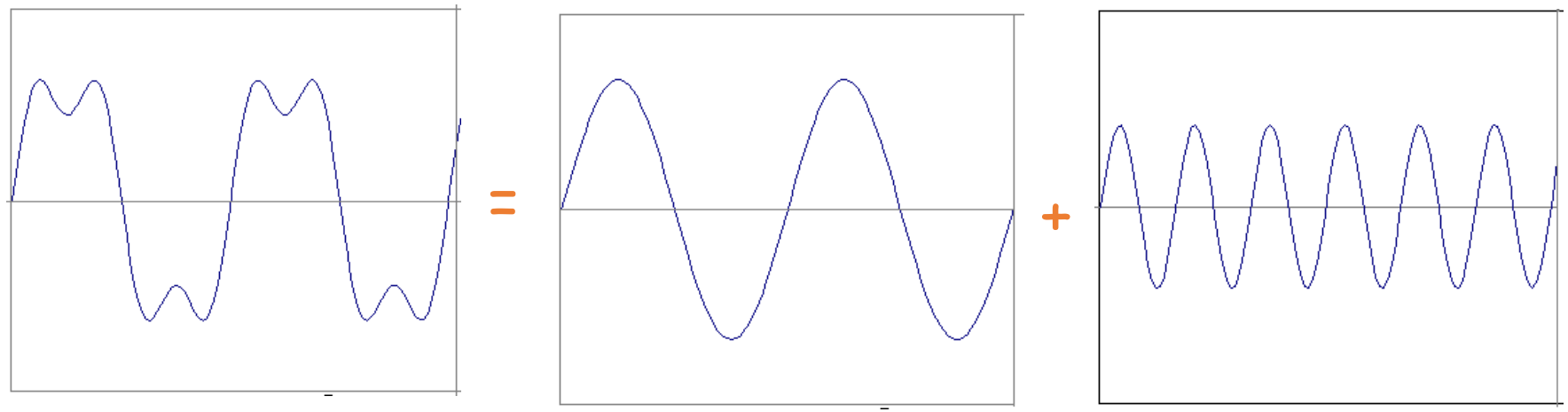
Time and Frequency

- example : $g(t) = \sin(2pi\, f\, t) + (1/3)\sin(2pi\, (3f)\, t)$



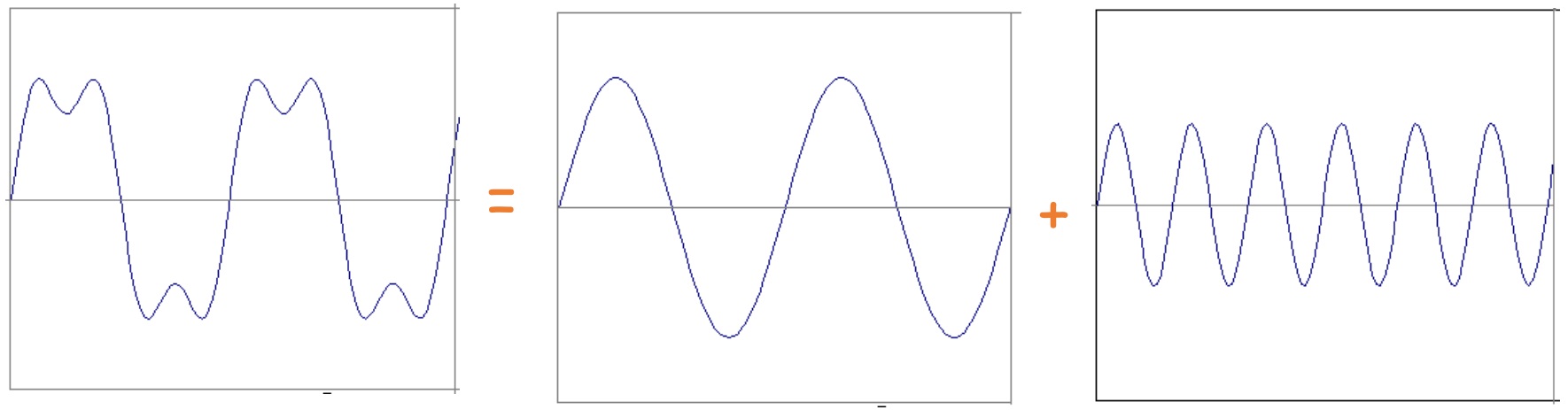
Time and Frequency

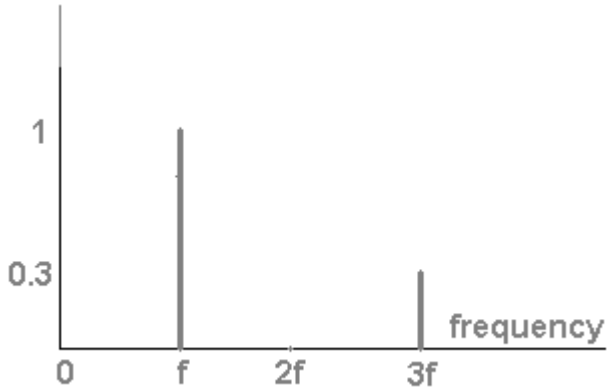
- example : $g(t) = \sin(2pi\, f\, t) + (1/3)\sin(2pi\, (3f)\, t)$



Frequency Spectra

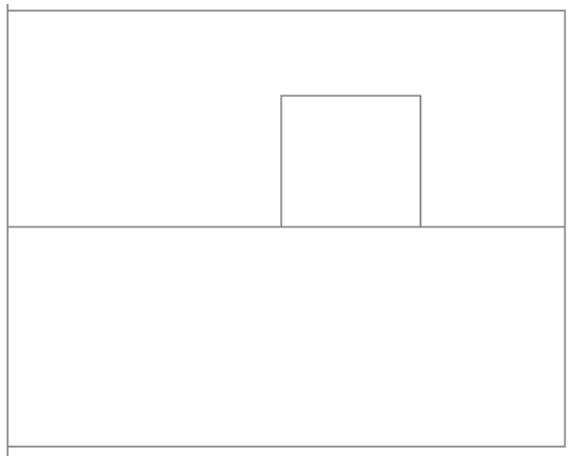
- example : $g(t) = \sin(2pi\, f\, t) + (1/3)\sin(2pi\, (3f)\, t)$



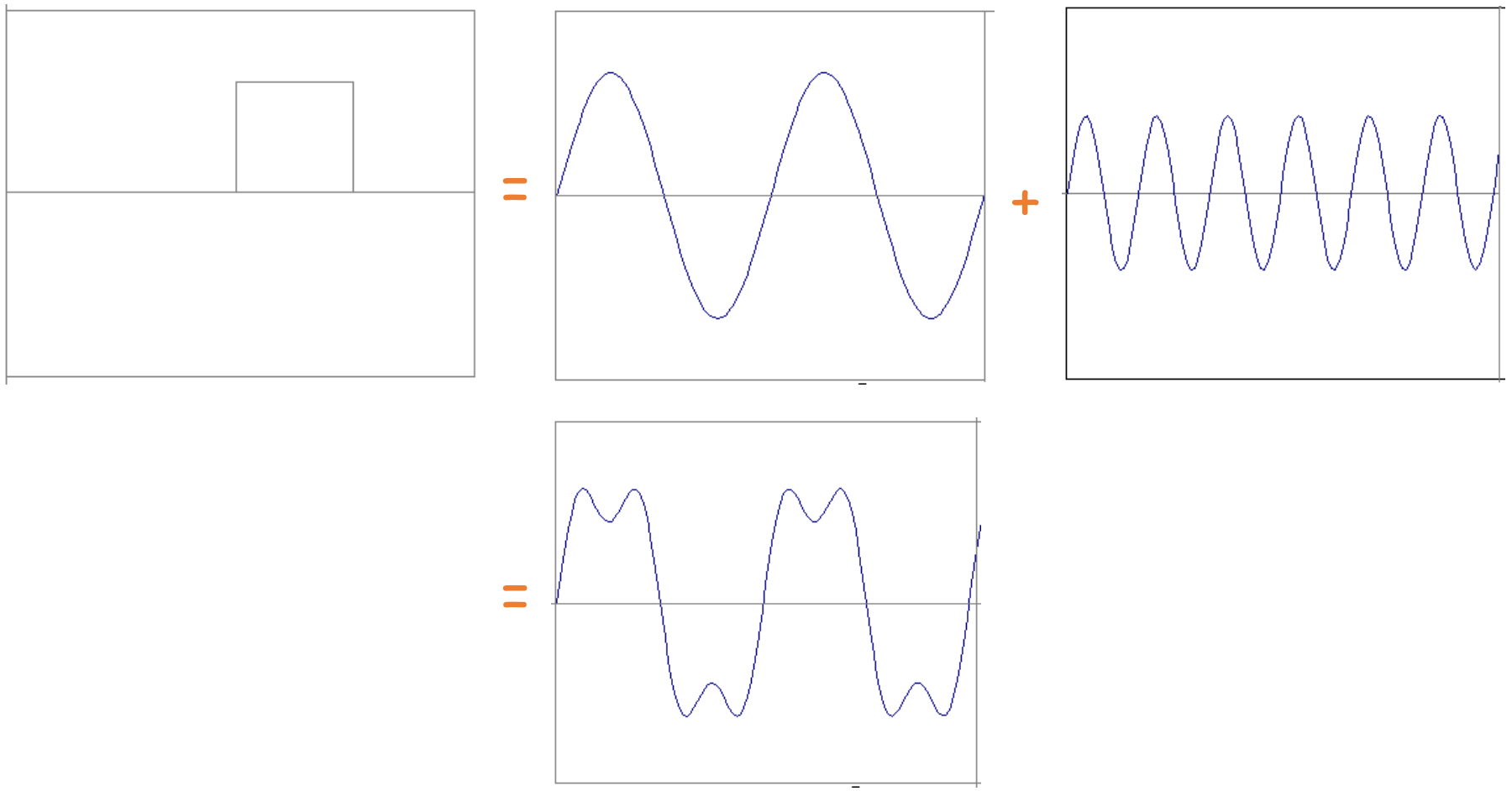


Frequency Spectra

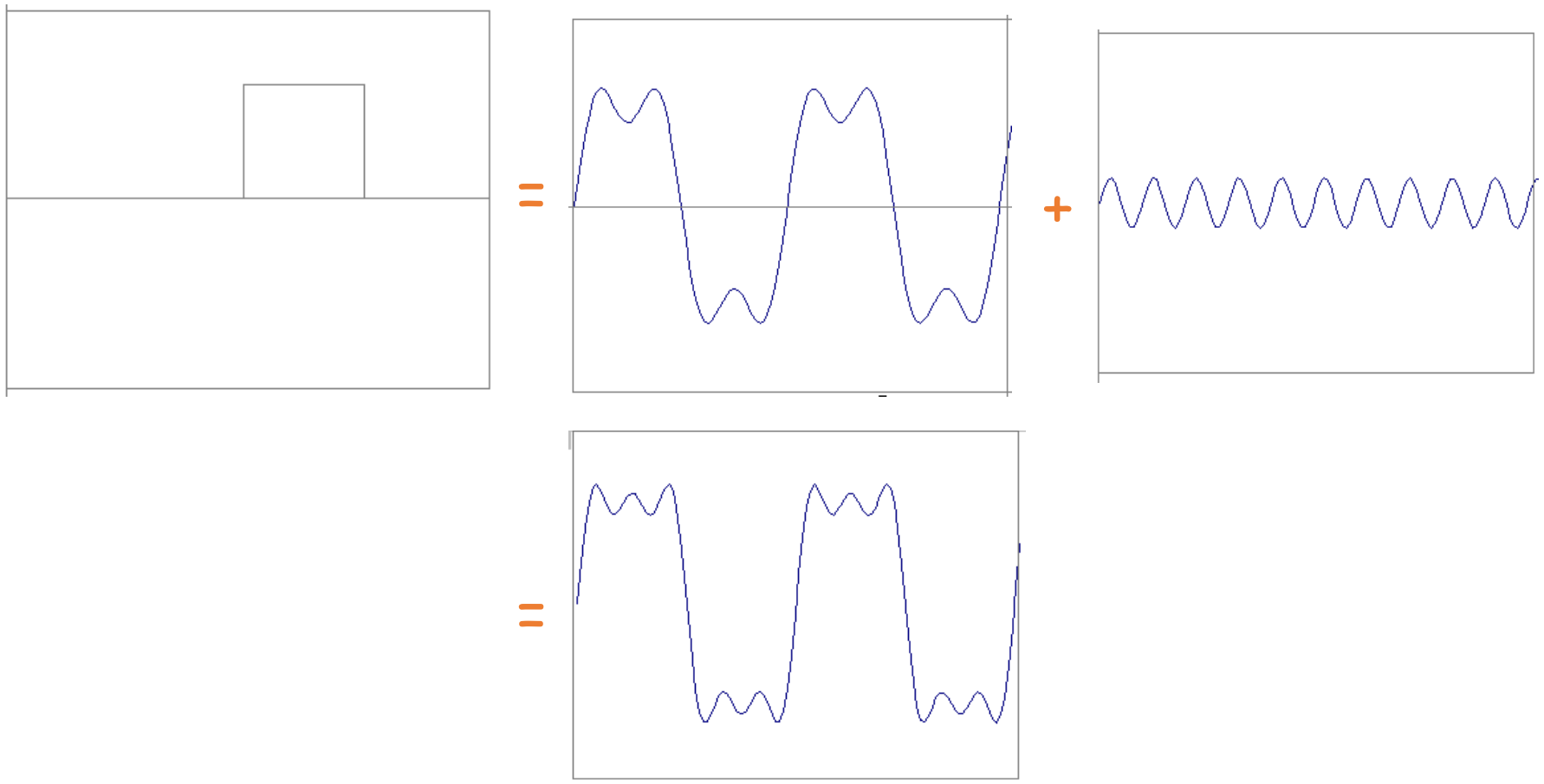
- Usually, frequency is more interesting than the phase



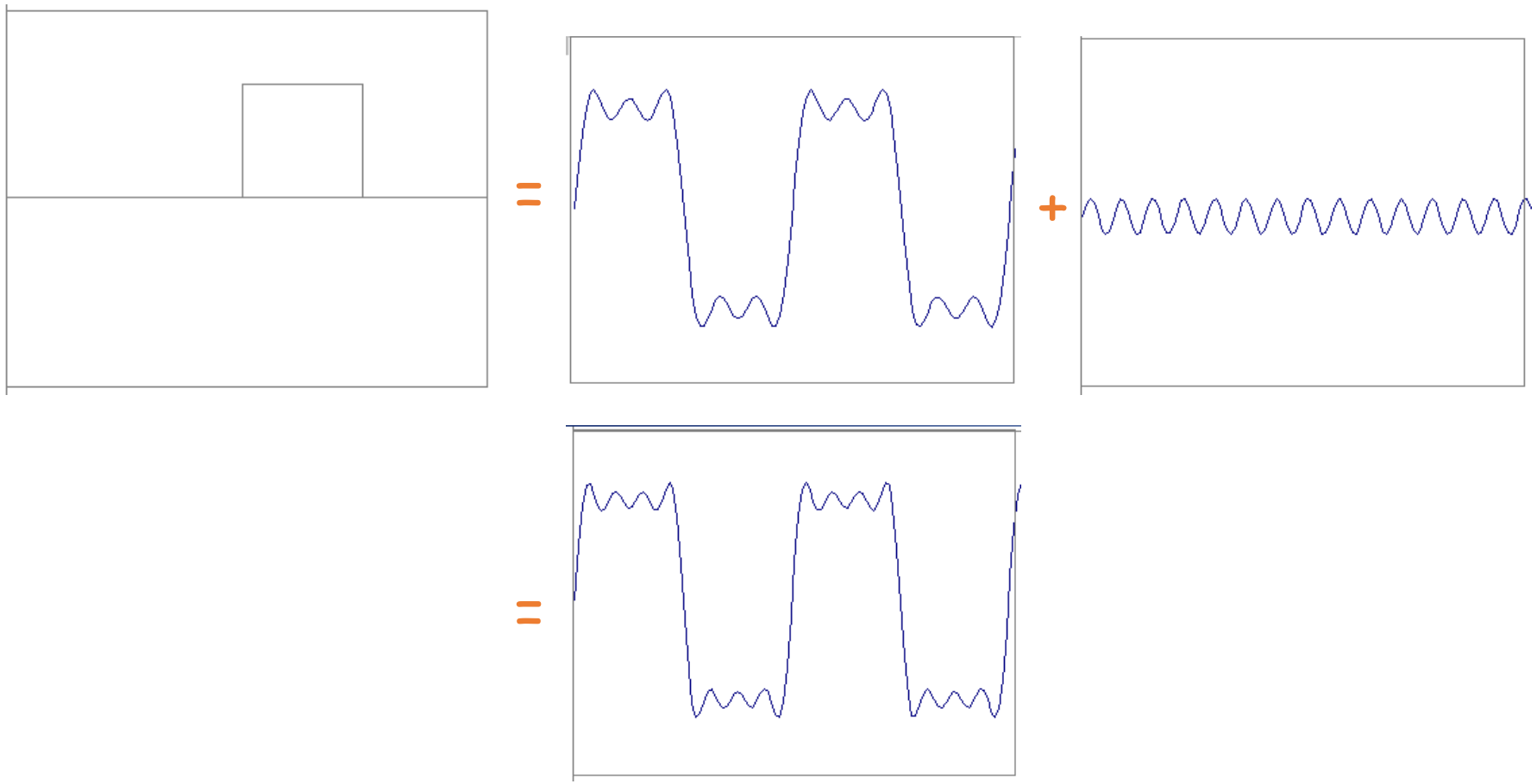
Frequency Spectra



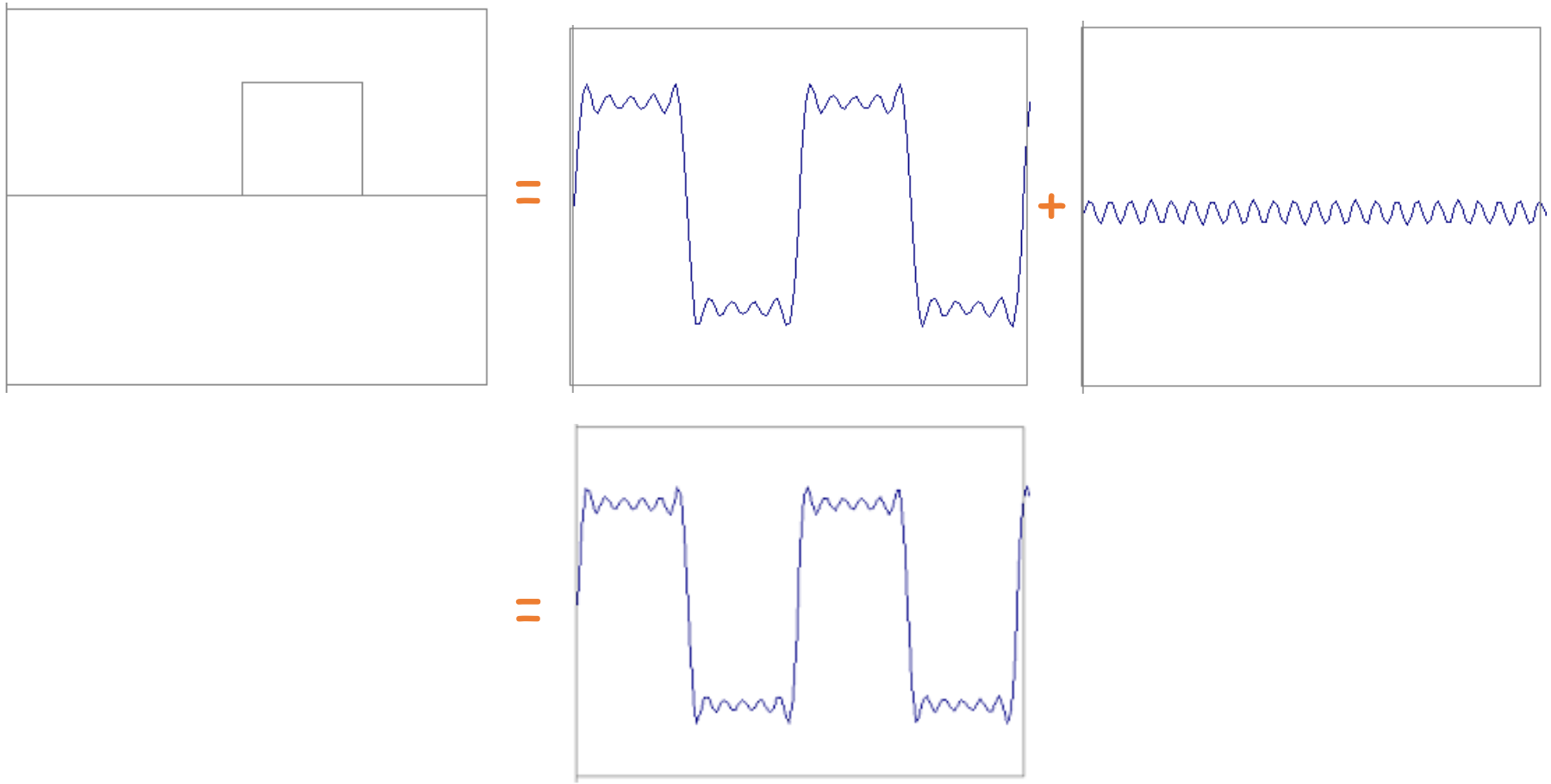
Frequency Spectra



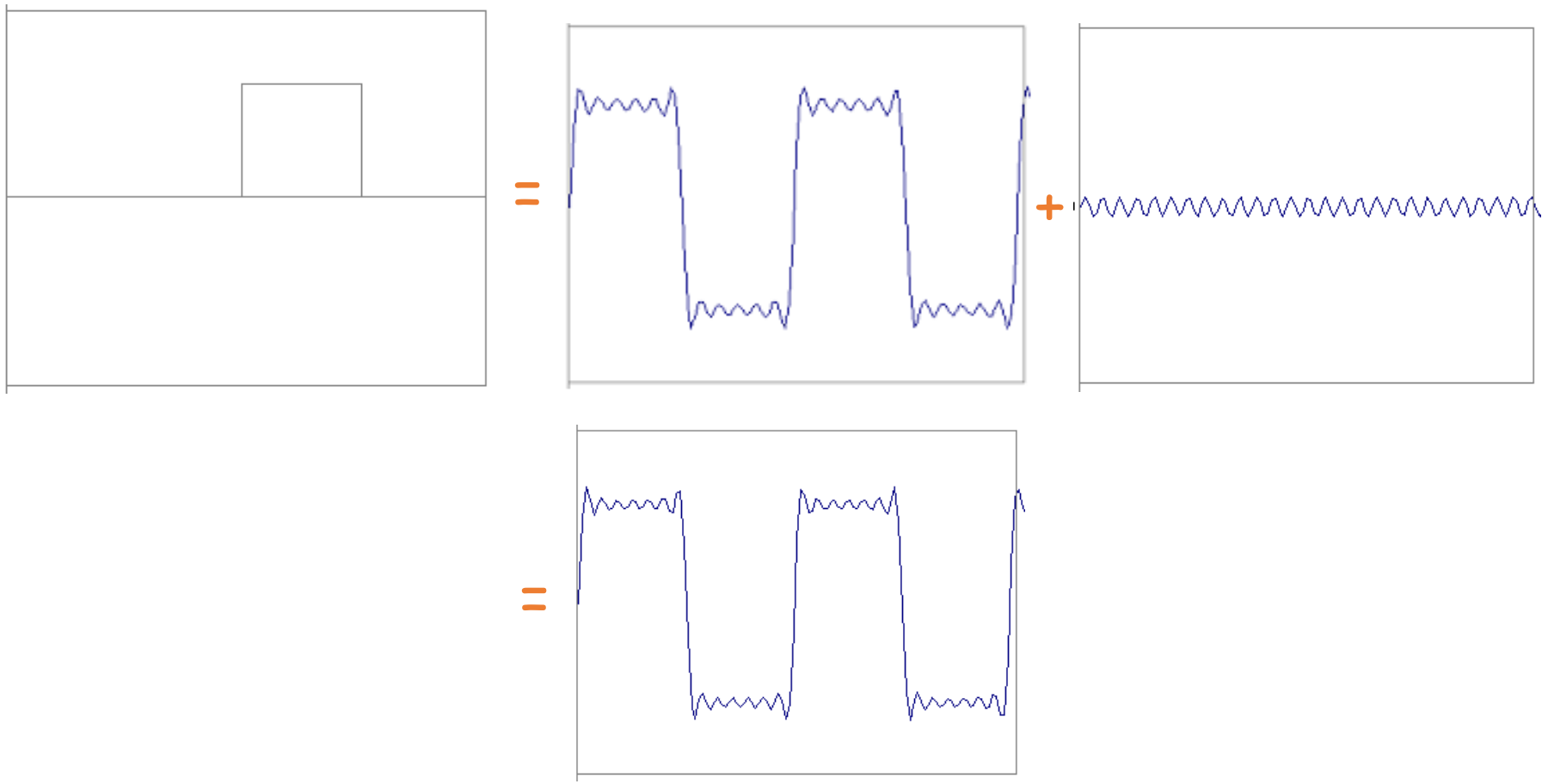
Frequency Spectra



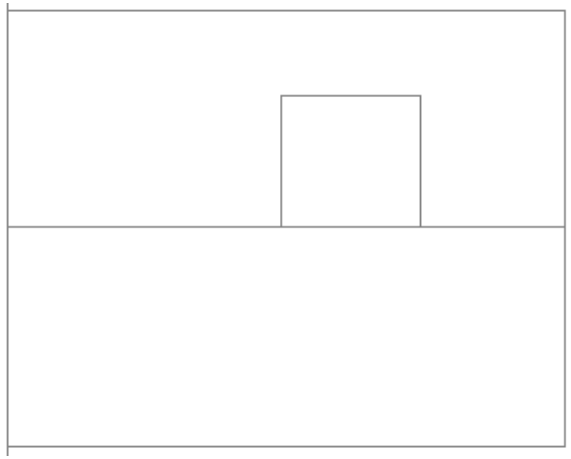
Frequency Spectra



Frequency Spectra

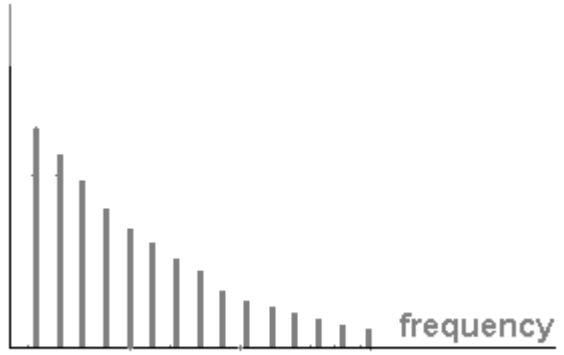


Frequency Spectra

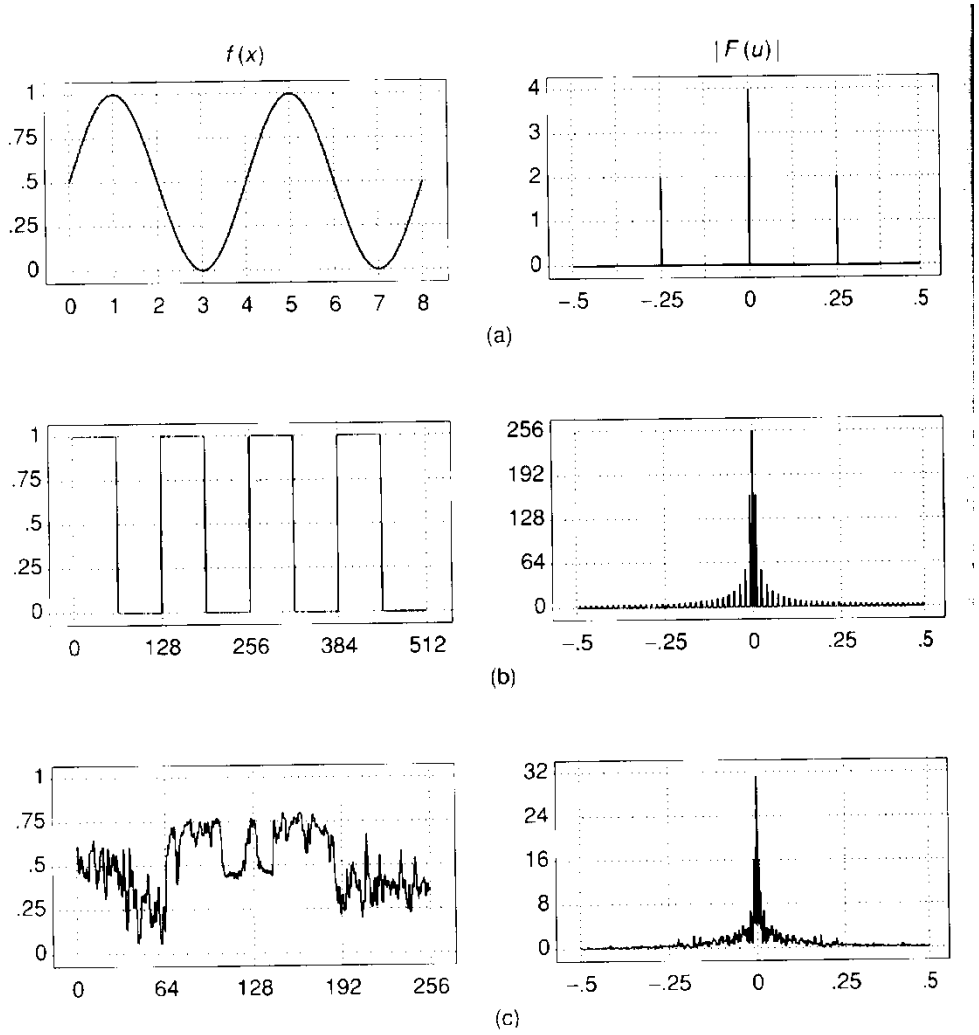


=

$$A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)$$



Frequency Spectra



Fourier Transform – more formally

Represent the signal as an infinite weighted sum of an infinite number of sinusoids

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i 2\pi u x} dx$$

Note: $e^{ik} = \cos k + i \sin k \qquad i = \sqrt{-1}$

Arbitrary function ⟶ Single Analytic Expression

Spatial Domain (*x*) ⟶ Frequency Domain (*u*)

(Frequency Spectrum *F(u)*)

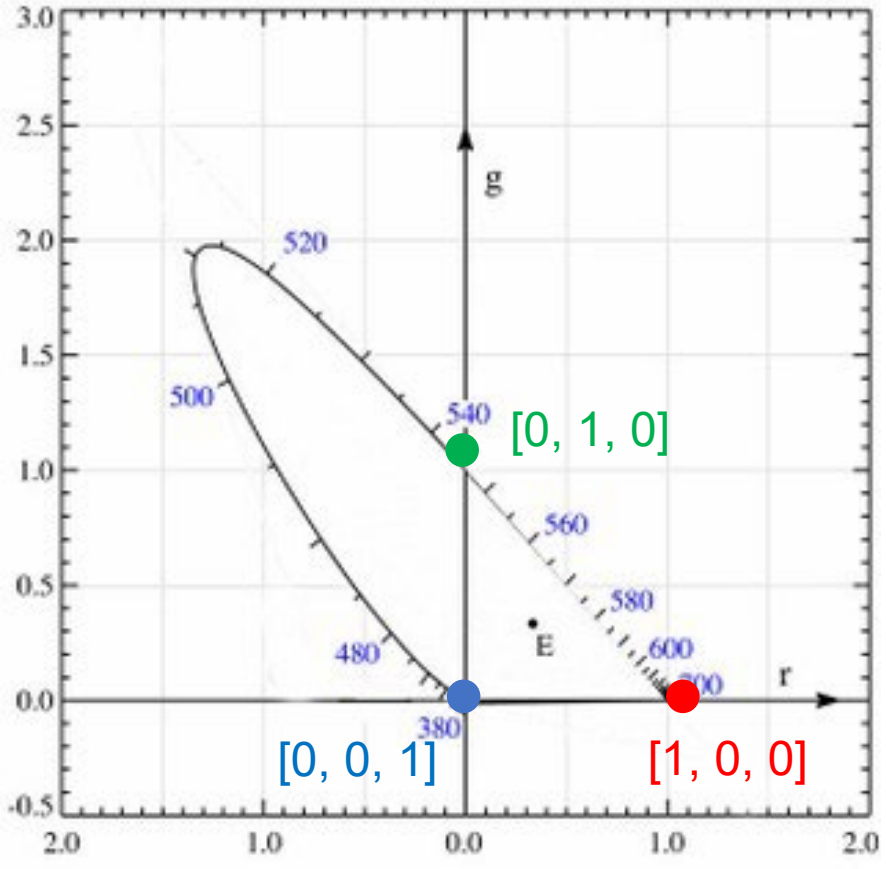
Inverse Fourier Transform (IFT)

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i 2\pi u x} dx$$

Color in Visualization – rg-Chromaticity

- Plot the locus in a 2D r-g plot
- b is implicit (i.e., b = 1-r-g)
- Although this is a 2D plot, it encodes 3D

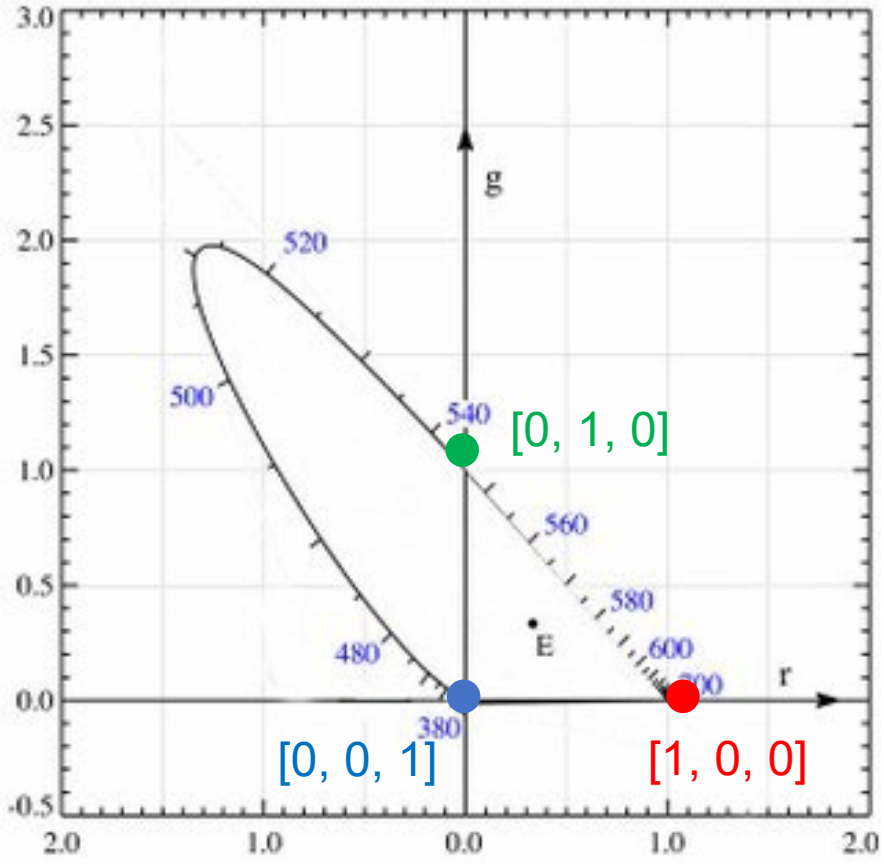
- Points on the curve?
- Connecting two points?
- Inside/outside the locus?



[wikipedia]

Color in Visualization – rg-Chromaticity

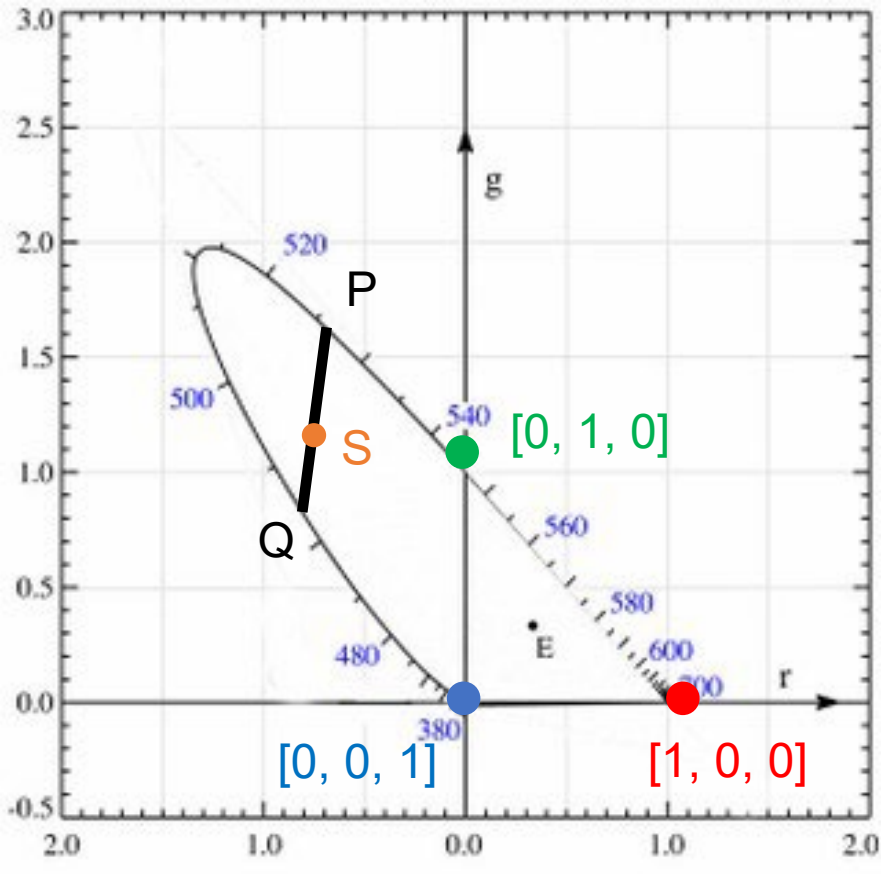
- The **curve** is the **spectral** locus
- Points on **locus**: spectral light



[wikipedia]

Color in Visualization – rg-Chromaticity

Connect two points on the locus: all the colors that are can be generated from combining those two spectral lights

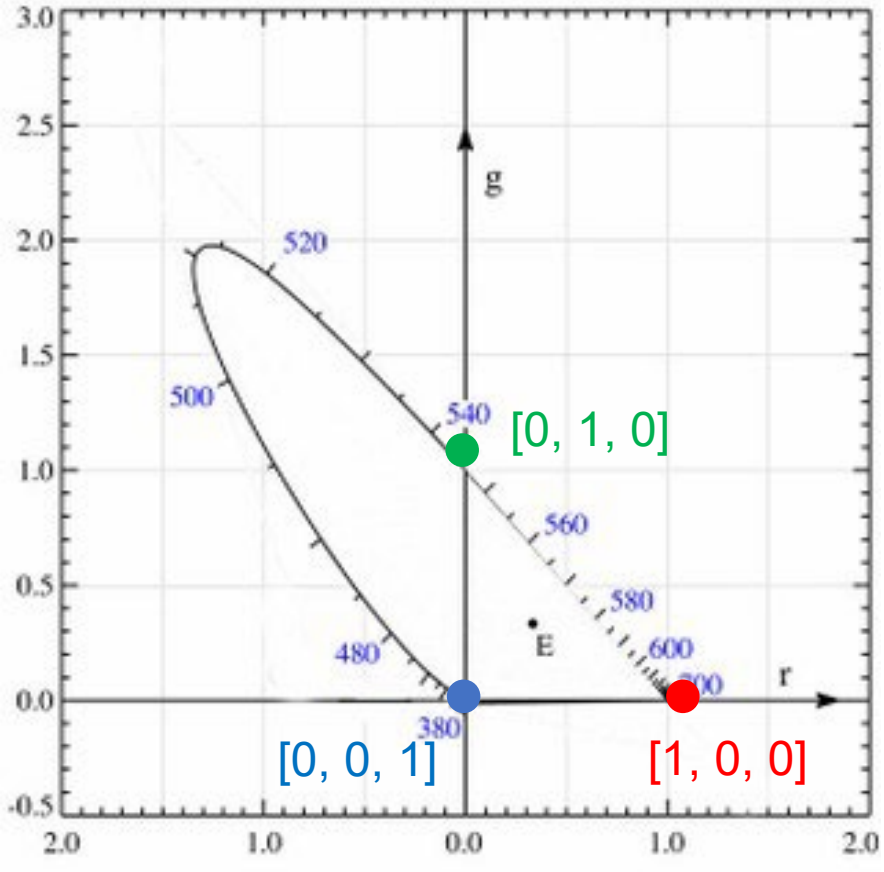


[wikipedia]

Color in Visualization – rg-Chromaticity

Points **inside locus**: all possible colors. They can be produced from some combination of other lights.

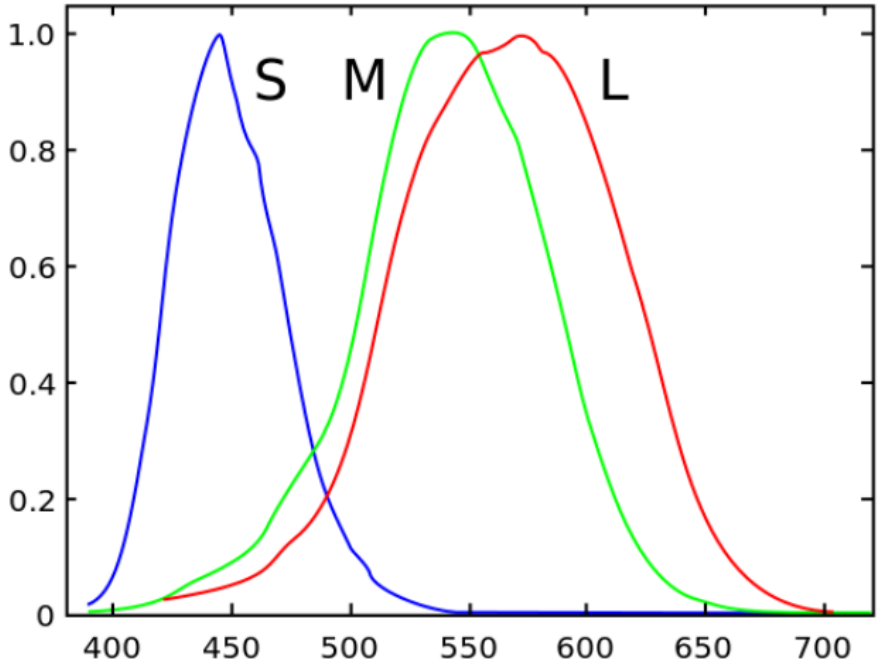
Points **outside locus** are imaginary colors; can't be produced from any combination of real lights.

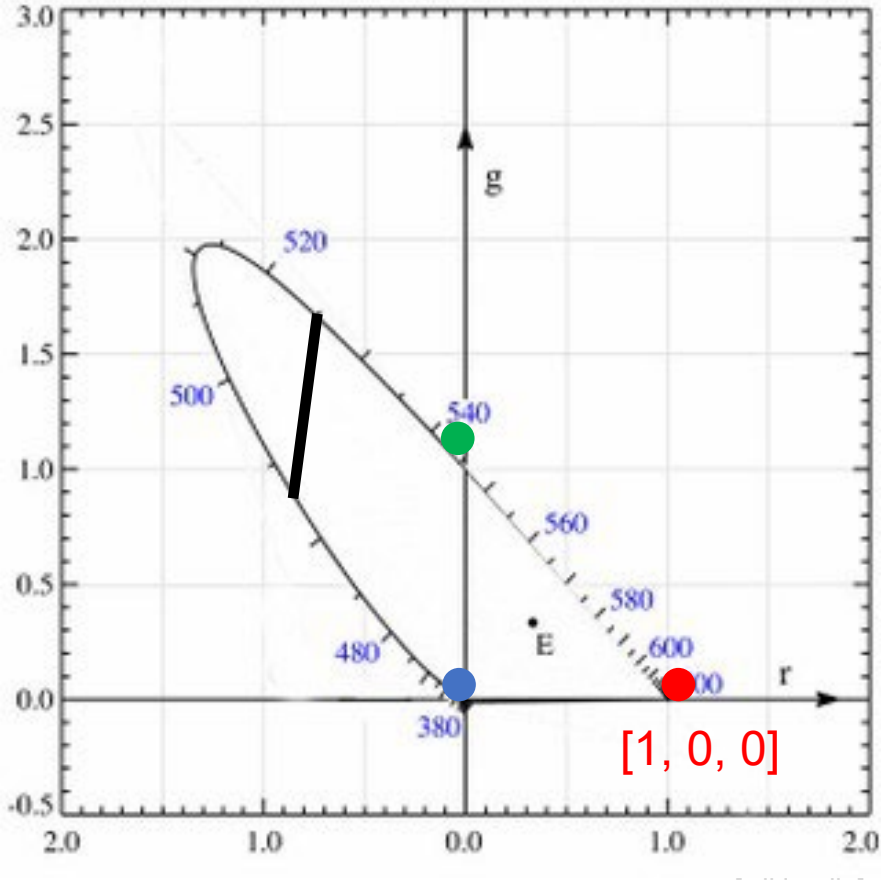


Color in Visualization - Chromaticity

Can the retina/display see/generate all colors?

Not with negative or impossible combination



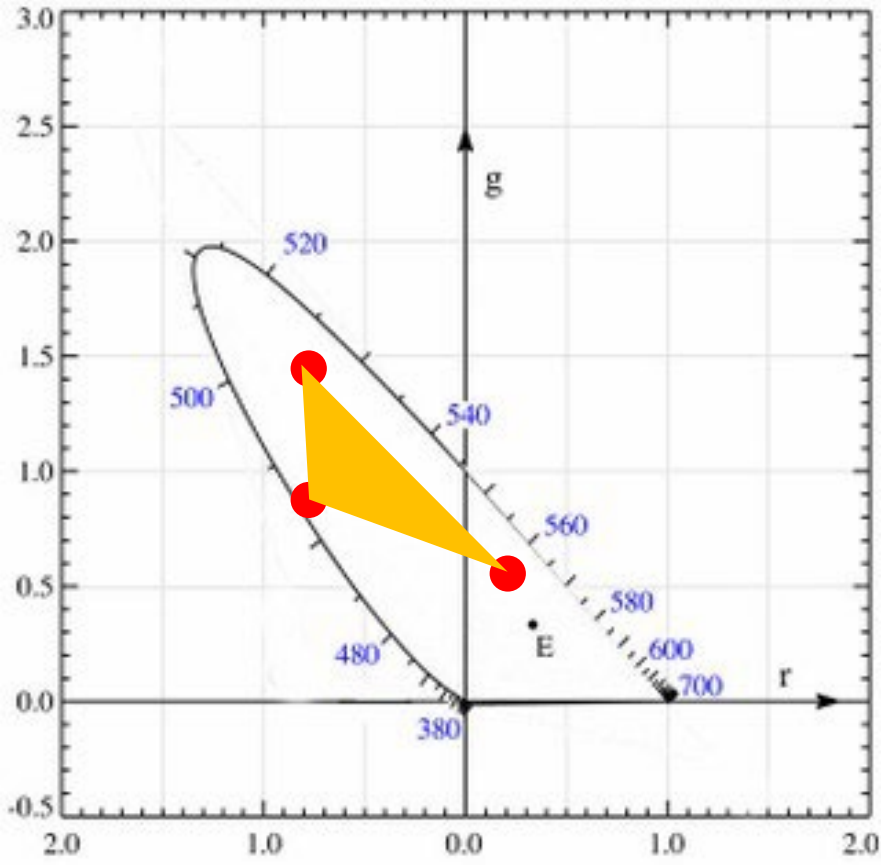


[wikipedia]

Color in Visualization - Gamut

Define **any** color space

- Pick three points on or inside the locus as the primary lights.
  - Primary lights need not be spectral lights
- A point inside the triangle is a color that can be produced in this color space
  - Every point inside a triangle can be expressed as a linear combination



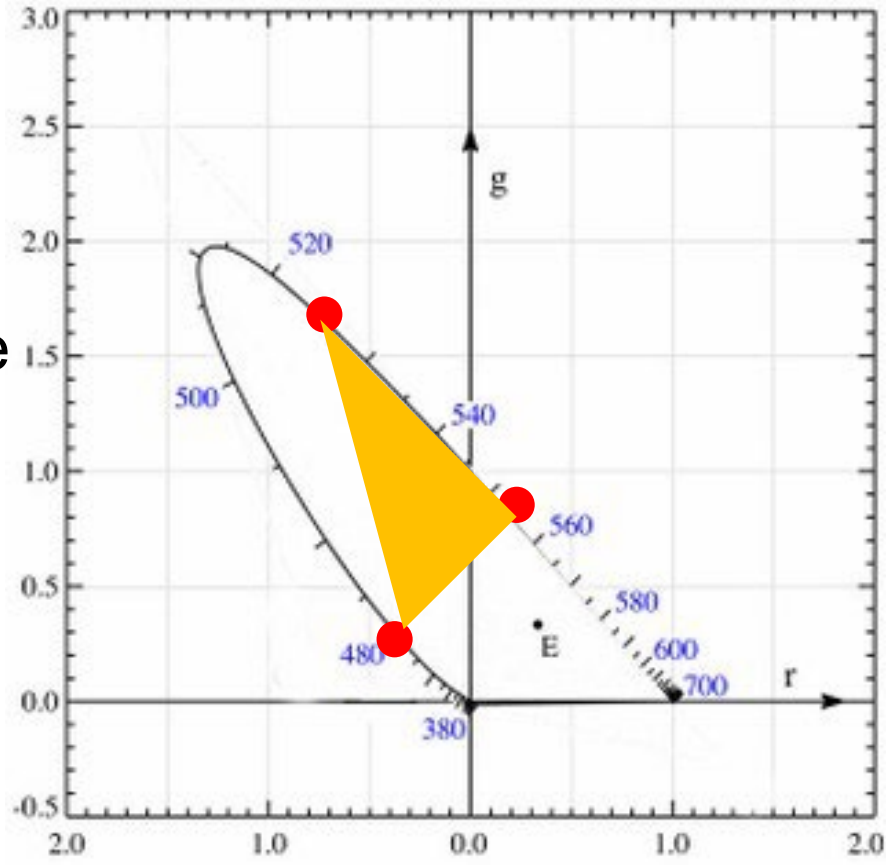
[wikipedia]

Color in Visualization - Gamut

The triangle is called the **gamut**

All the colors that can be produced in this color space

A gamut is **3D**



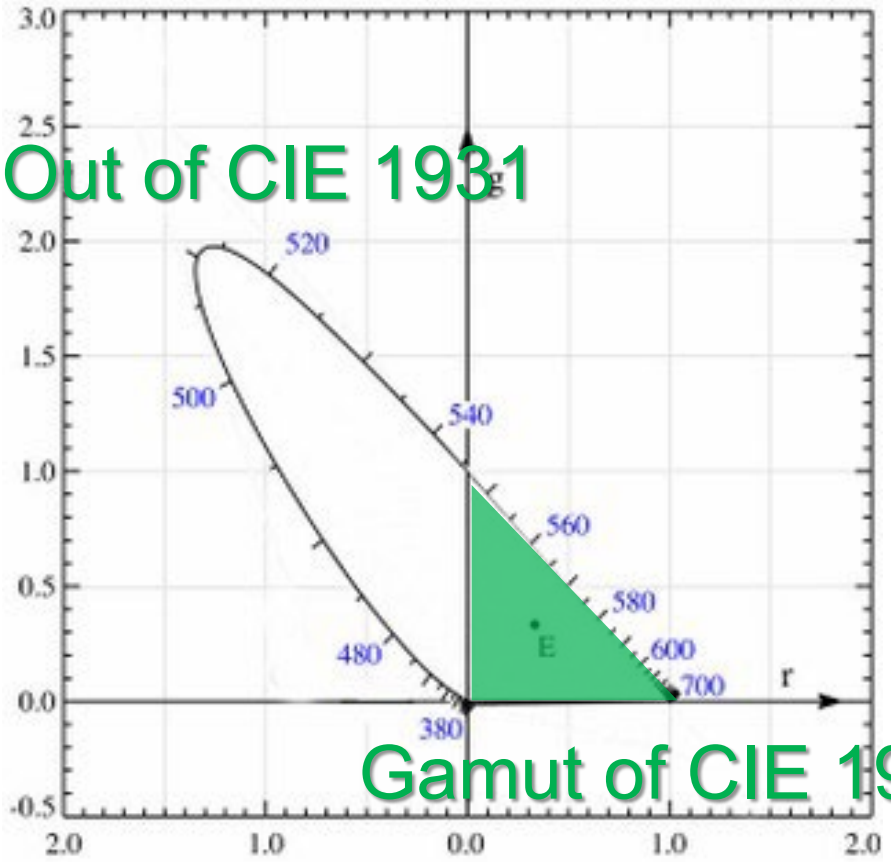
[wikipedia]

Gamut of Color Space

Some colors are expressed in negative RGB values, which is mathematically OK but not very intuitive.

Goal: design a color space so that all visible colors (point on and inside the spectral locus) have positive values.

Better Design?

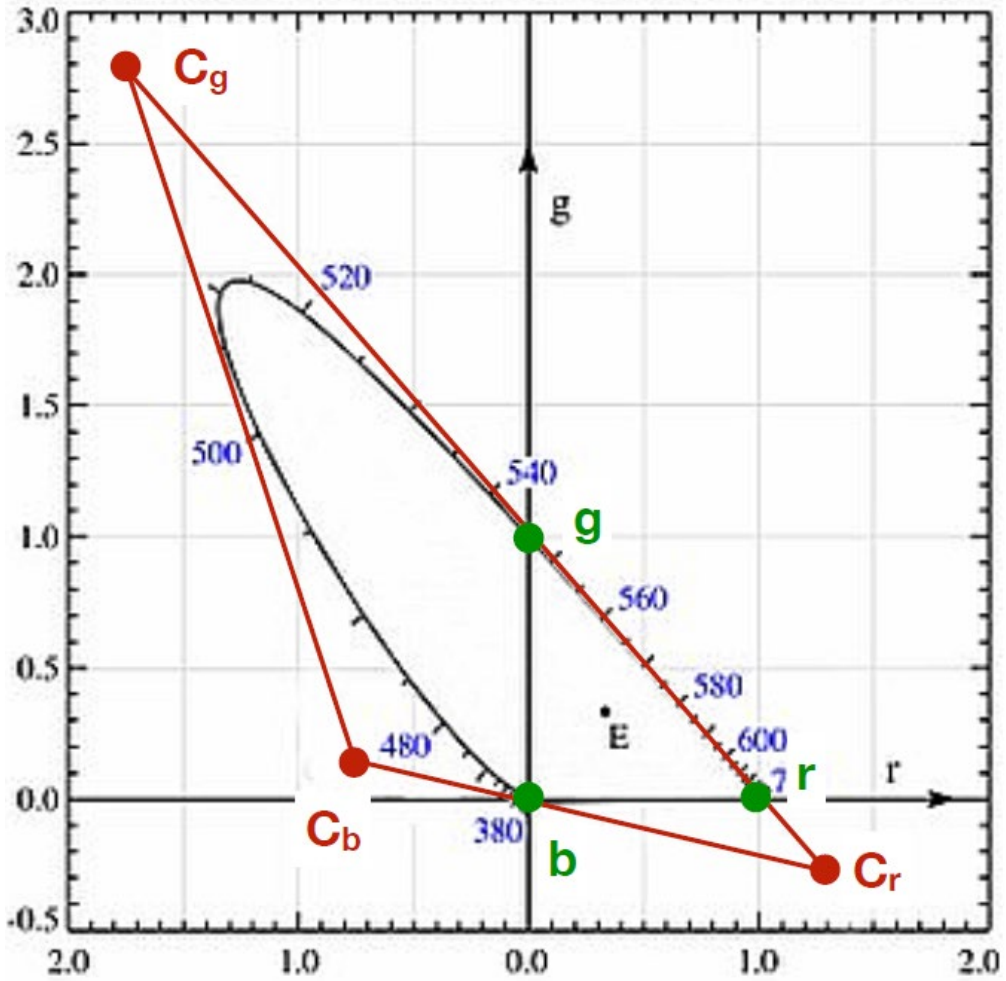


[wikipedia]

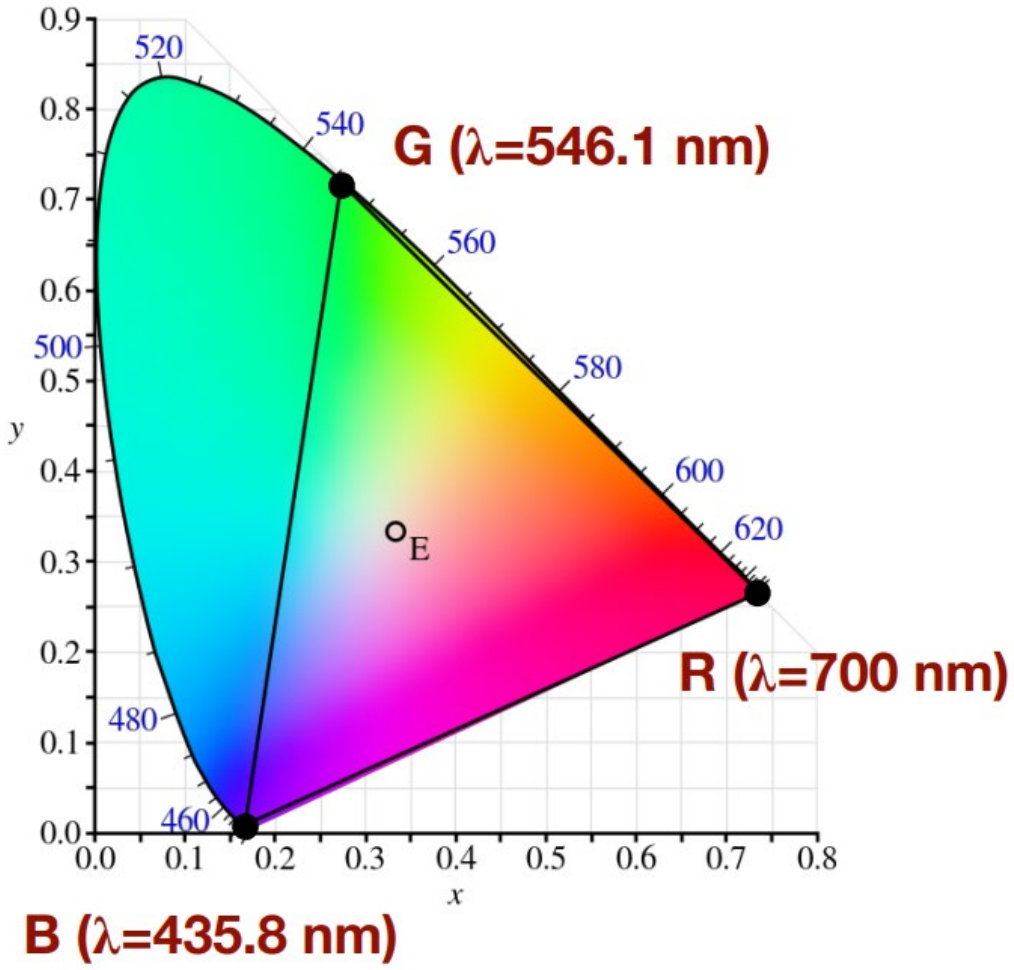
XYZ Color Space

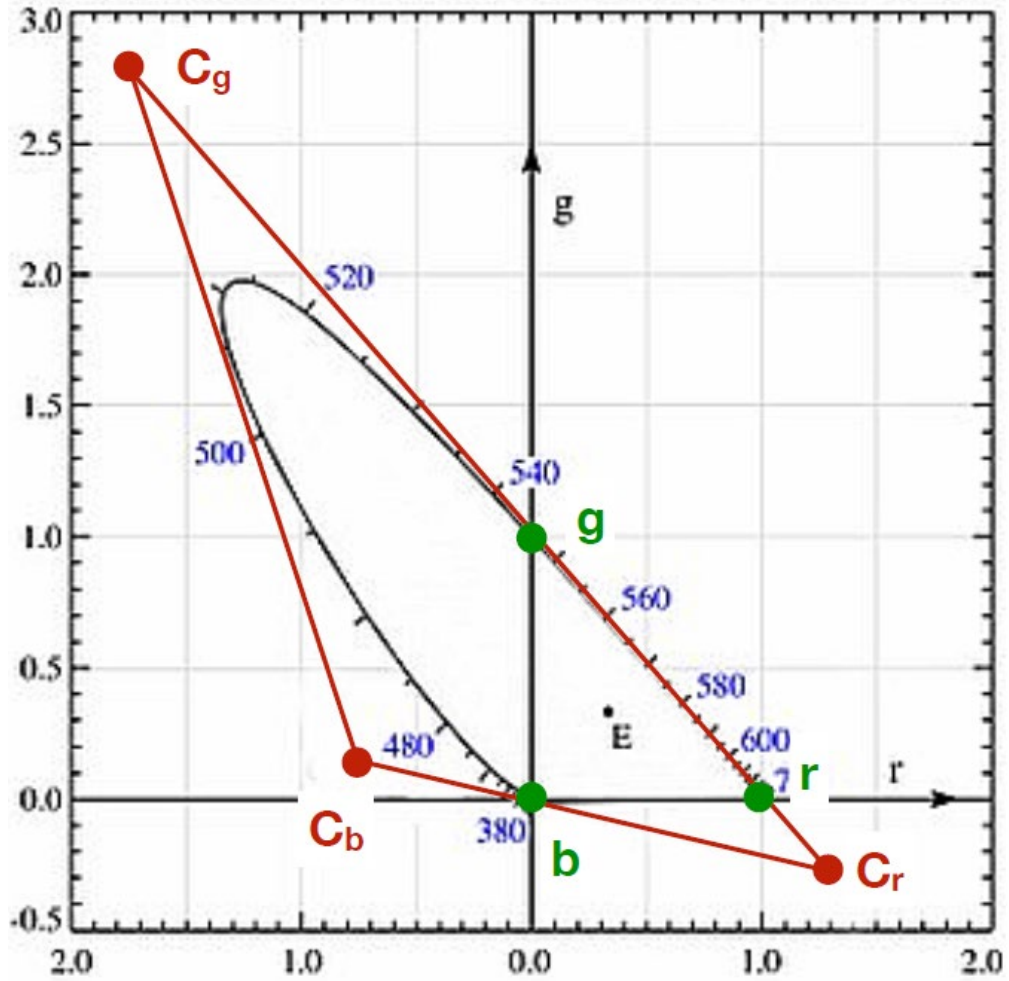
Pick primary lights in such a way that the triangle covers the entire spectral locus.

Make the chromaticities of those primary lights [0, 0, 1], [1, 0, 0], and [1, 0, 0].



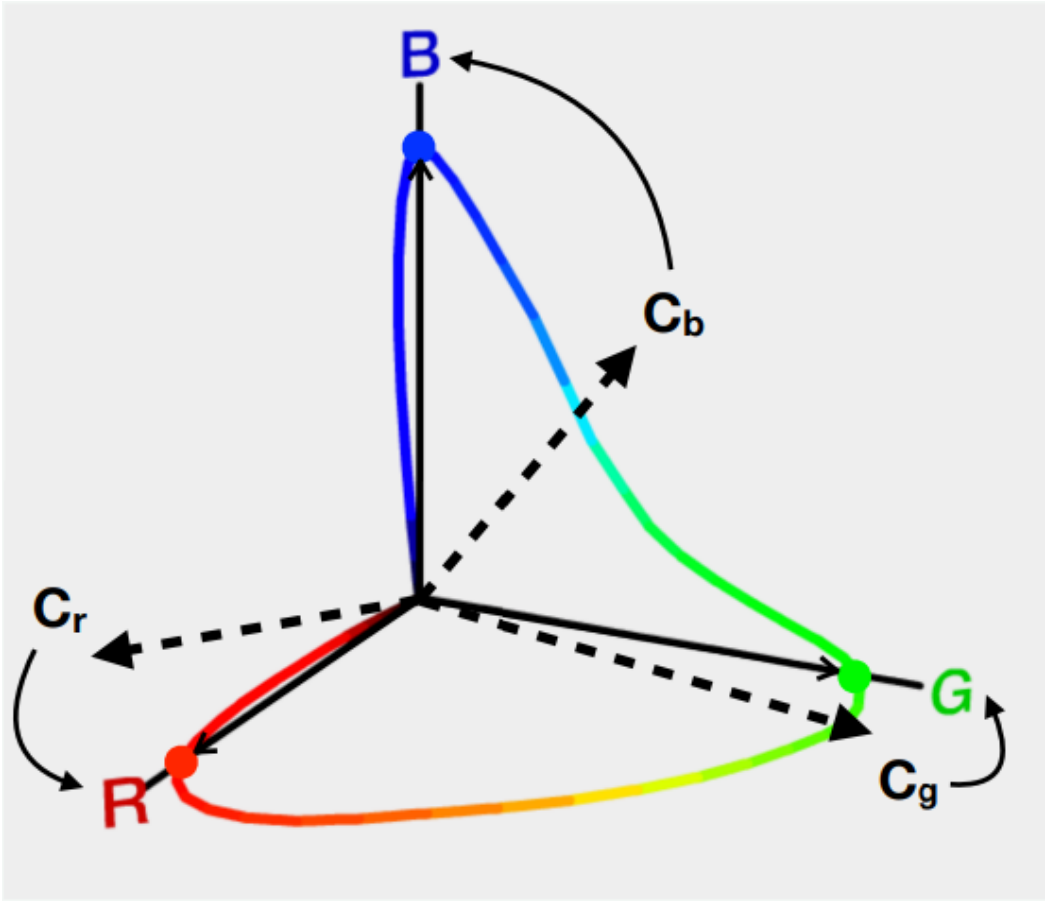
XYZ Color Space





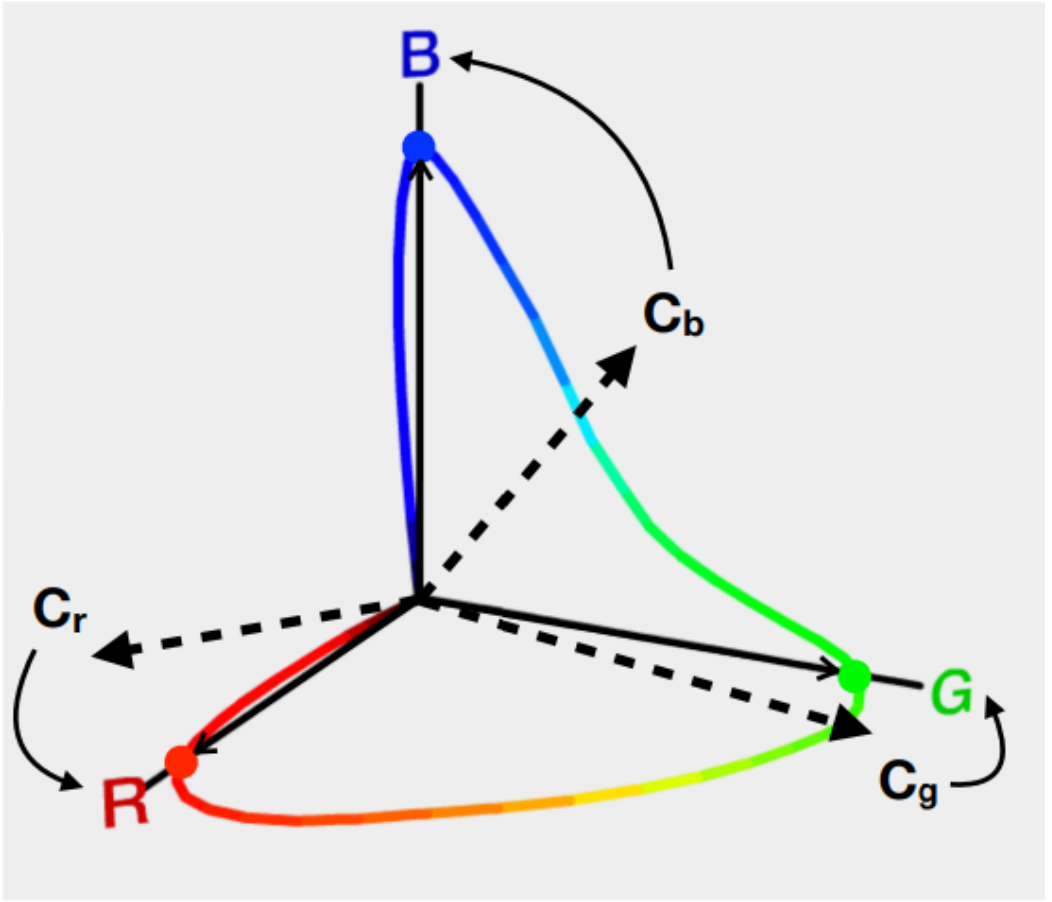
XYZ Color Space

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} R_x, G_x, B_x \\ R_y, G_y, B_y \\ R_z, G_z, B_z \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

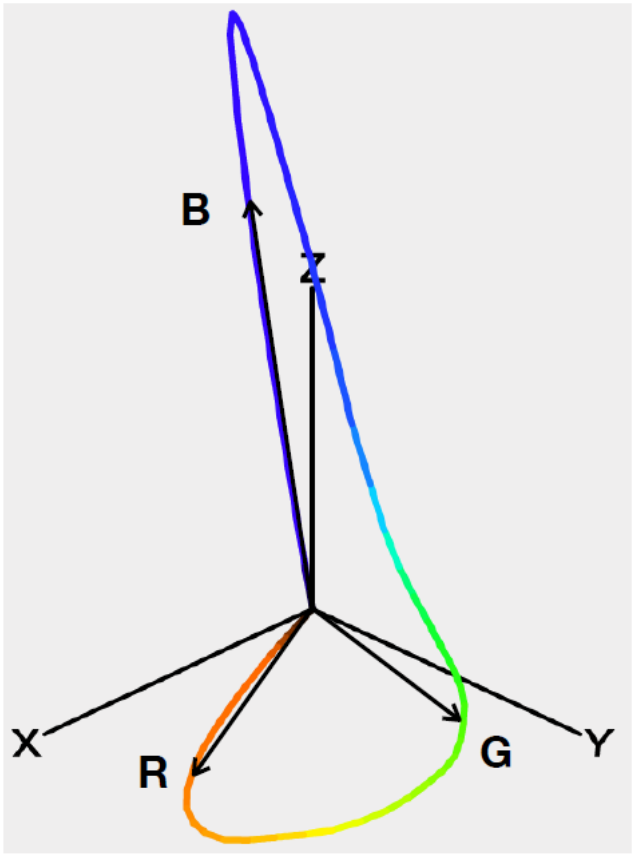


XYZ Color Space

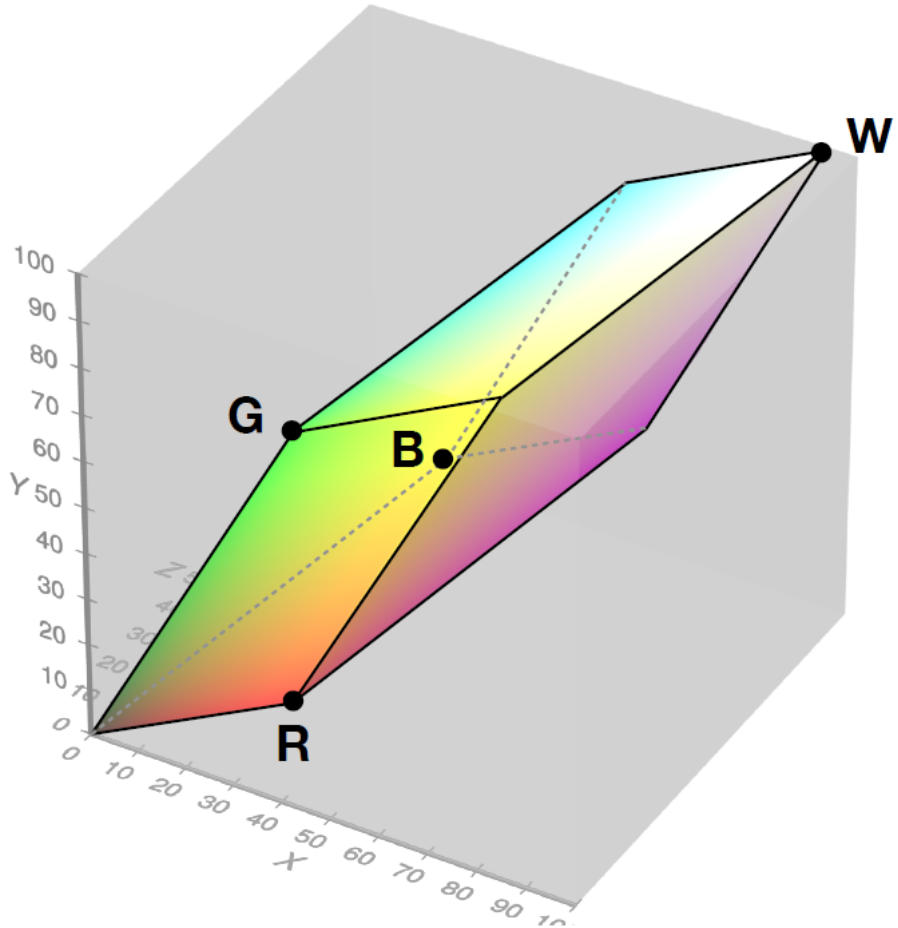
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{b_{21}} \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \frac{1}{0.176\,97} \begin{bmatrix} 0.490\,00 & 0.310\,00 & 0.200\,00 \\ 0.176\,97 & 0.812\,40 & 0.010\,63 \\ 0.000\,00 & 0.010\,00 & 0.990\,00 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$



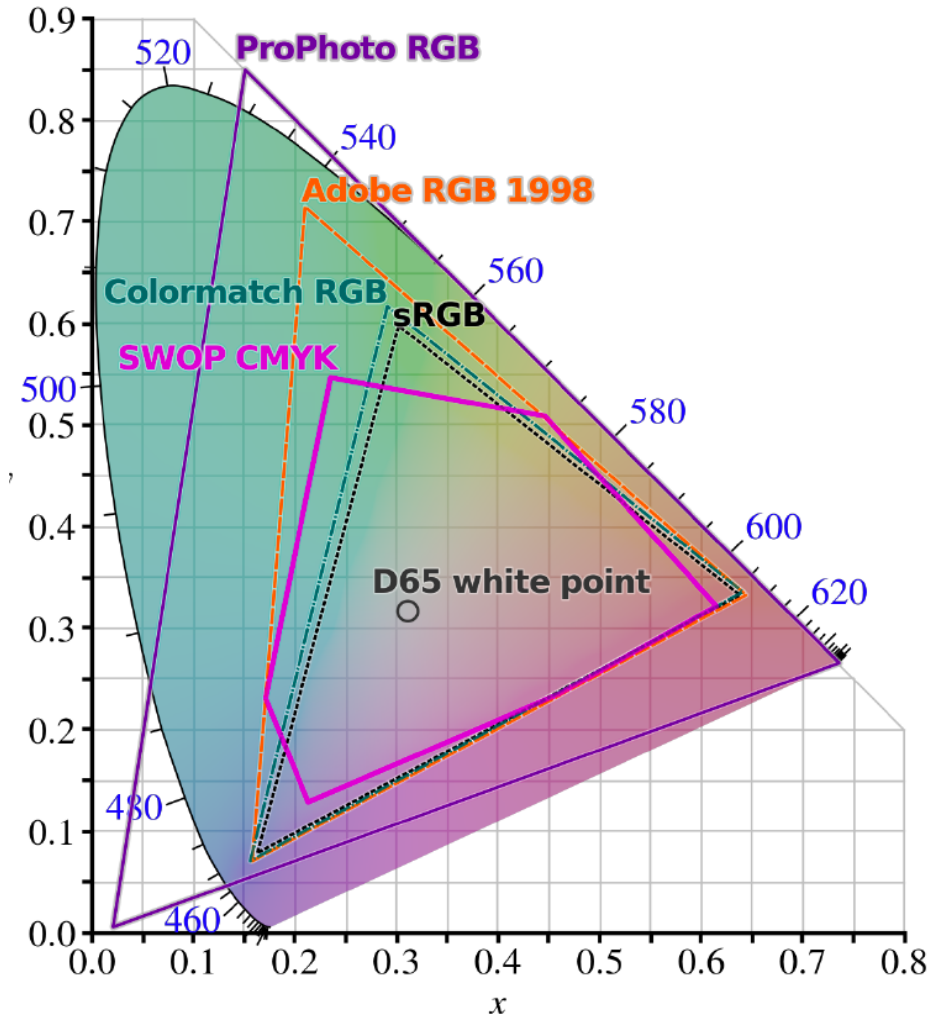
Color in Visualization - Gamut



CIE 1931 XYZ Space



Color in Visualization - Gamut

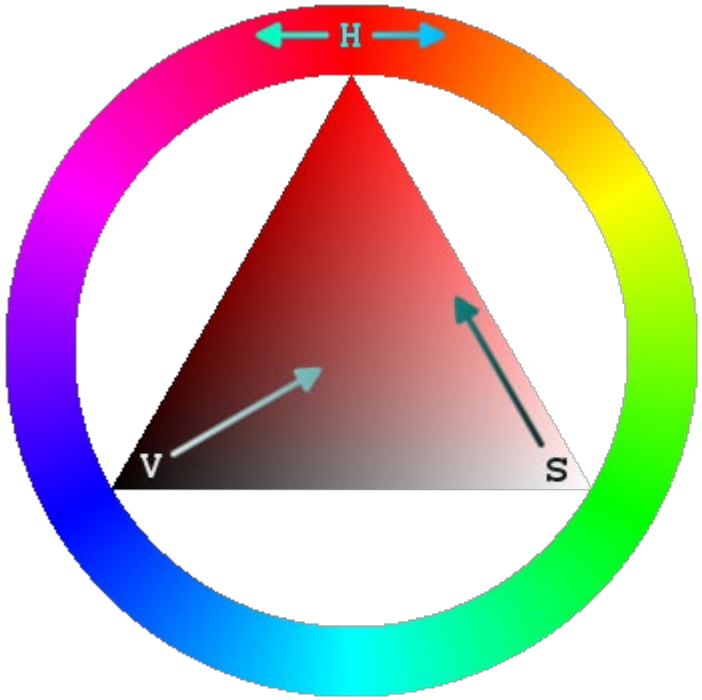


ProPhoto RGB has a wide gamut, but ~13% of is imaginary. Two of the three primaries are not physically realizable

https://www.msi.com/blog/why-dci-p3-is-the-new-standard-of-color-gamu

Case Study: HSV vs. RGB

Hue, Saturation, Lightness



$h \in [0, 360)$ $s, l \in [0,1]$

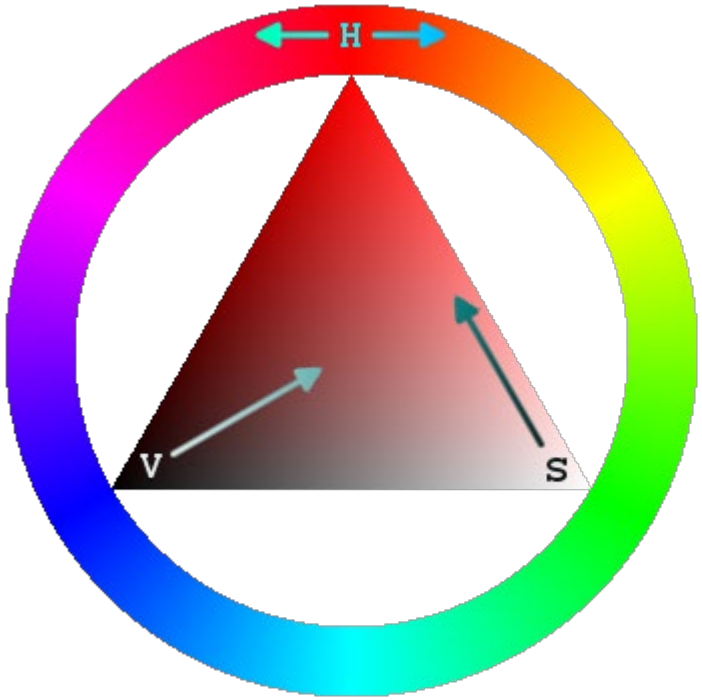
$$h = \begin{cases} 0^\circ & \text{if } max = min \\ 60^\circ \times \frac{g-b}{max-min} + 0^\circ, & \text{if } max = r \text{ and } g \geq b \\ 60^\circ \times \frac{g-b}{max-min} + 360^\circ, & \text{if } max = r \text{ and } g < b \\ 60^\circ \times \frac{b-r}{max-min} + 120^\circ, & \text{if } max = g \\ 60^\circ \times \frac{r-g}{max-min} + 240^\circ, & \text{if } max = b \end{cases}$$

$$s = \begin{cases} 0, & \text{if } max = 0 \\ \frac{max-min}{max} = 1 - \frac{min}{max}, & \text{otherwise} \end{cases}$$

$$v = max$$

Case Study: HSV vs. RGB

Hue, Saturation, Lightness



$$h_i = \left\lfloor \frac{h}{60} \right\rfloor$$

$$f = \frac{h}{60} - h_i$$

$$p = v \times (1 - s)$$

$$q = v \times (1 - f \times s)$$

$$t = v \times (1 - (1 - f) \times s)$$

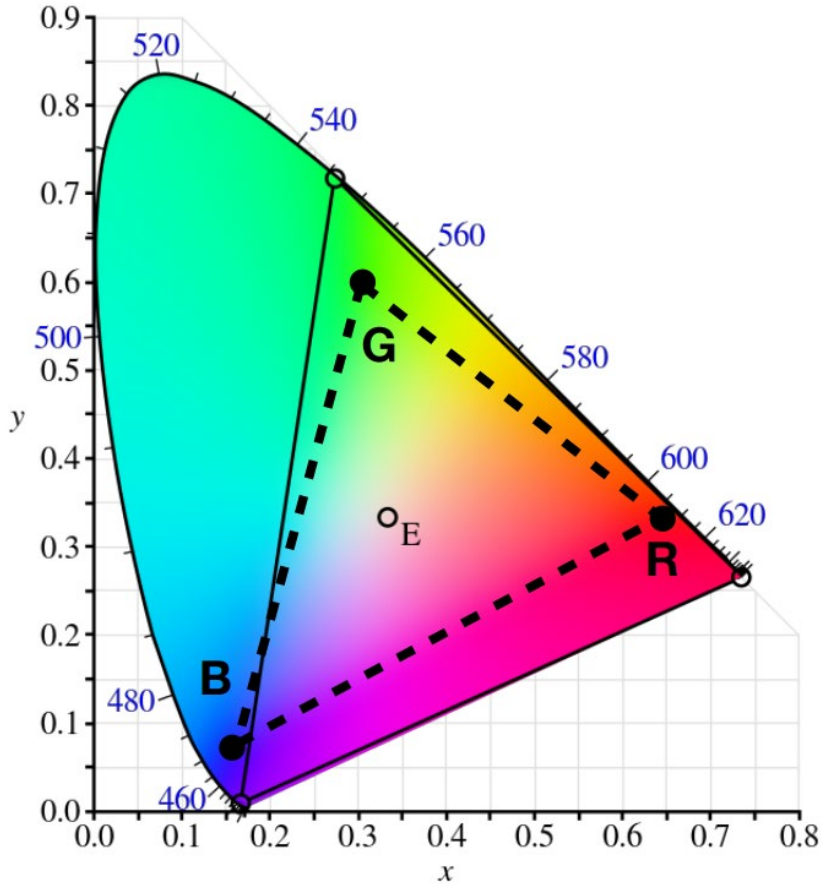
$$(r, g, b) = \begin{cases} (v, t, p), & \text{if } h_i = 0 \\ (q, v, p), & \text{if } h_i = 1 \\ (p, v, t), & \text{if } h_i = 2 \\ (p, q, v), & \text{if } h_i = 3 \\ (t, p, v), & \text{if } h_i = 4 \\ (v, p, q), & \text{if } h_i = 5 \end{cases}$$

Design Your Own Color Space

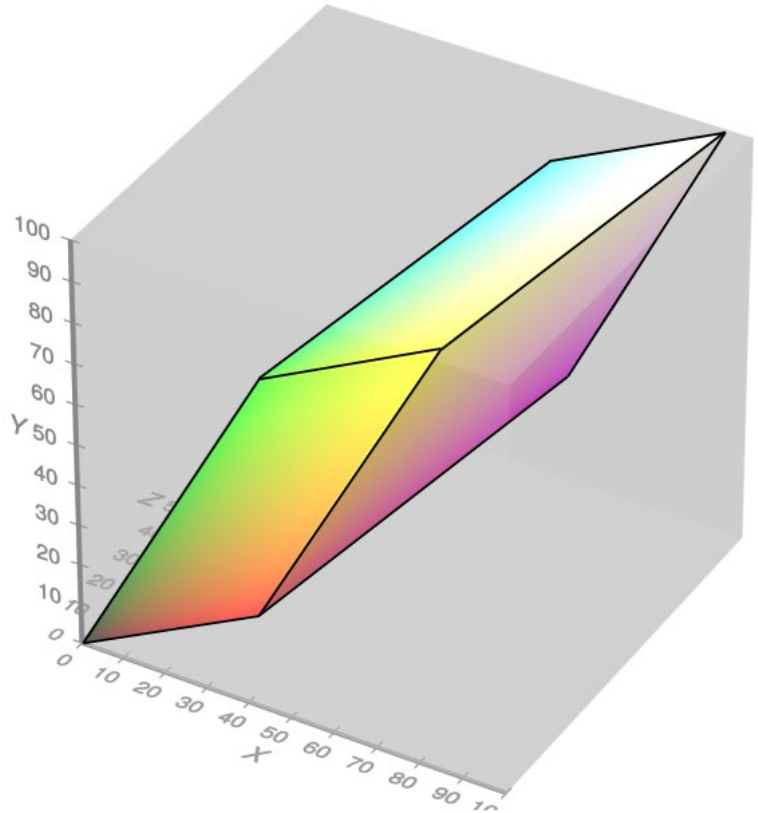
Step 1: Pick Primaries
Step 2: Pick “White”
Step 3: Specify the “Intensity” of White
Step 4: Map the Gamut to a Color Cube
Step 5: Gamma Correction

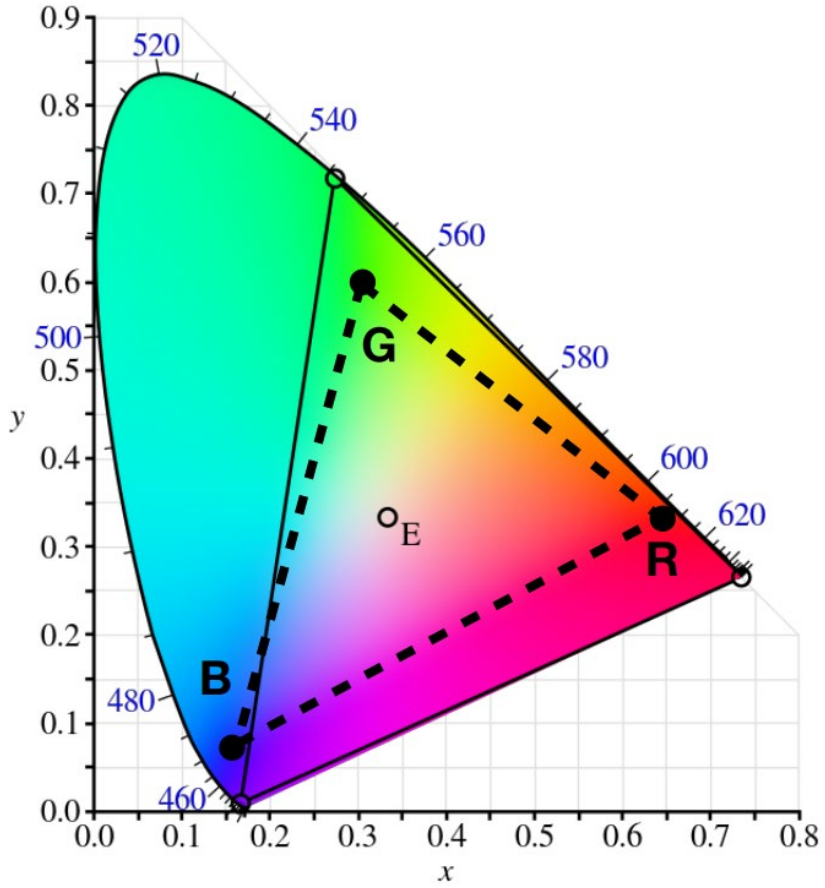
Design Your Own Color Space – S1: Primaries

Are primitives enough to define the space?

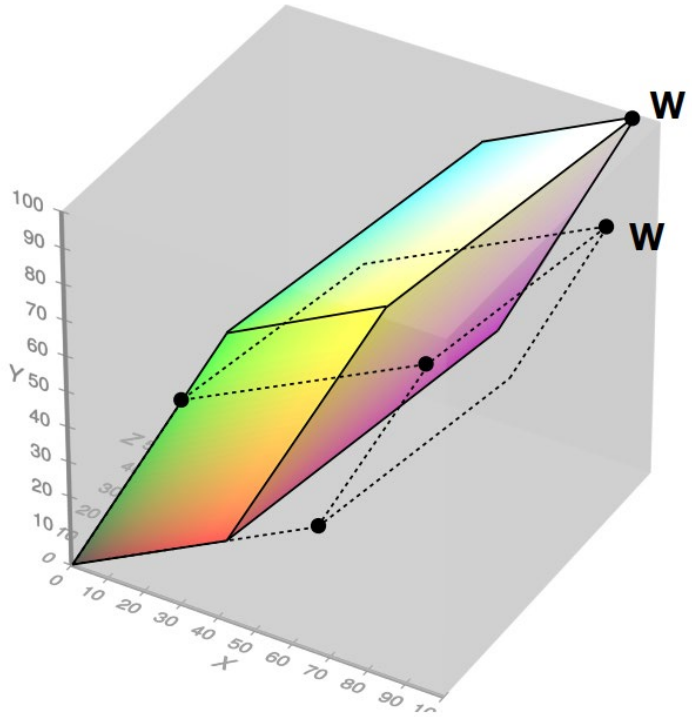


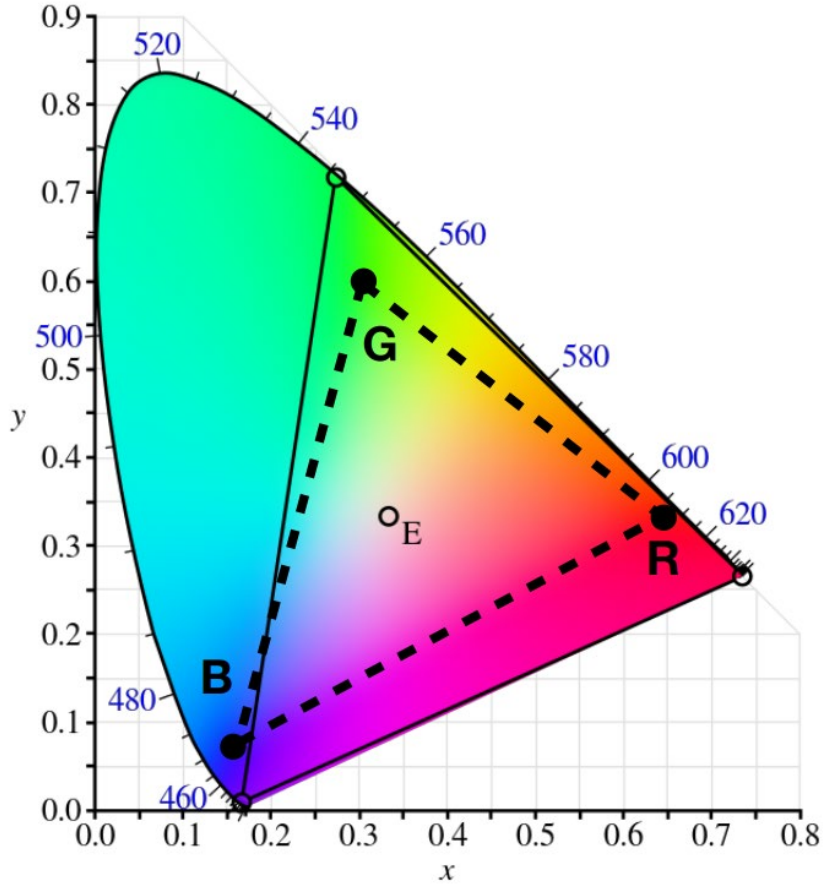
Design Your Own Color Space – S1: Primaries





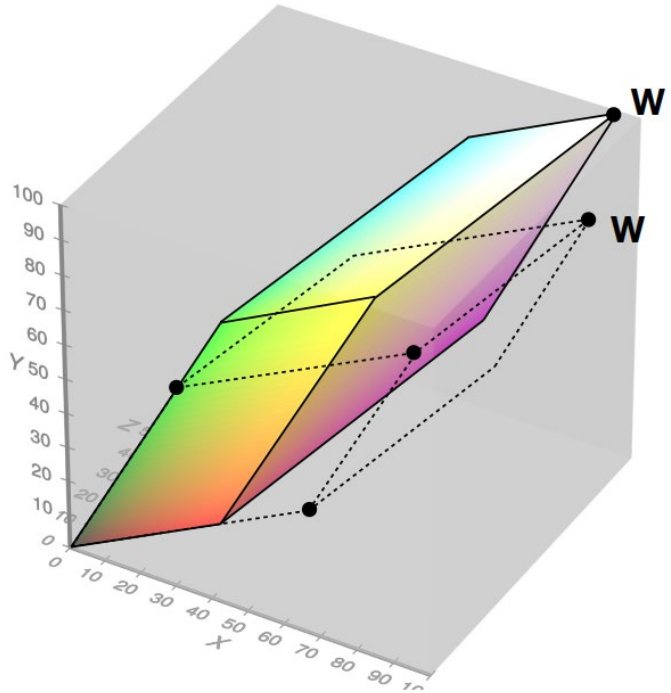
Design Your Own Color Space – S2: Def “White”

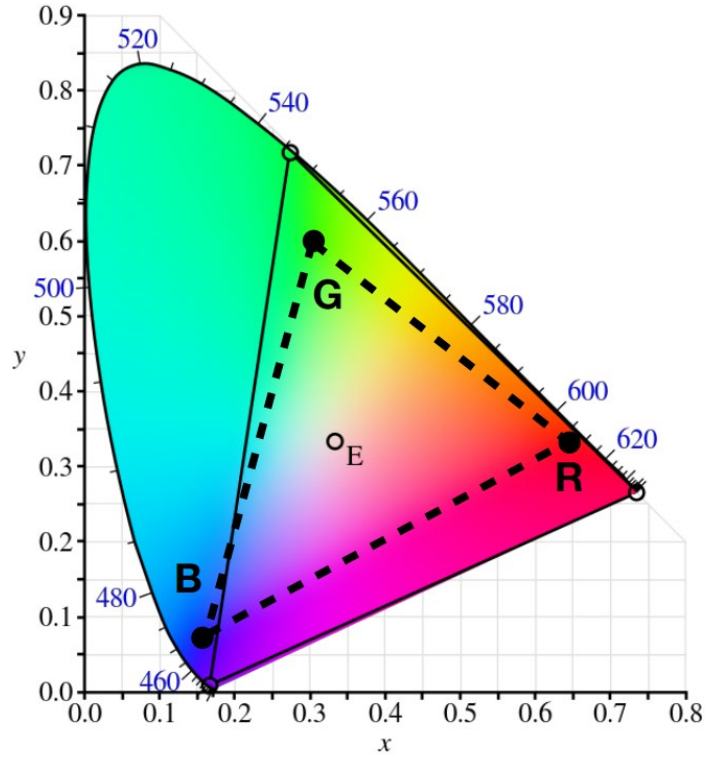




Design Your Own Color Space – S2: Def “White”

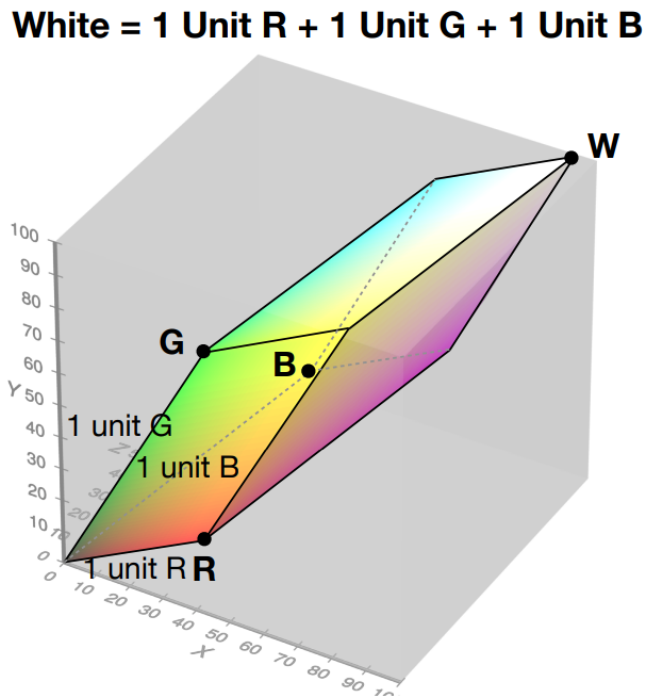
Empirically: equal amount of maximum primary lights in a color space is white.
Scientifically: there are many “white” colors.
They refer to spectra emitted by black-bodies under different temperatures.
A color space has to pick what its reference white is.

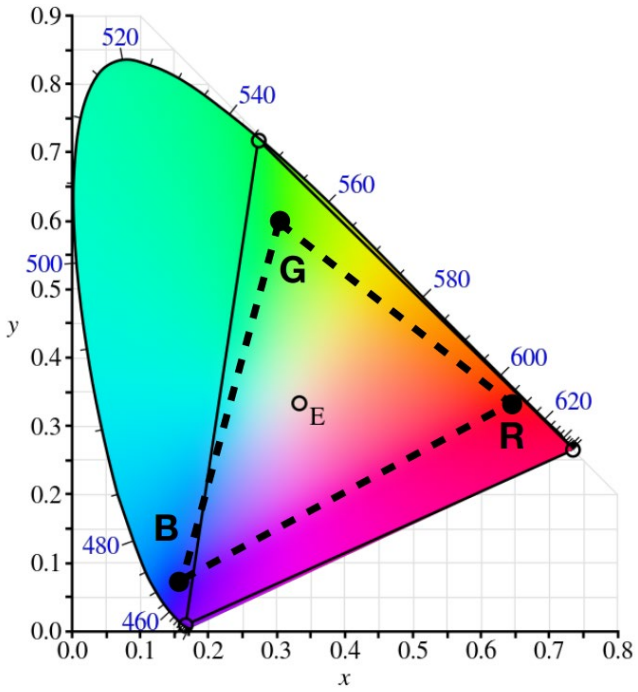




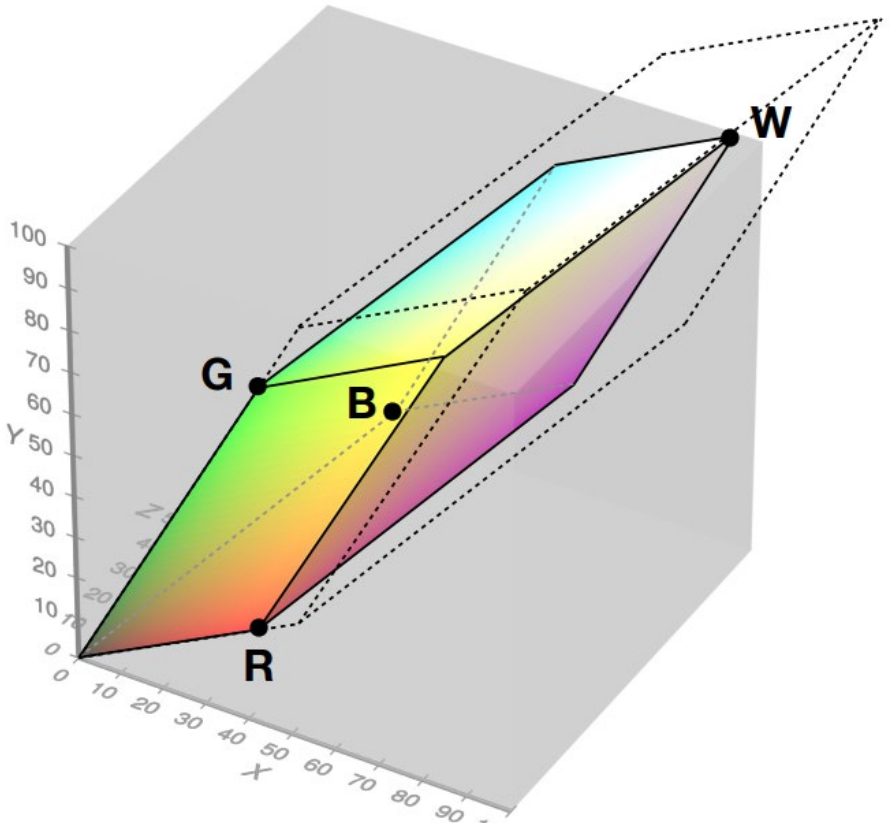
Design Your Own Color Space – S2: Def “White”

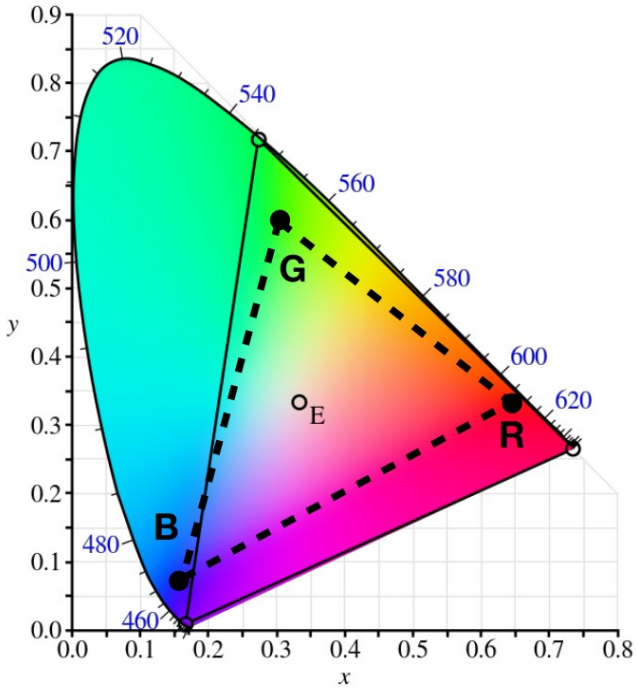
Empirically: equal amount of maximum primary lights in a color space is white.
Scientifically: there are many “white” colors.
They refer to spectra emitted by black-bodies under different temperatures.
A color space has to pick what its reference white is.



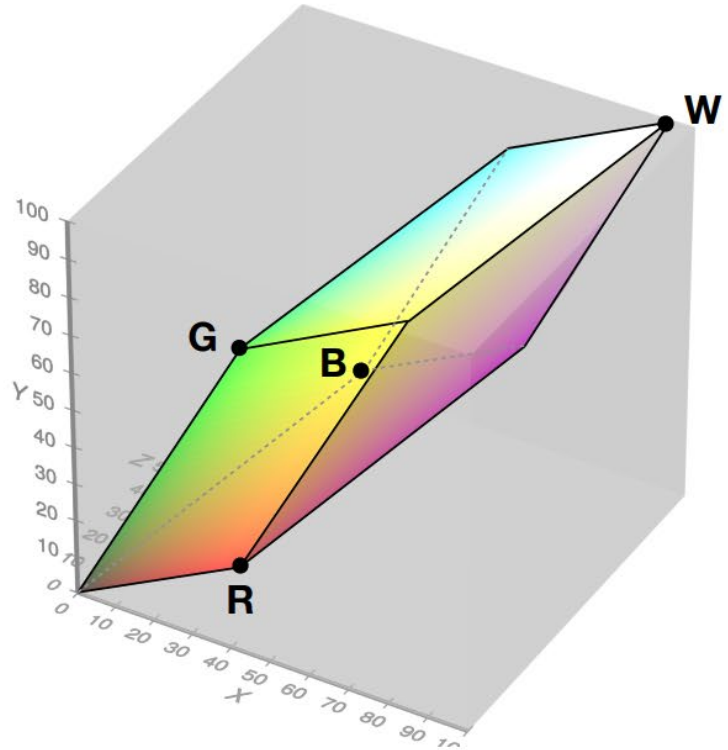


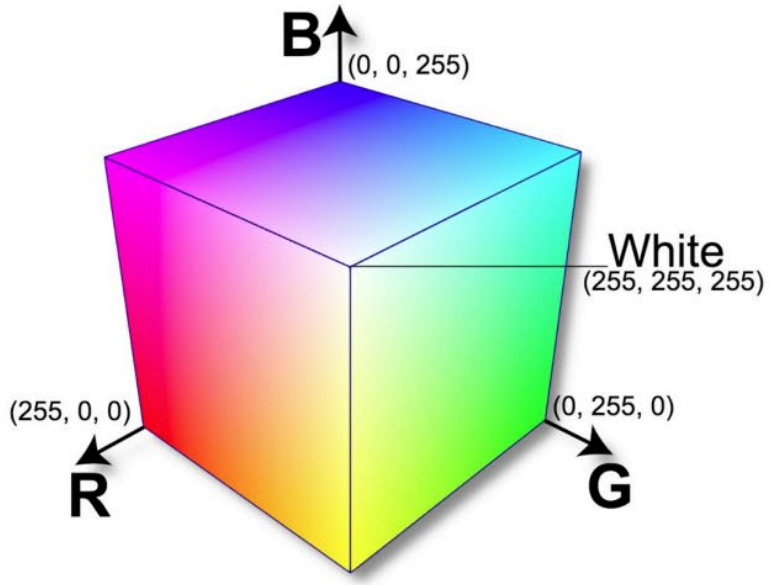
Design Your Own Color Space – S3: Def Intensity





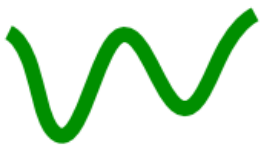
Design Your Own Color Space – S4: Mapping



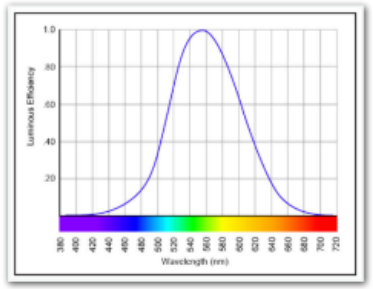


Design Your Own Color Space – S5: Brightness Gamma Correction

Light spectrum (power) $\Phi(\lambda)$

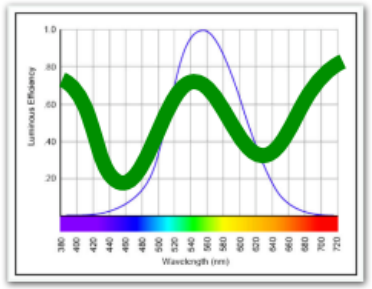


→ LEF V(λ)



→ Cone responses

$$Y(\lambda) \propto \int \Phi(\lambda) V(\lambda) d\lambda$$

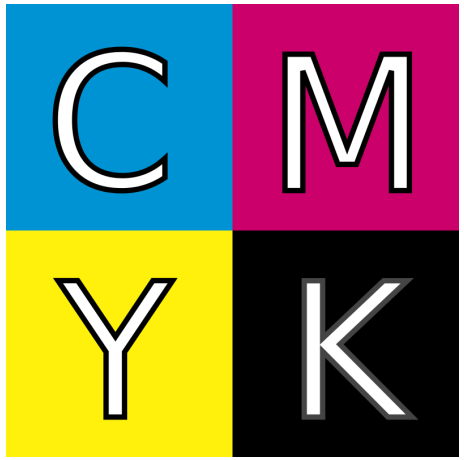


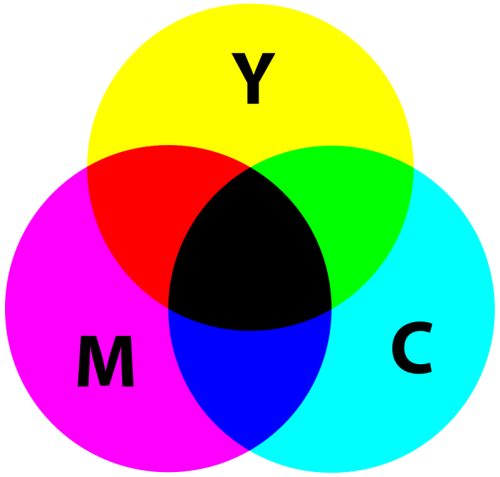
→ Visual Cortex → Brightness/ Lightness (L)

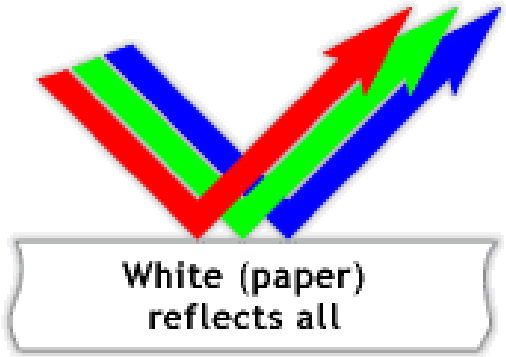
Dark-adapted eye: $L = Y^{0.4}$

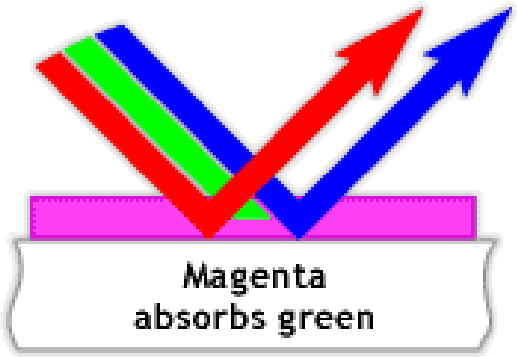
Bright-adapted eye: $L = Y^{0.5}$

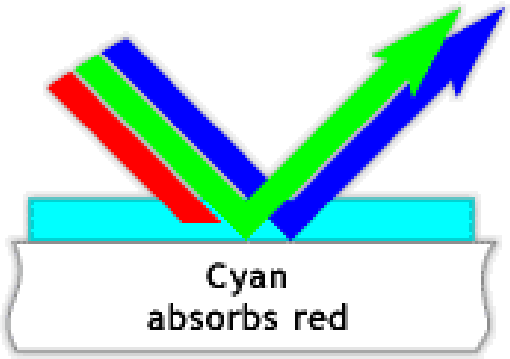
Additive vs. Subtractive Color Space

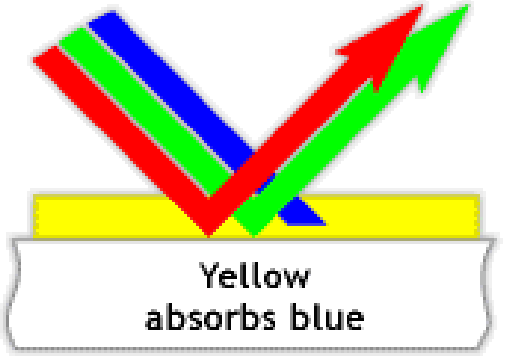












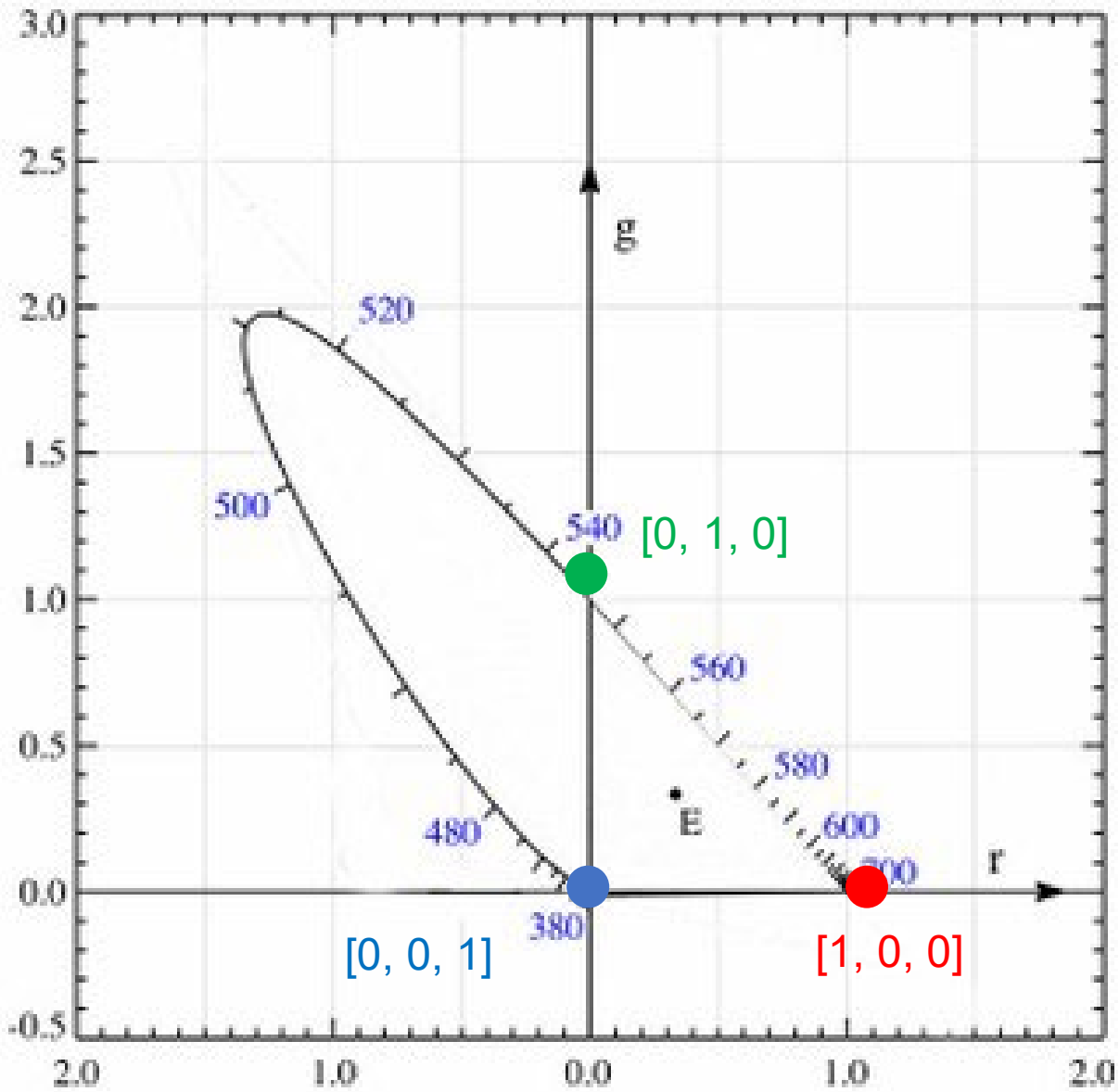
Where We Are

- **Basic mathematical tools**
- **Color: definition, space, and map**
- 2D visualization
- Human perception of 3D
- 3D visualization
- Application case studies

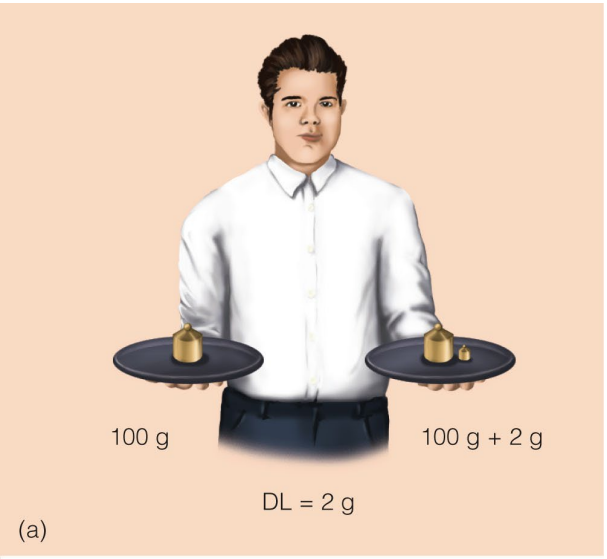
Definition

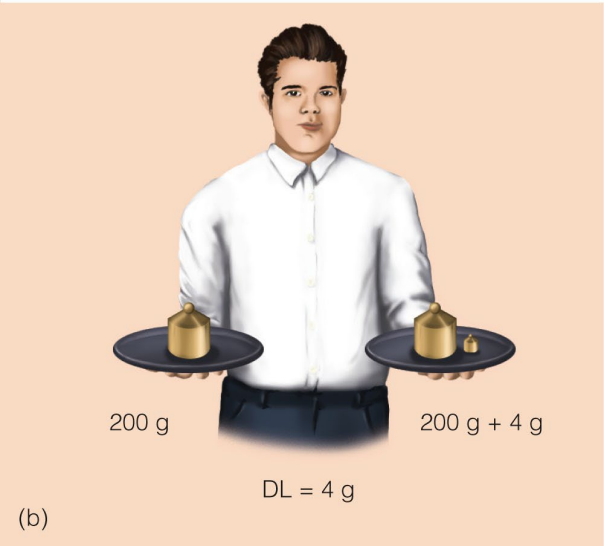
Metamerism & Perception

Chromaticity & Gamut


3.0
2.5
2.0
1.5
1.0
0.5
0.0
-0.5
2.0
1.0
0.0
1.0
2.0
g
r
520
500
540
560
580
600
480
380
E
[0, 1, 0]
[0, 0, 1]
[1, 0, 0]

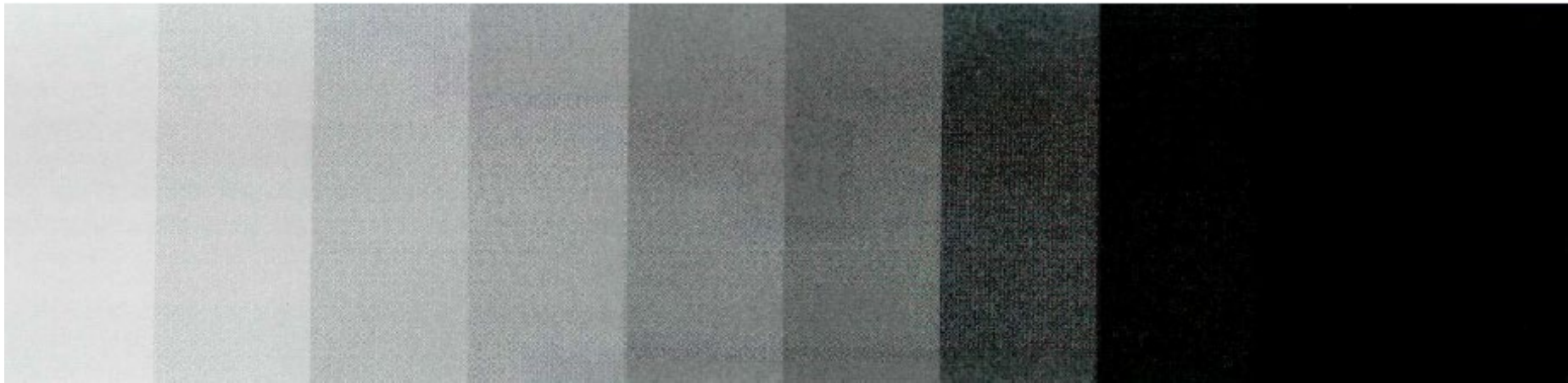
Gamma Correction



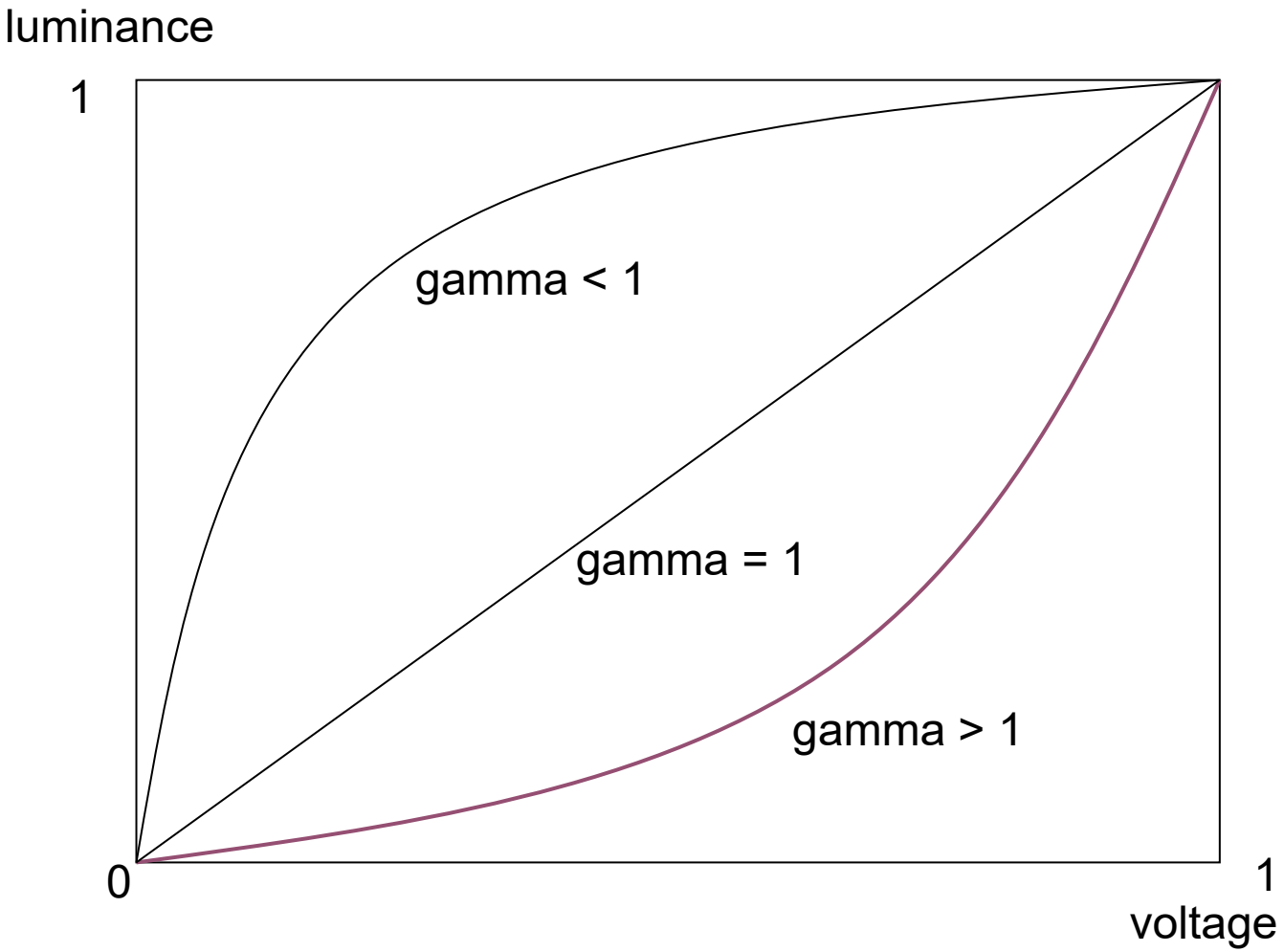


Webster's Law

$$\frac{\Delta I}{I} = k$$



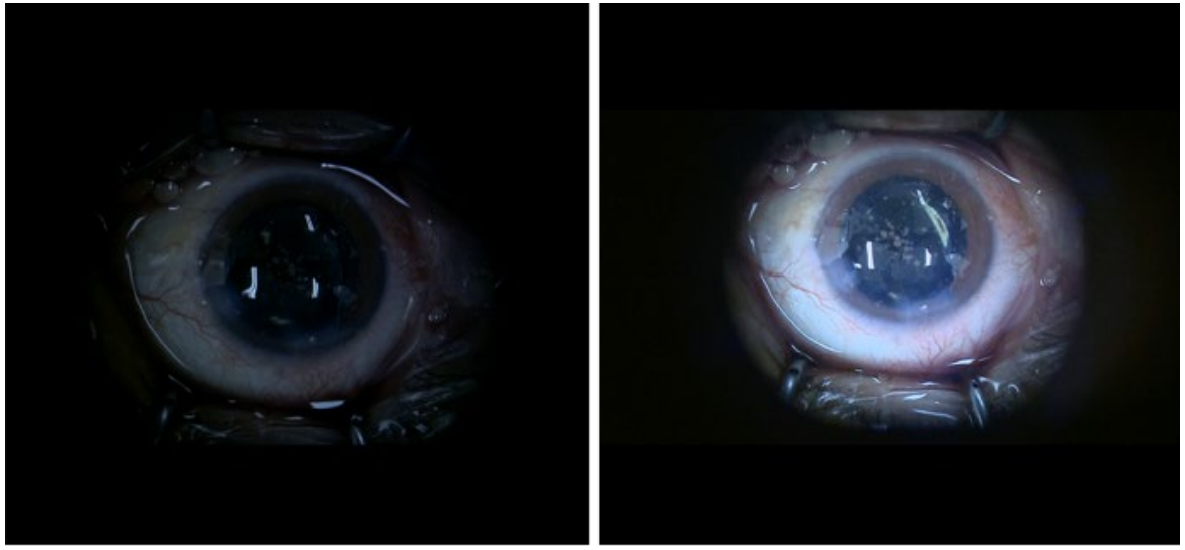
Gamma Correction – Perceptual Uniformity



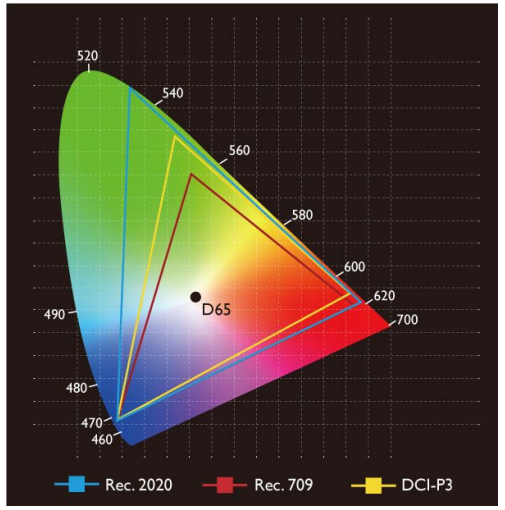
$$R_{corrected} = \left(\frac{R_{uncorrected}}{R_{max}}\right)^{\gamma} \times R_{max}$$

(High) Dynamic Range [HDR]

The ratio between the largest and smallest values that a certain quantity can assume.



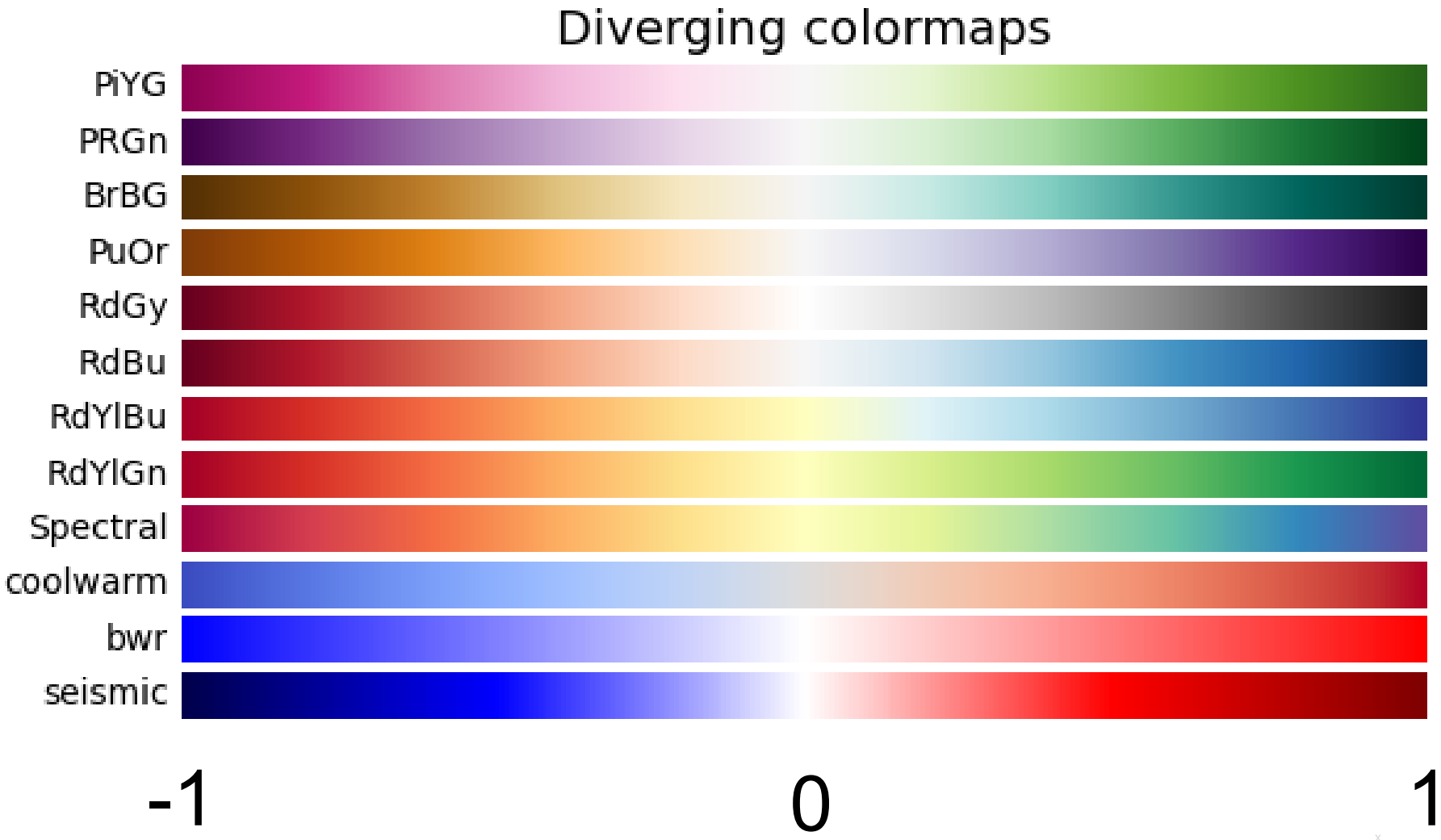
Conventional camera MCC-1000MD



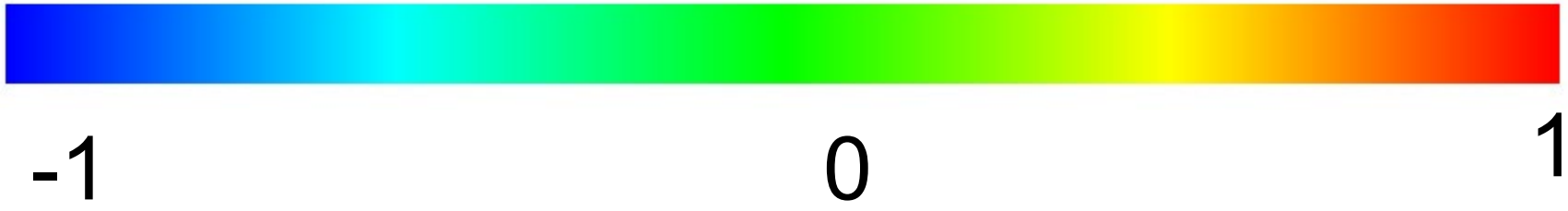
3. Color Map

[assignment-related]

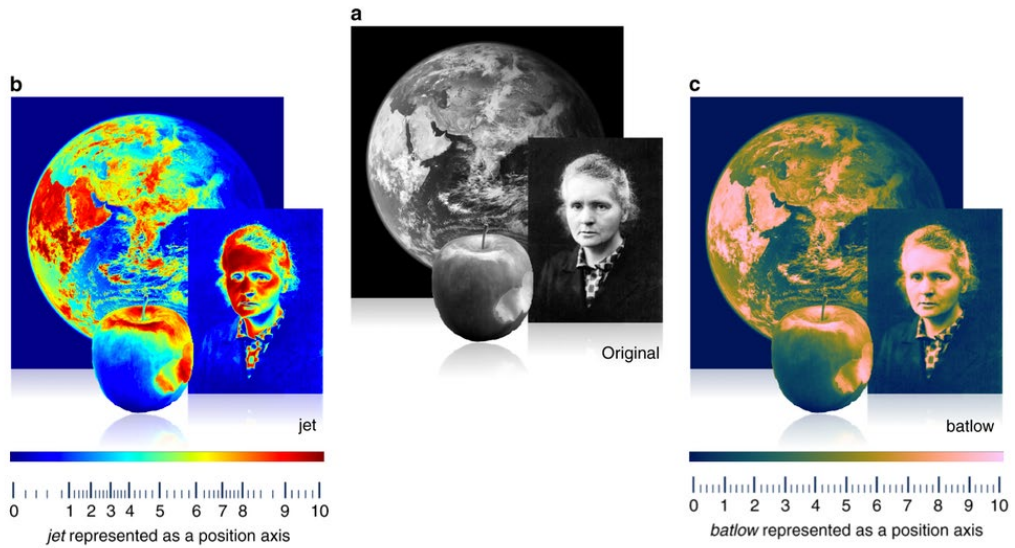
Mapping between Color and Value



(The Poor) Rainbow Color Map

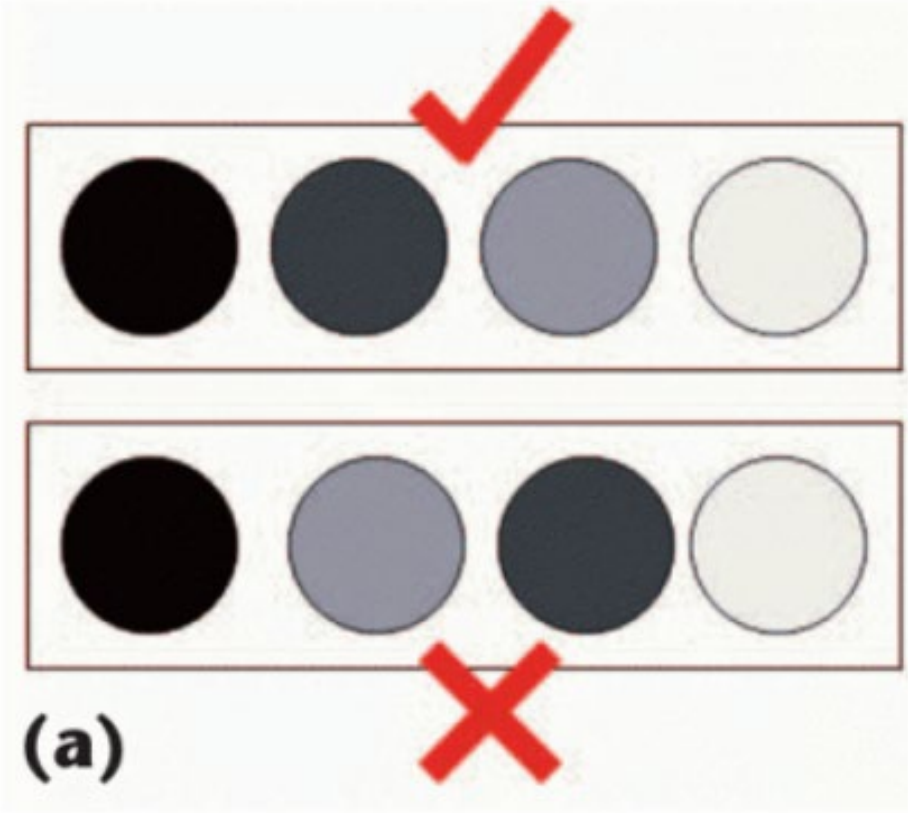


1. No perceptual ordering (confusing)
2. No luminance variation (obscures details)
3. Sharp transitions in color as sharp transitions in data (misleading)



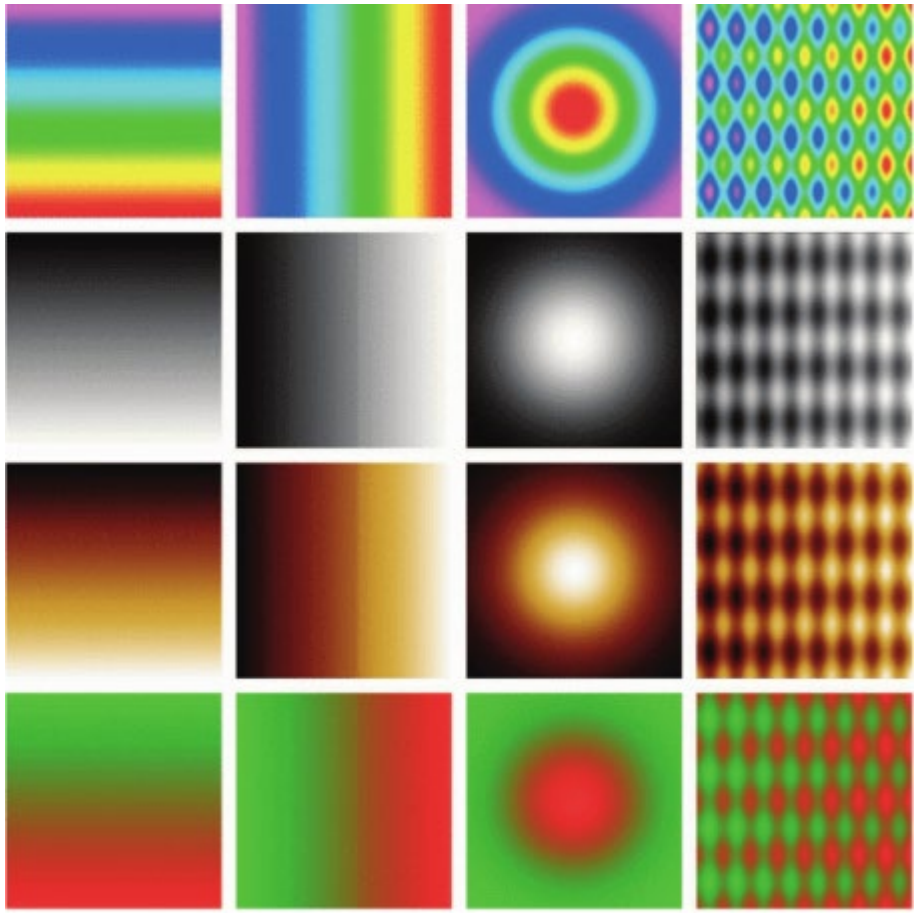
(The Poor) Rainbow Color Map

perceptual ordering



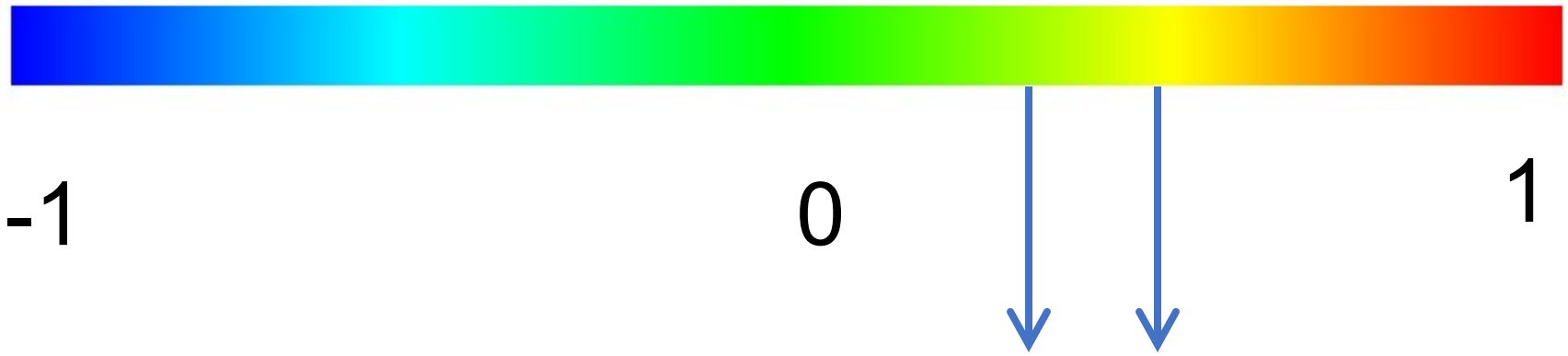
(The Poor) Rainbow Color Map

luminance variation



(The Poor) Rainbow Color Map

Sharp transitions in color as sharp transitions in data



Color Maps

Categorical

Color 

Does not imply magnitude differences

Distinct hues with similar emphasis

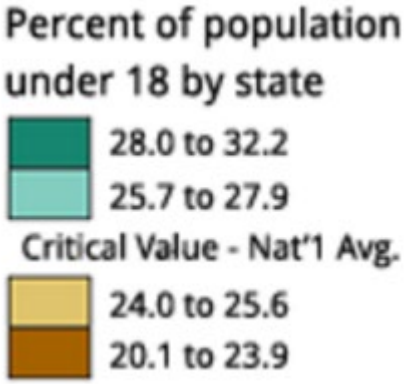
Sequential

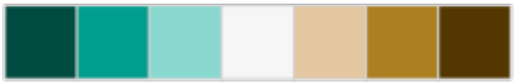
Color 

Best for ordered data that progresses

Luminosity channel effectively employed

Diverging

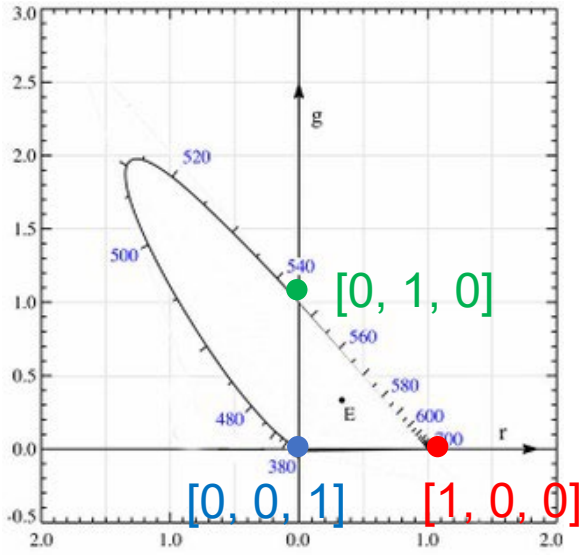


Color 

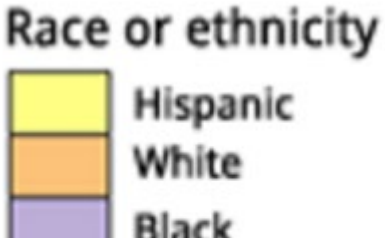
For data with a “diverging” (mid) point

Equal emphasis on mid-range critical values and extremes at both ends

Discussion: Color Maps in rg-Chromaticity



Categorical

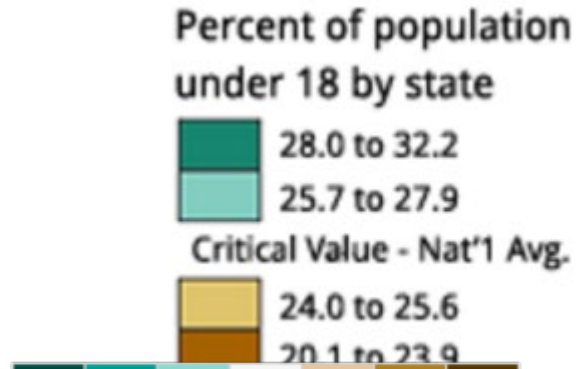


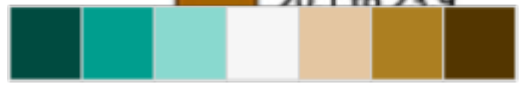
Color 

Sequential

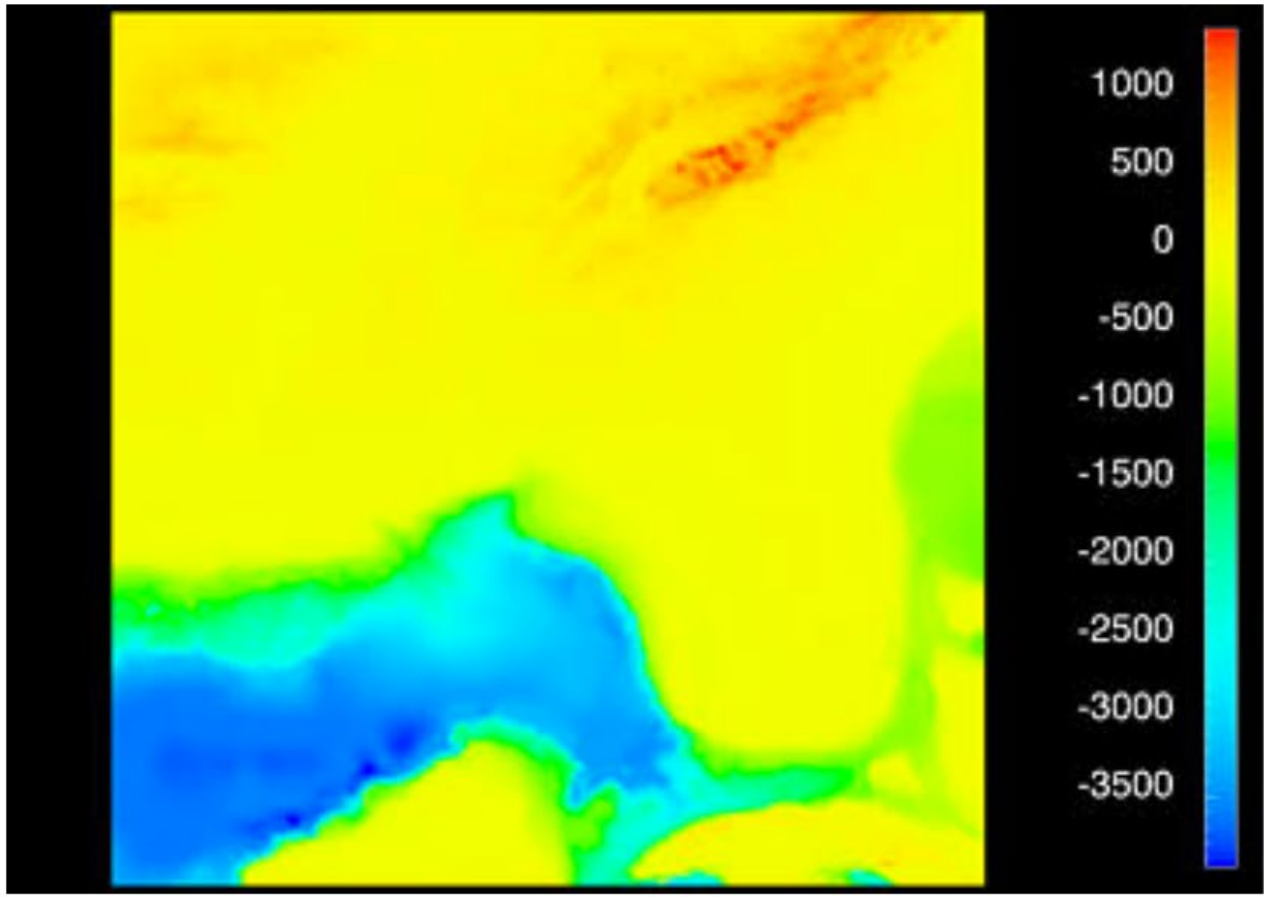
Color 

Diverging

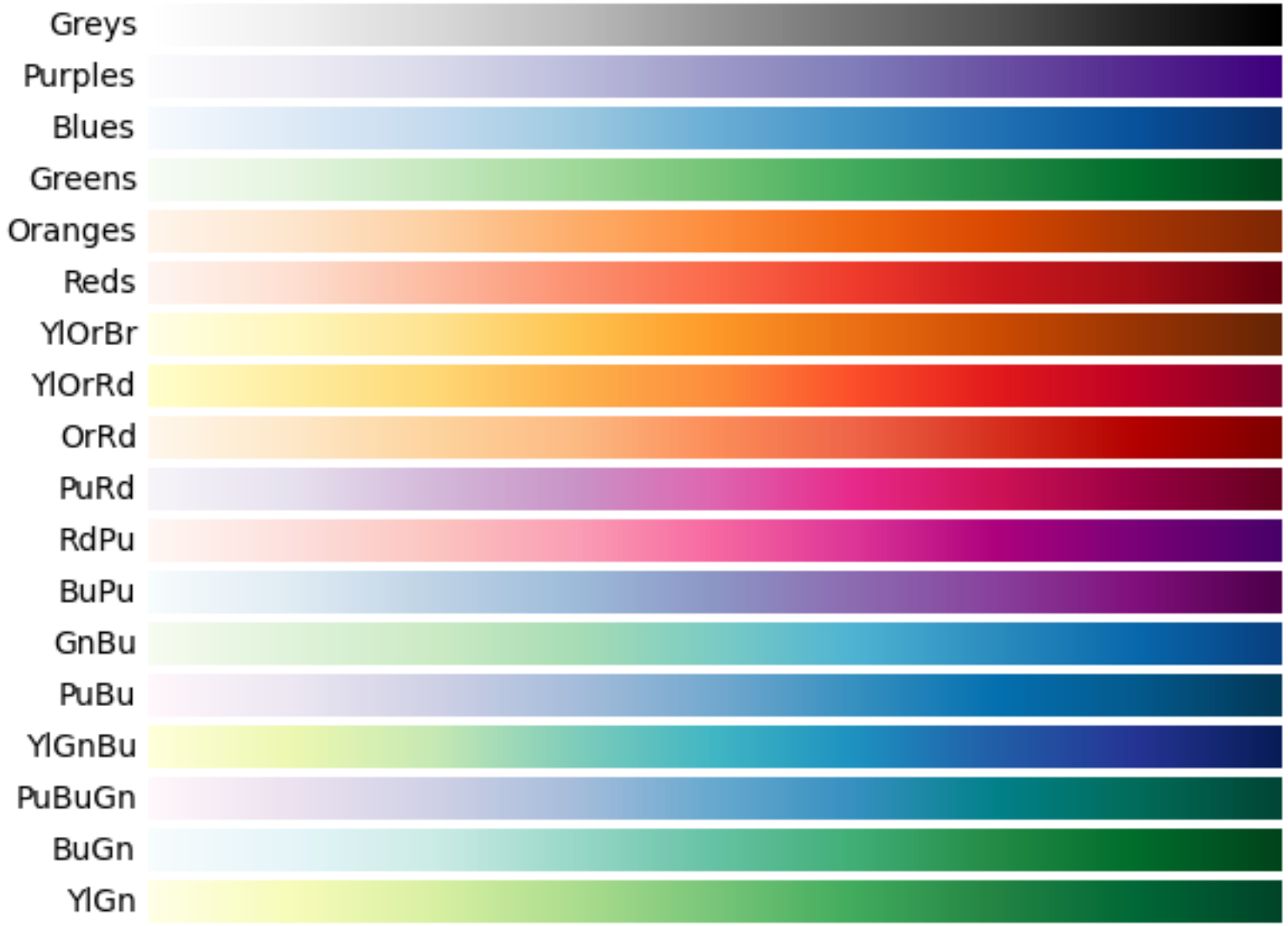


Color 

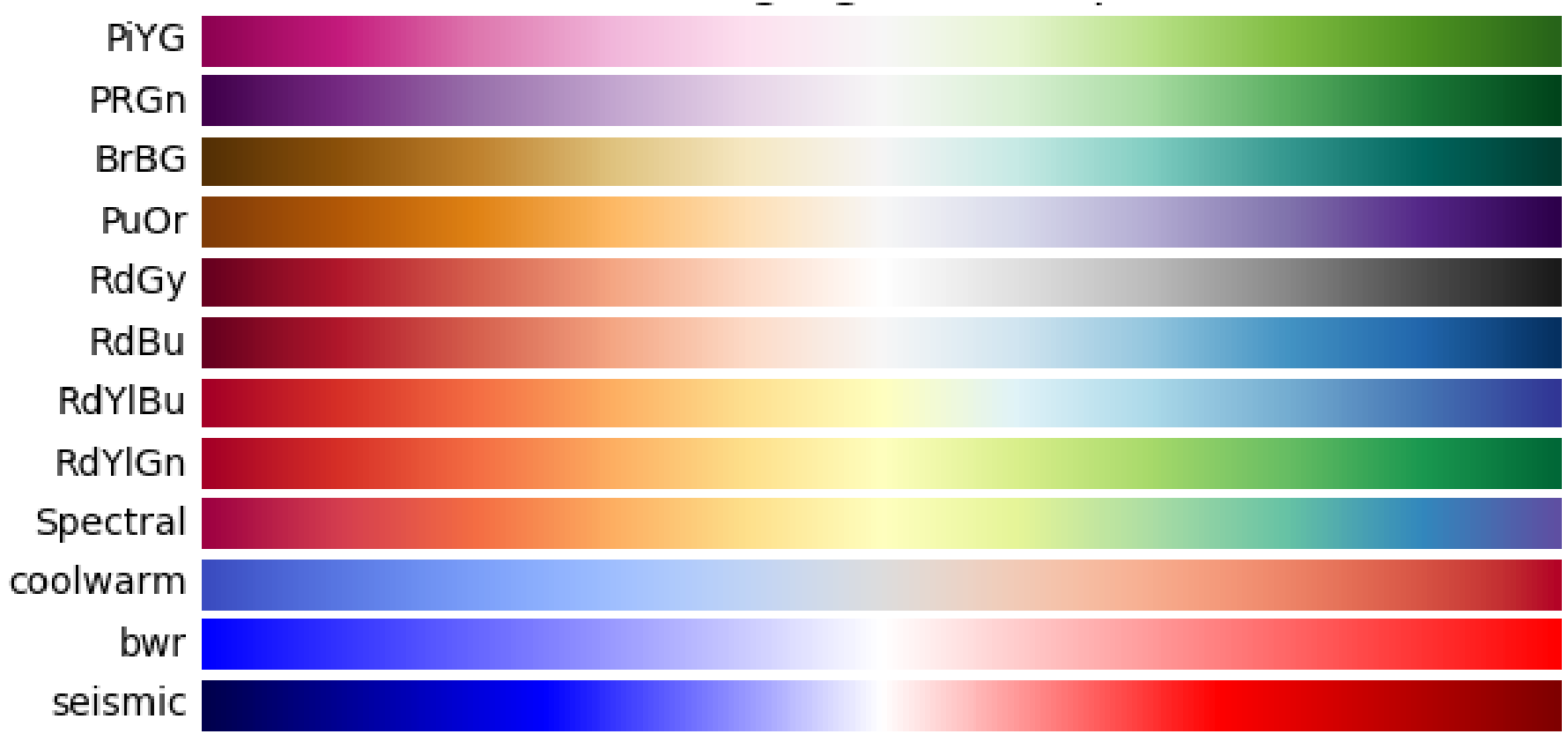
Discussion:



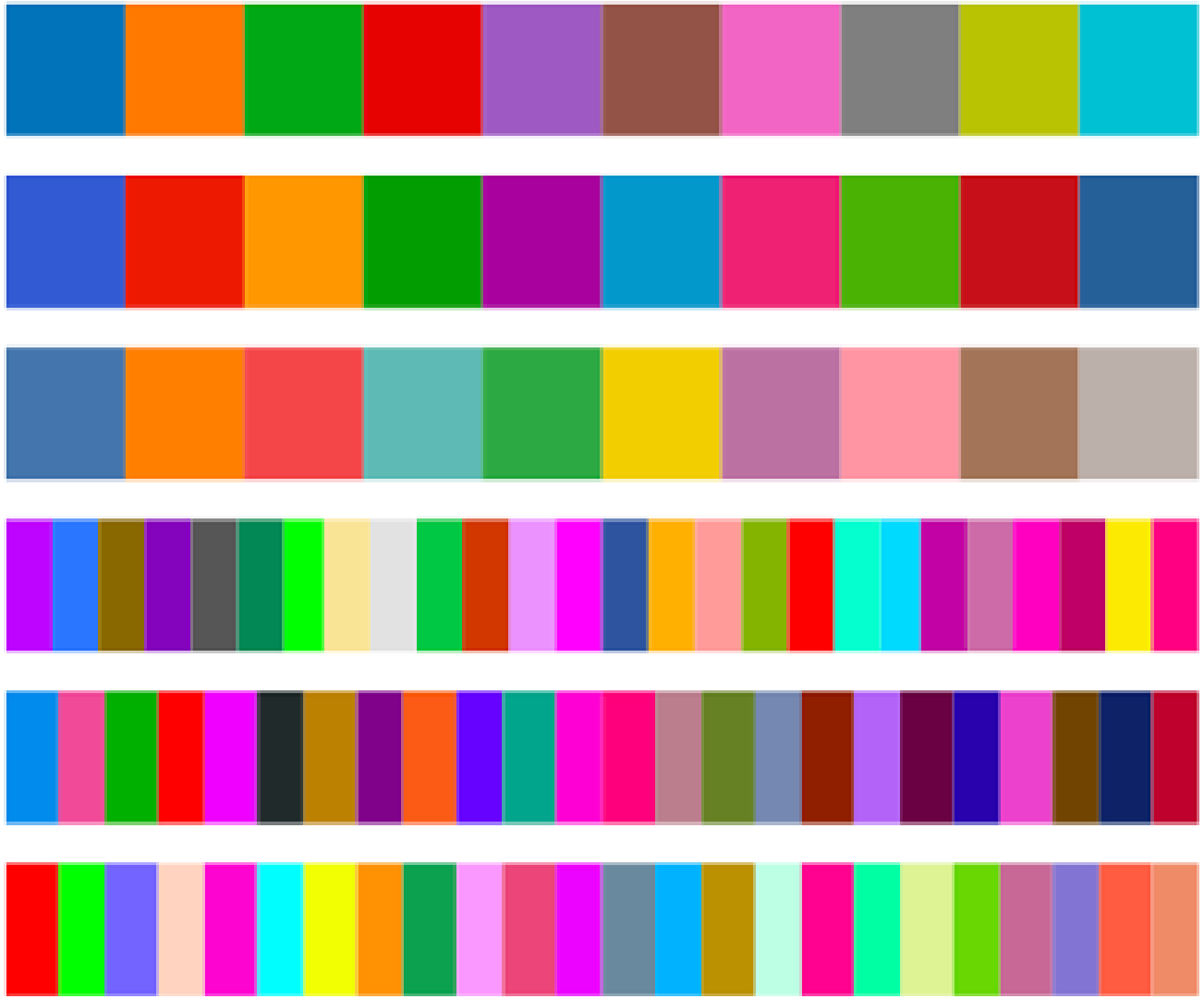
Exercise:


Greys
Purples
Blues
Greens
Oranges
Reds
YlOrBr
YlOrRd
OrRd
PuRd
RdPu
BuPu
GnBu
PuBu
YlGnBu
PuBuGn
BuGn
YlGn

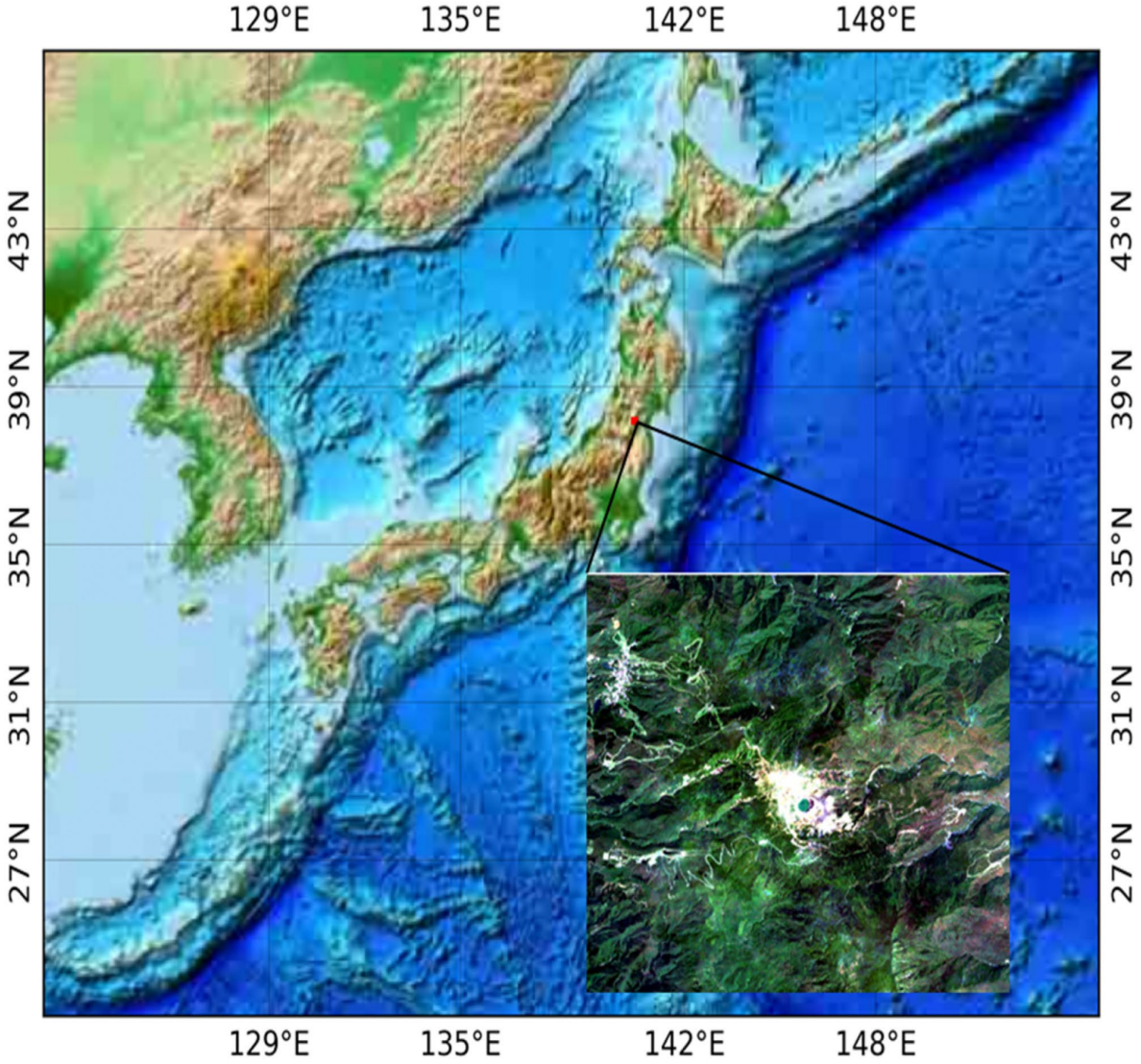
Exercise:


PiYG
PRGn
BrBG
PuOr
RdGy
RdBu
RdYlBu
RdYlGn
Spectral
coolwarm
bwr
seismic

Exercise:



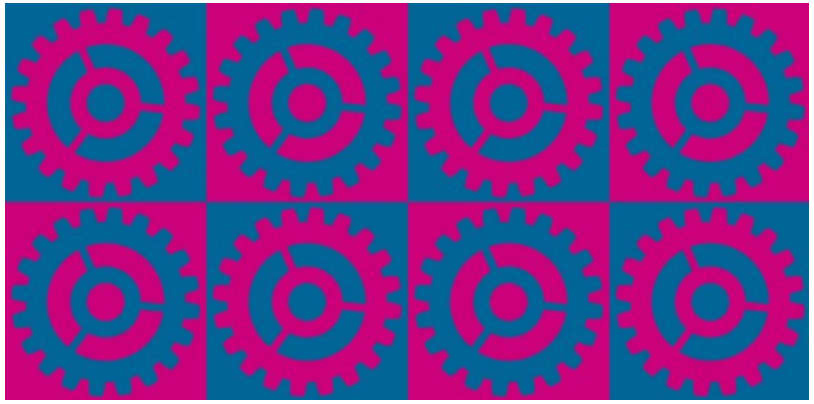
Discussion: Piece-Wise Sequential



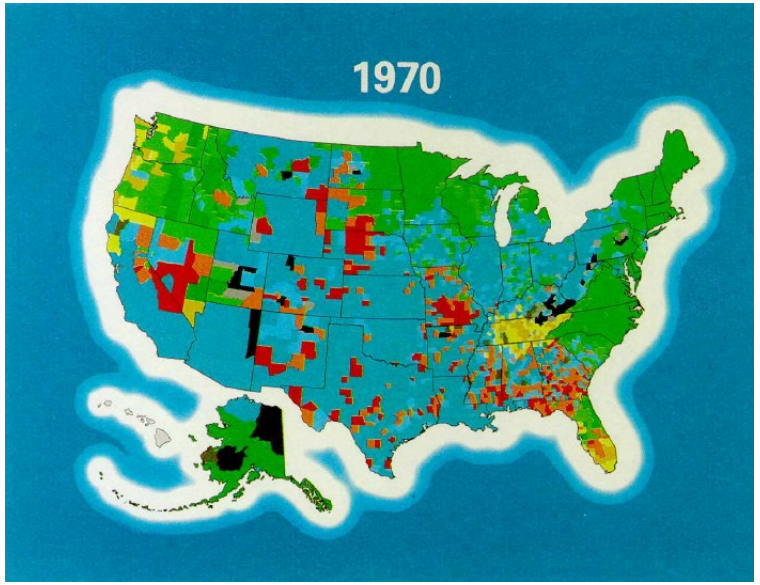
4. Color Design Rules/Laws

Color Rules

First rule: Pure, bright or very strong colors have loud, unbearable effects when they stand unrelieved over large areas adjacent to each other, but extraordinary effects can be achieved when they are used sparingly on or between dull background tones.

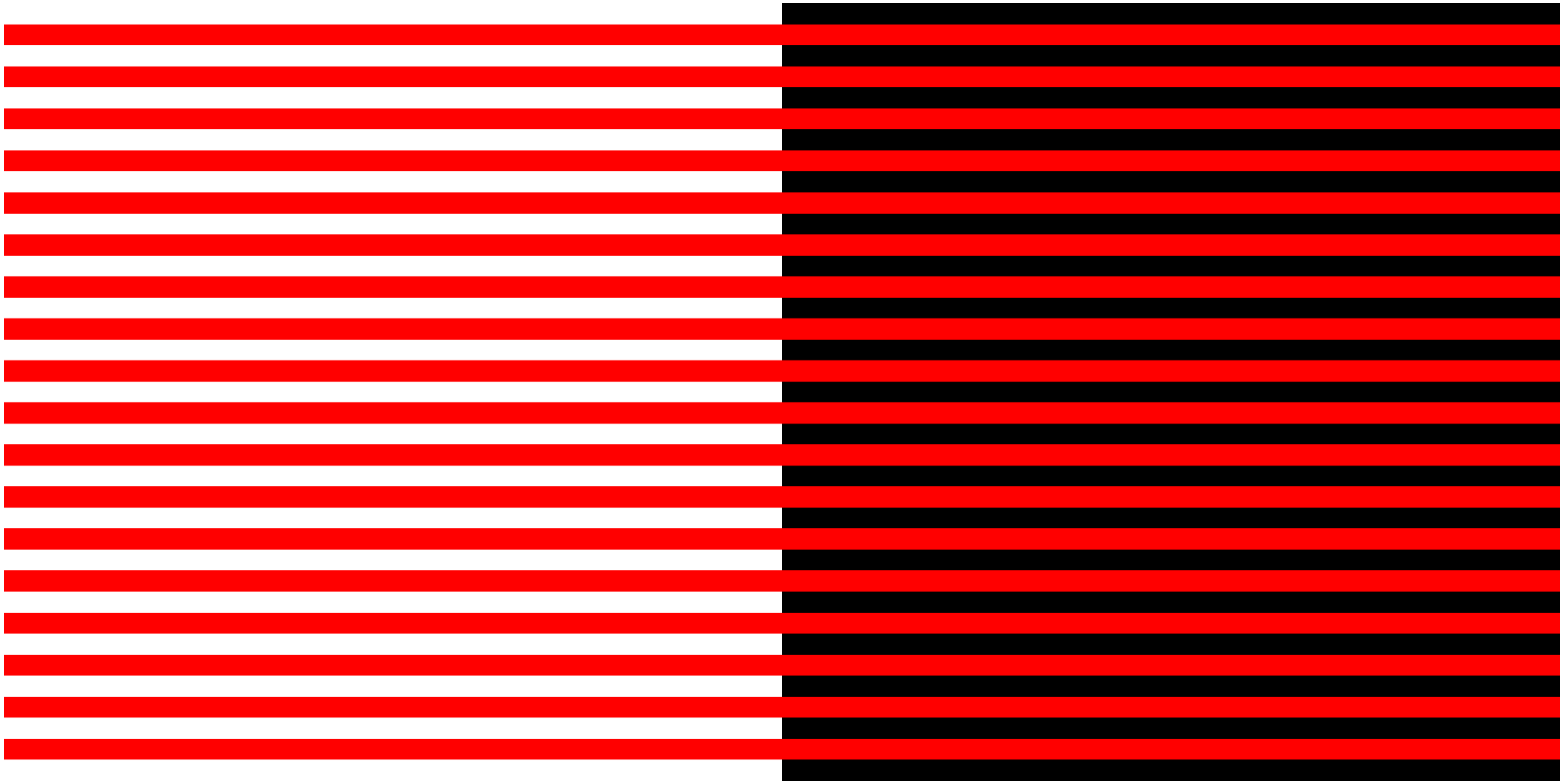


Second rule: The placing of light, bright colors mixed with white next to each other usually produces unpleasant results, especially if the colors are used for large areas



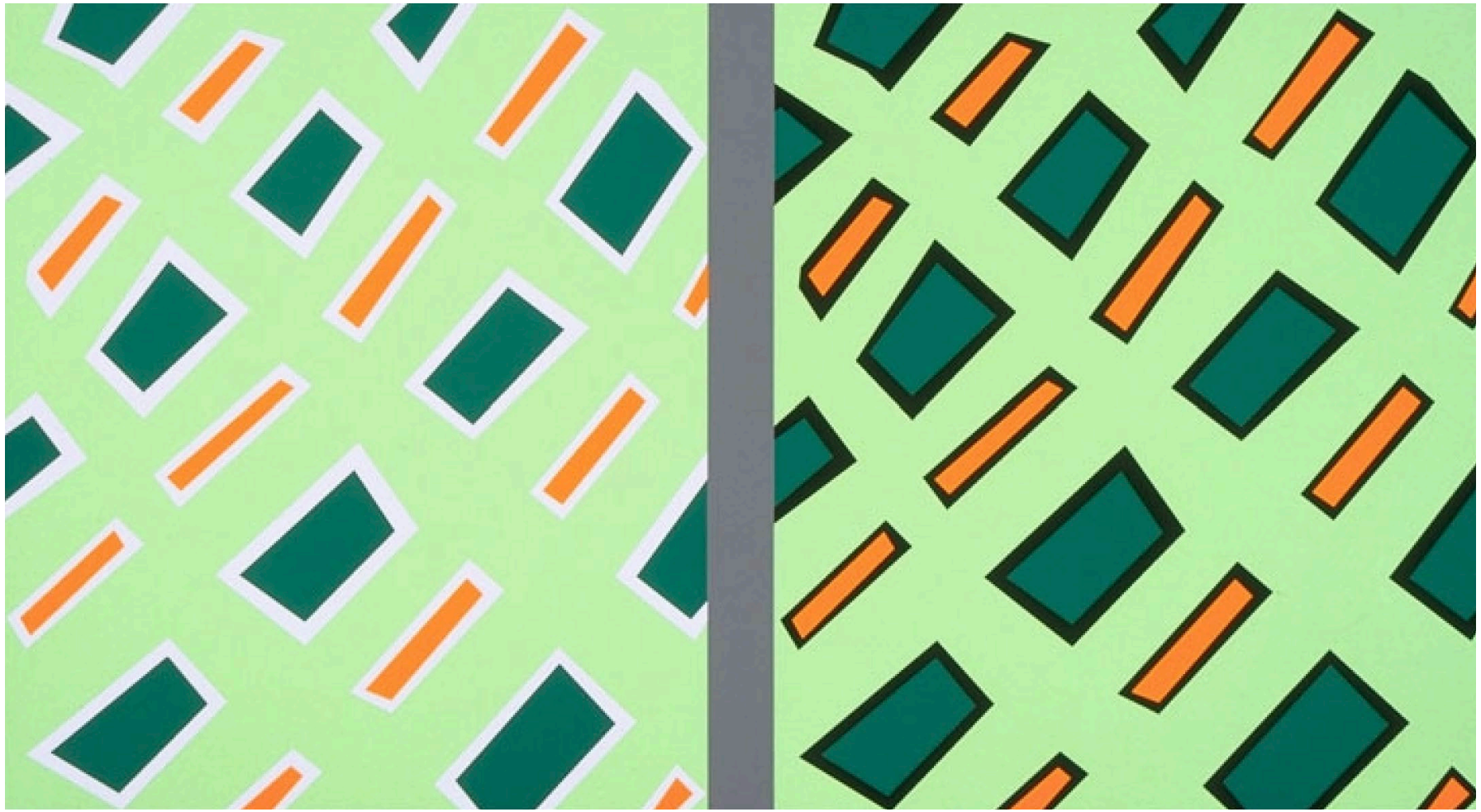
Bezold Effect

a color may appear different depending on its relation to adjacent colors.



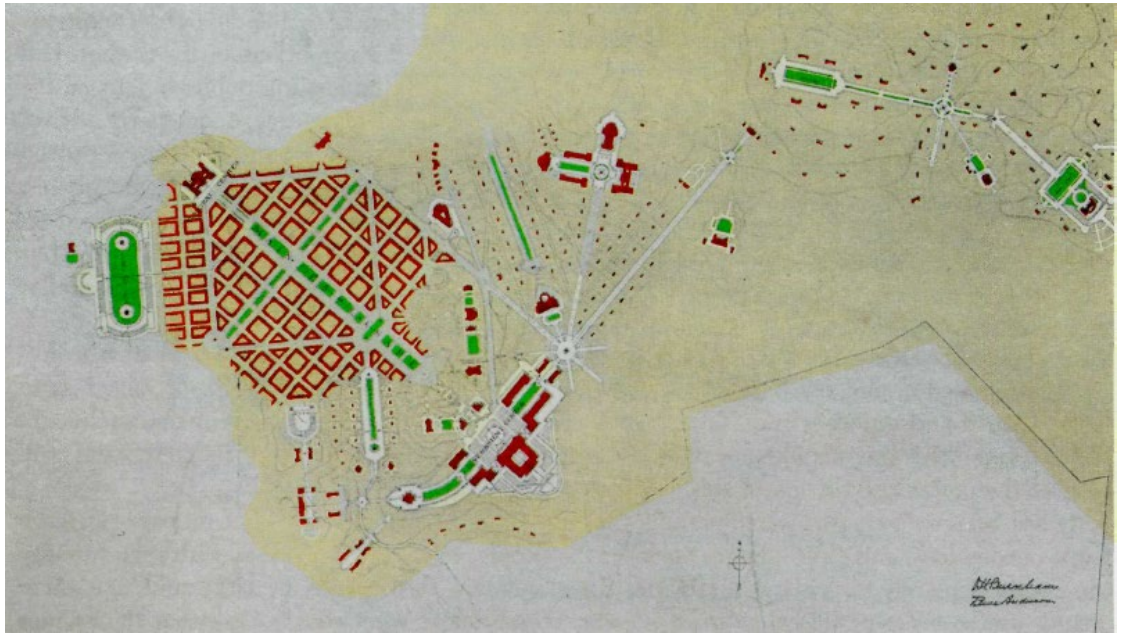
Bezold Effect

a color may appear different depending on its relation to adjacent colors.

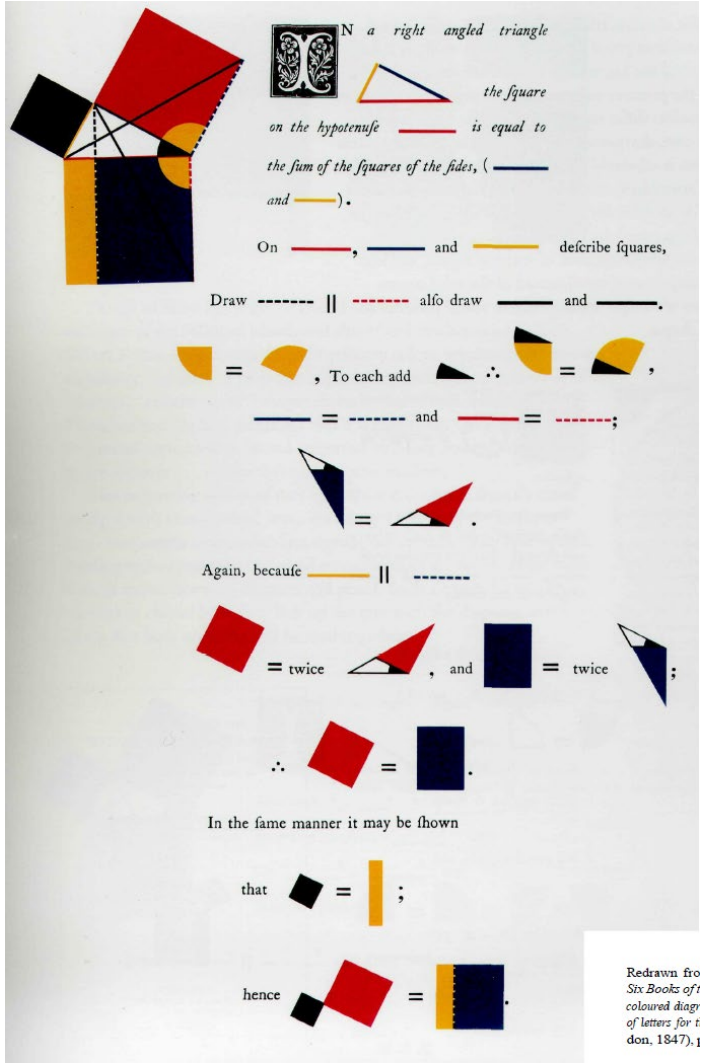


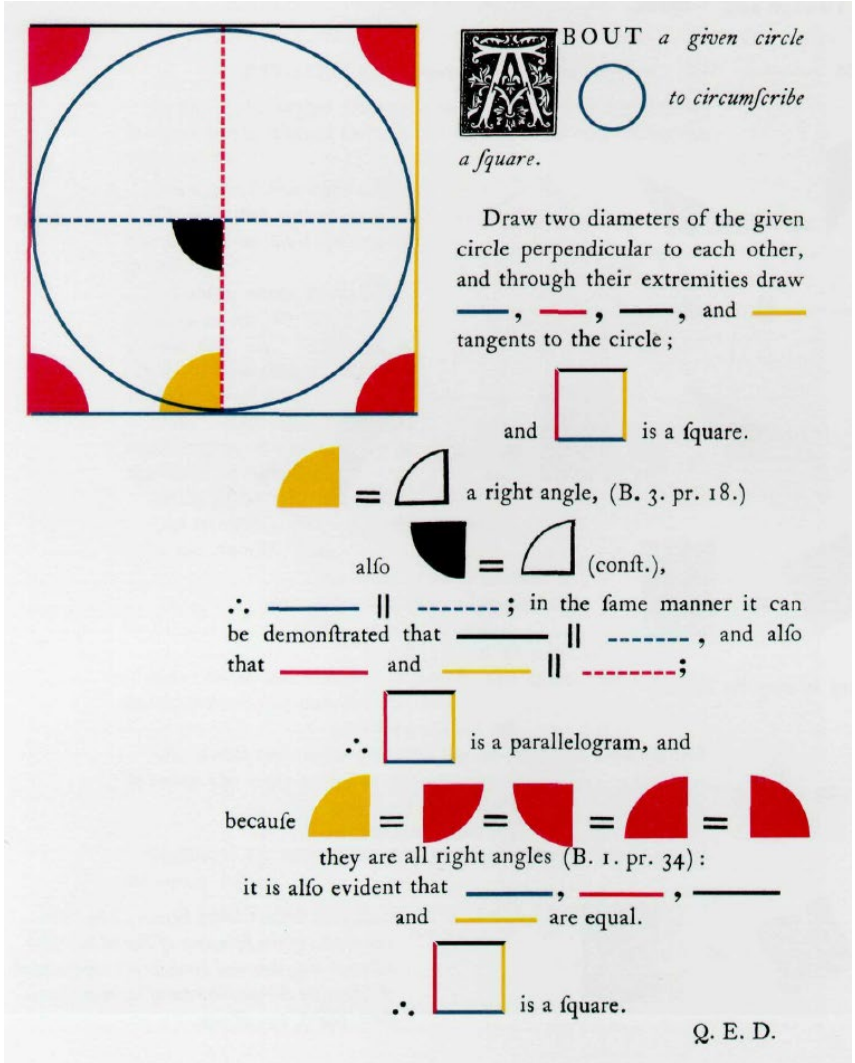
Color and Data Information

Color spots against a light gray or muted field highlight and italicize data, and also help to weave an overall harmony.

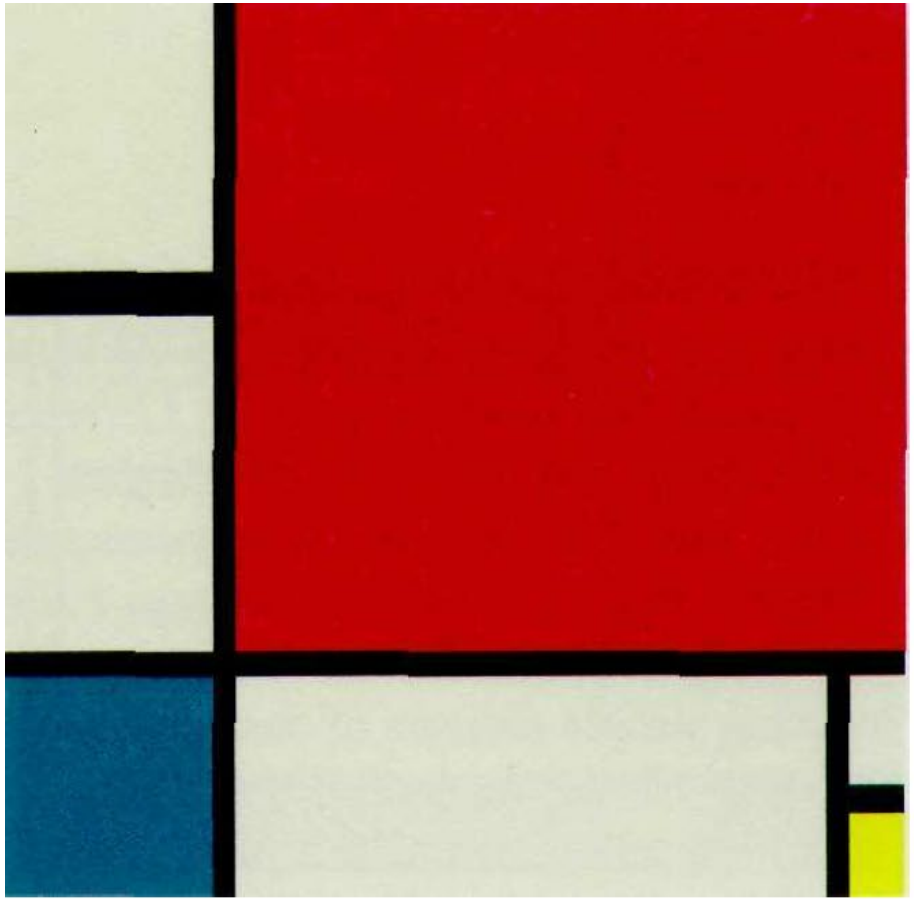


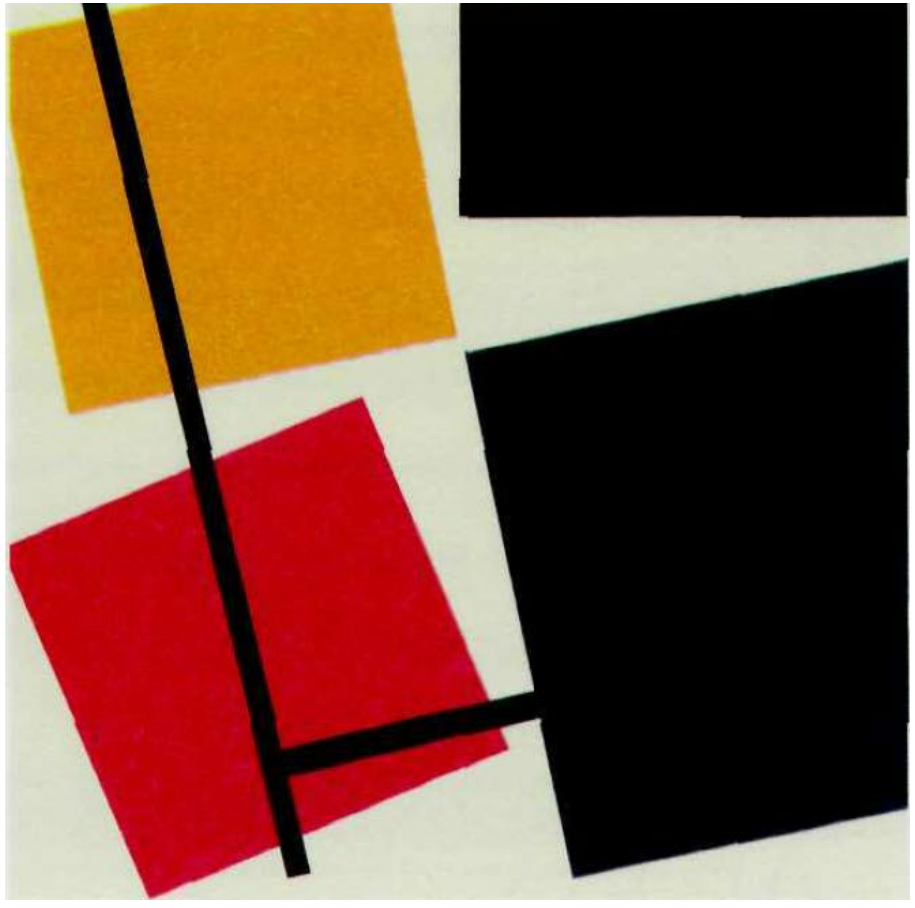
Color and Data Information



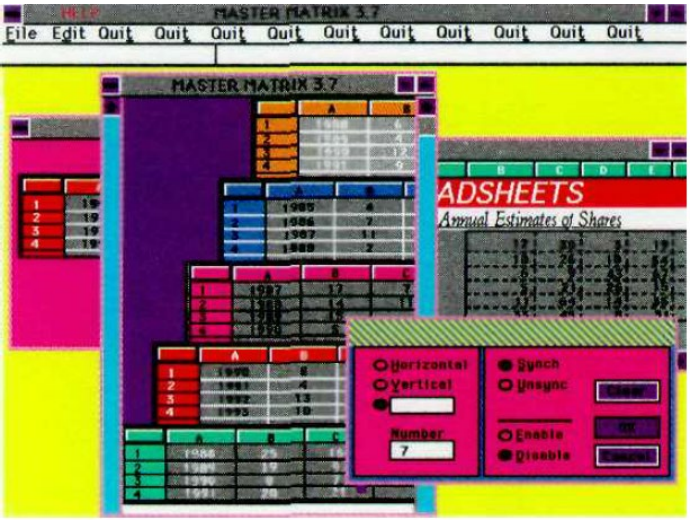


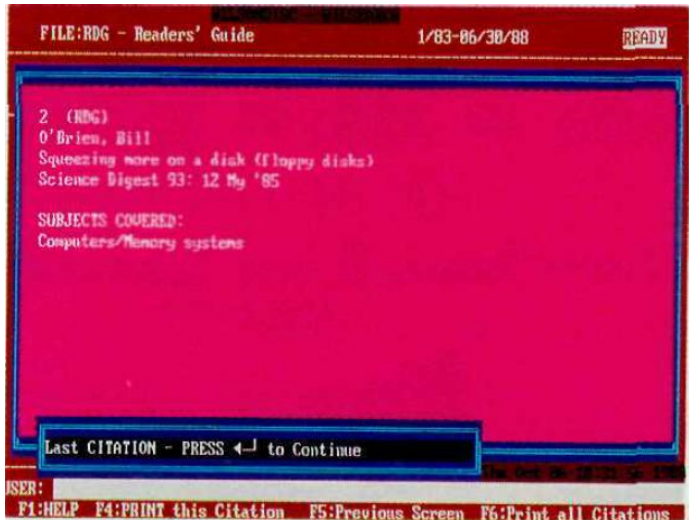
Color and Data Information

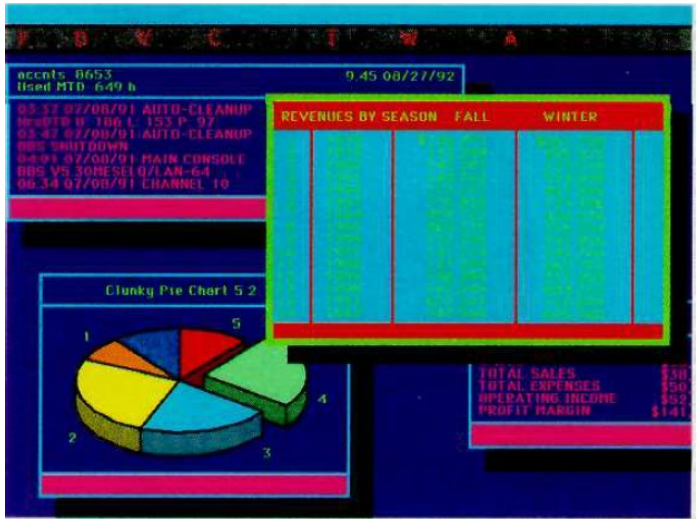




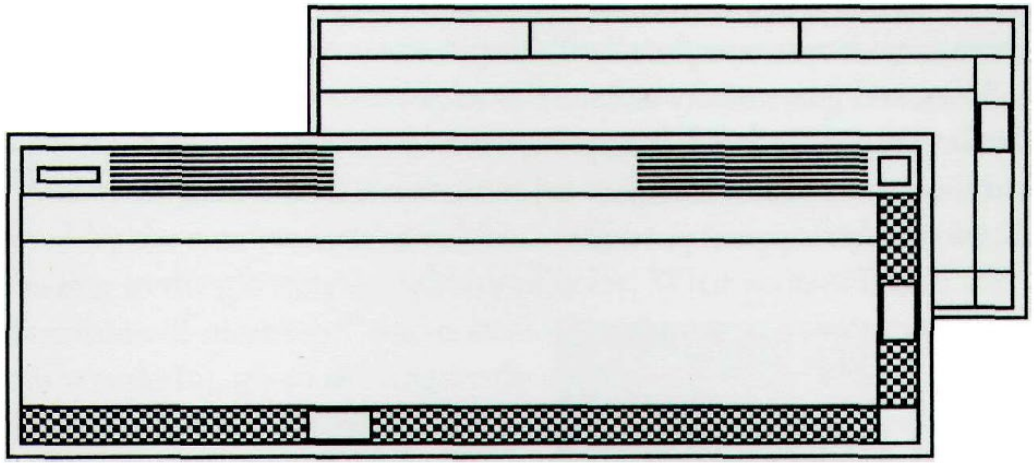
Bad Color Usage

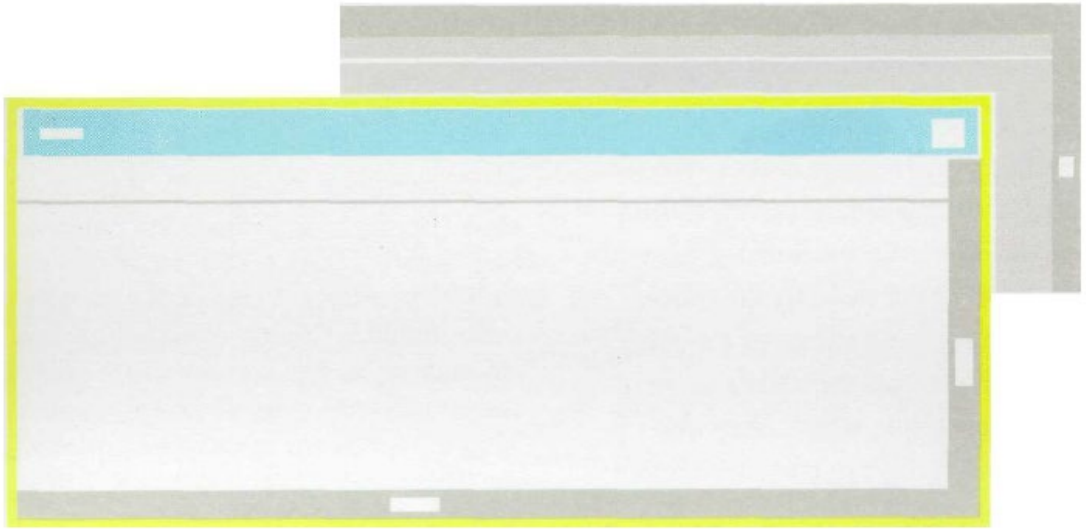




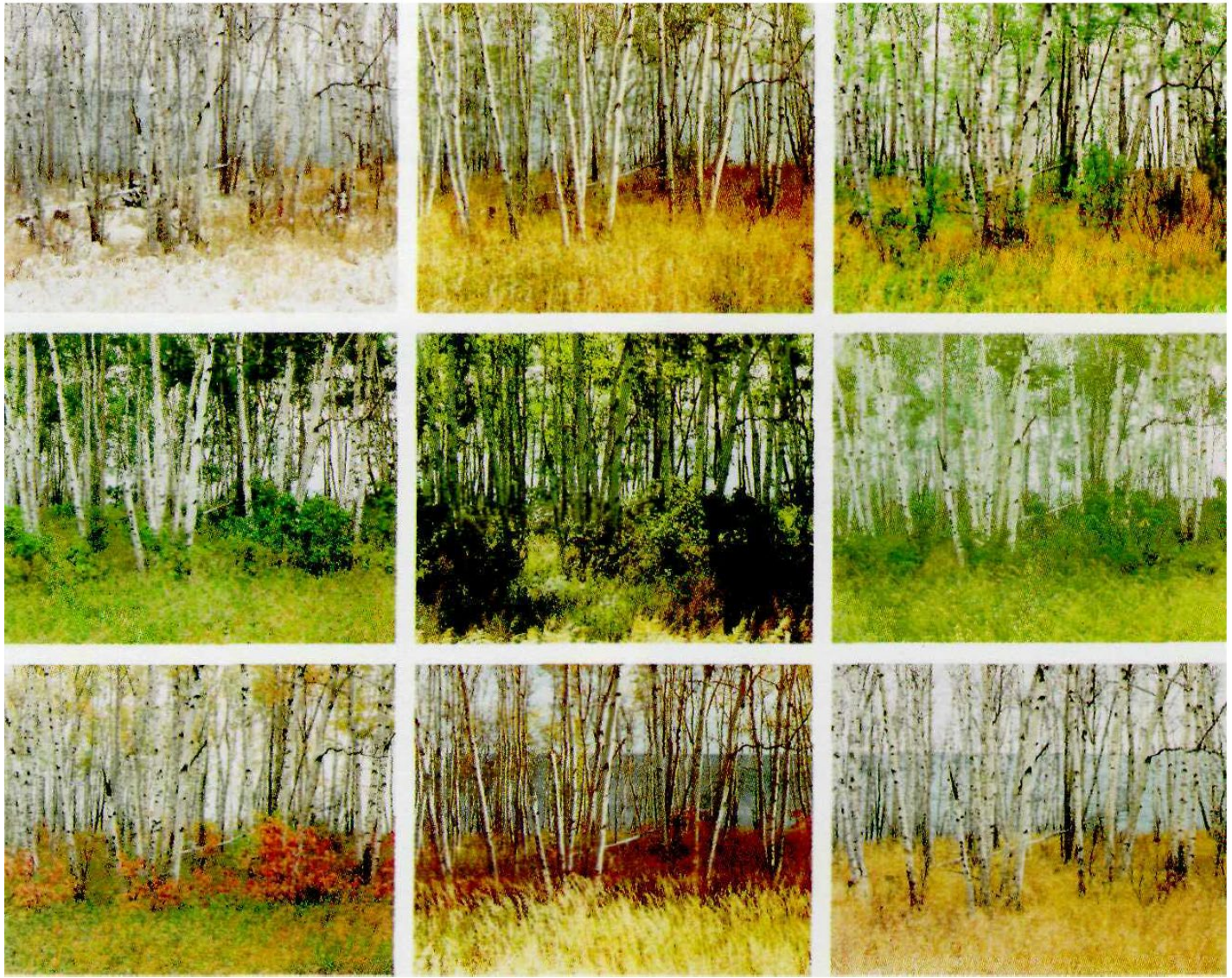


Bad Color Usage



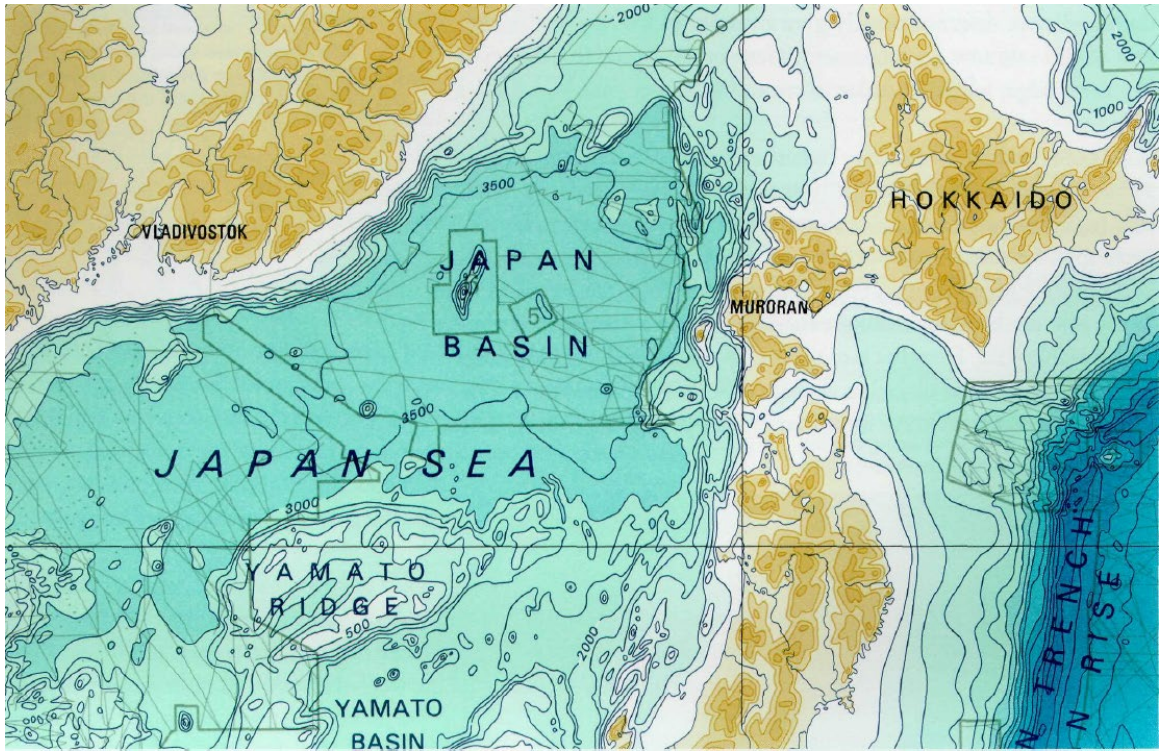


Natural Color Palette

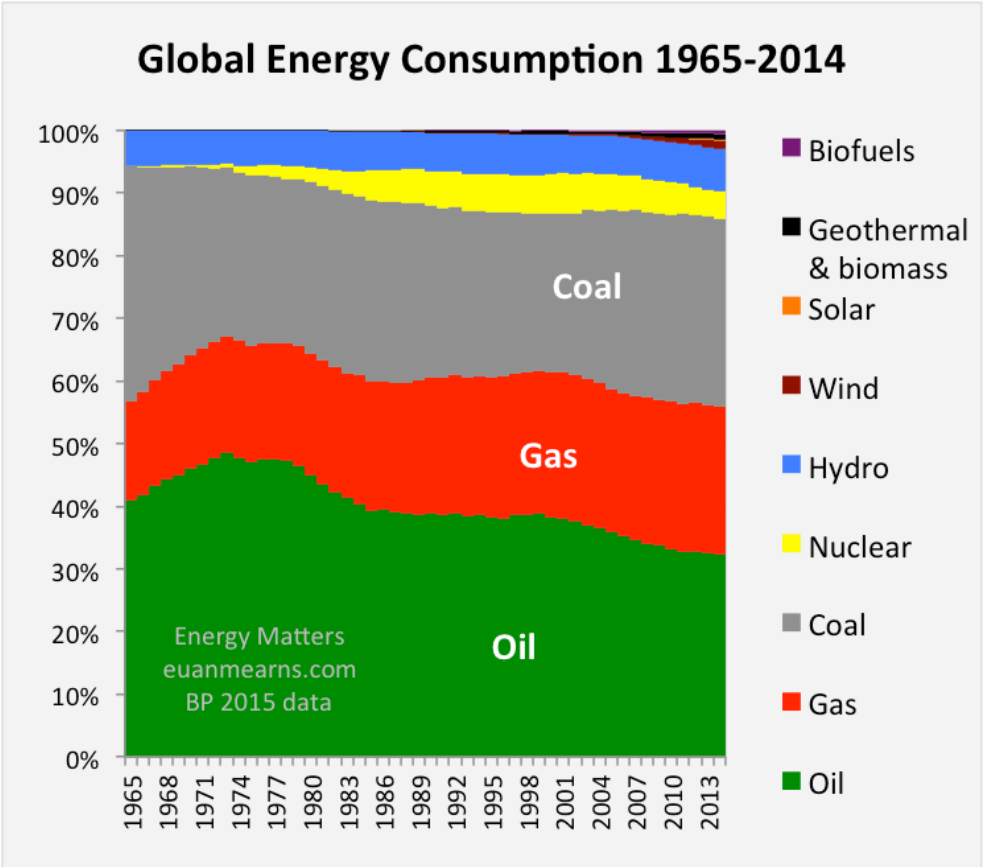


Color Rules

Third rule: Large area background or base-colors should do their work most quietly,
allowing the smaller, bright areas to stand out most vividly, if the former are muted,
grayish or neutral.



Fourth rule: If a picture is composed of two or more large, enclosed areas in different colors, then the picture falls apart.

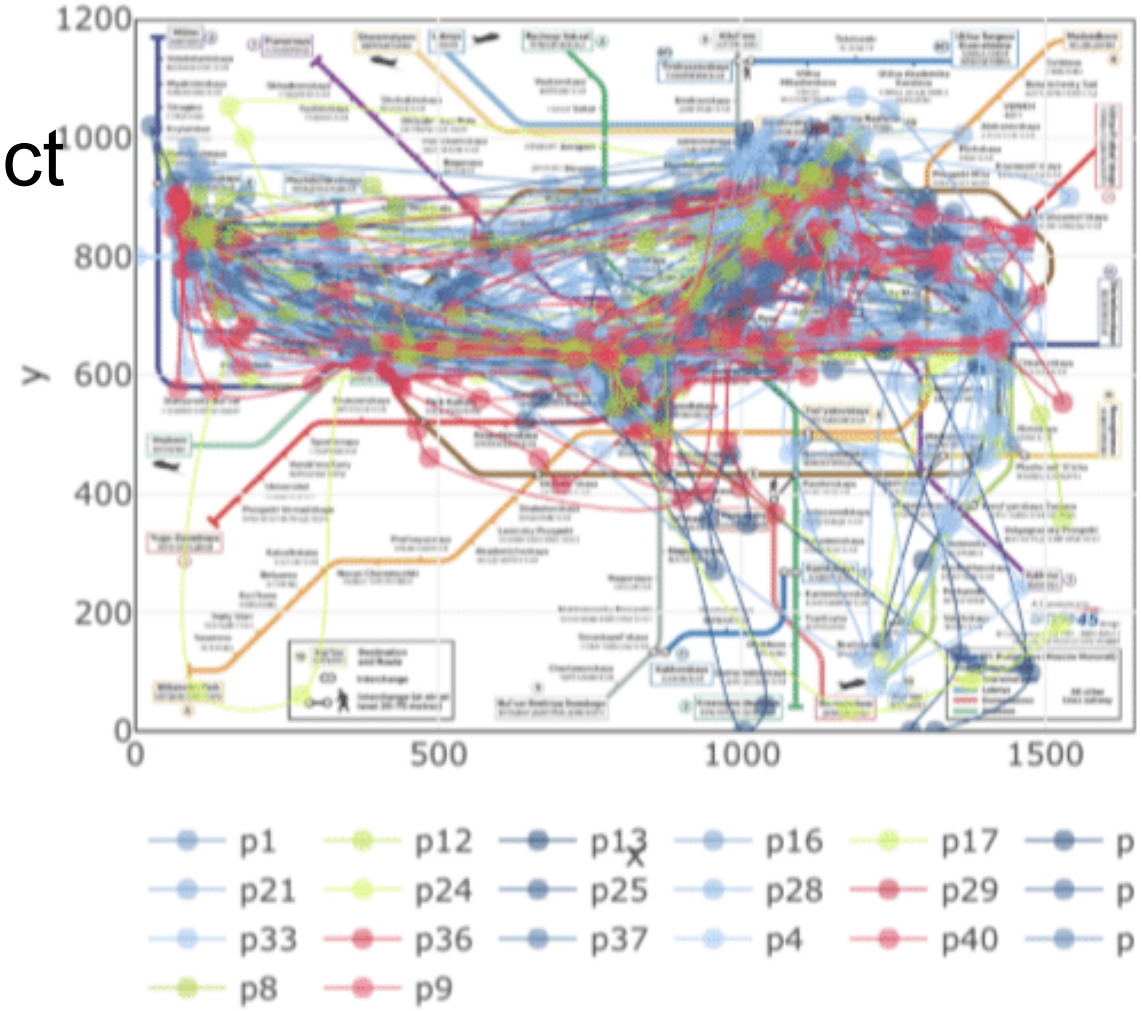


Making Color Effective

- Avoid clutter
- Legibility
- Accommodate color-deficient

Avoid Color Clutter

- Too many colors spoil the effect
- Examples
  - Maps/charts
  - Text logos



Legibility

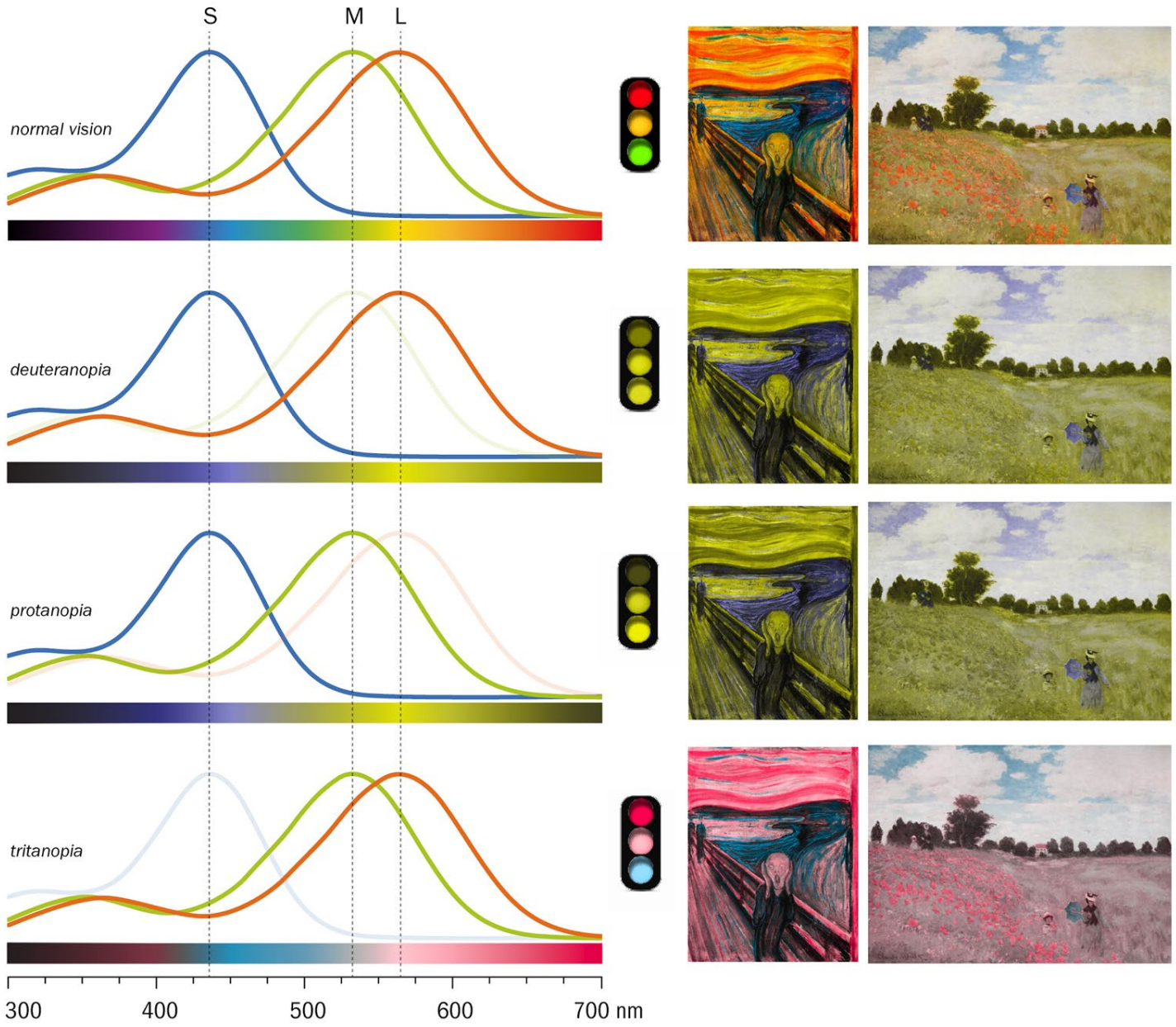
- Luminance Contrast
  - Hue alone doesn’t make an edge
  - Make legible in grayscale
  - Drop shadows and similar tricks

ddddddddddd

Color Deficient

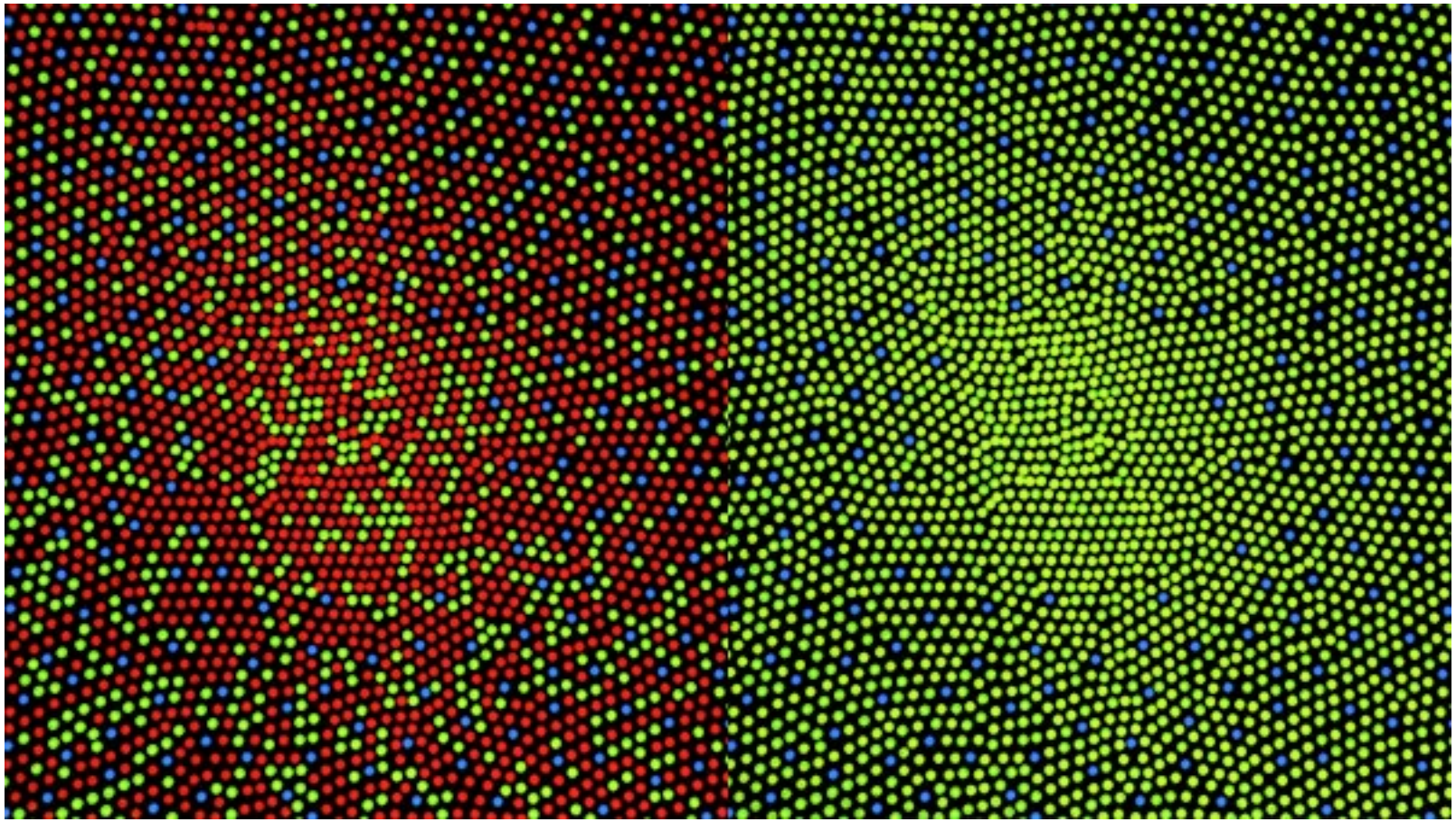
eye response to color in color blindness

RELATIVE ABSORPTION OF COLOR PHOTORECEPTORS
AND APPEARANCE OF SPECTRUM AND OBJECTS



http://mkweb.bcgsc.ca/colorblind

Missing or defective photoreceptors



10% male and 1% female have some color deficiencies

Lack of M and L receptors most common.

Can’t distinguish red from green.

← BACK TO BLOG

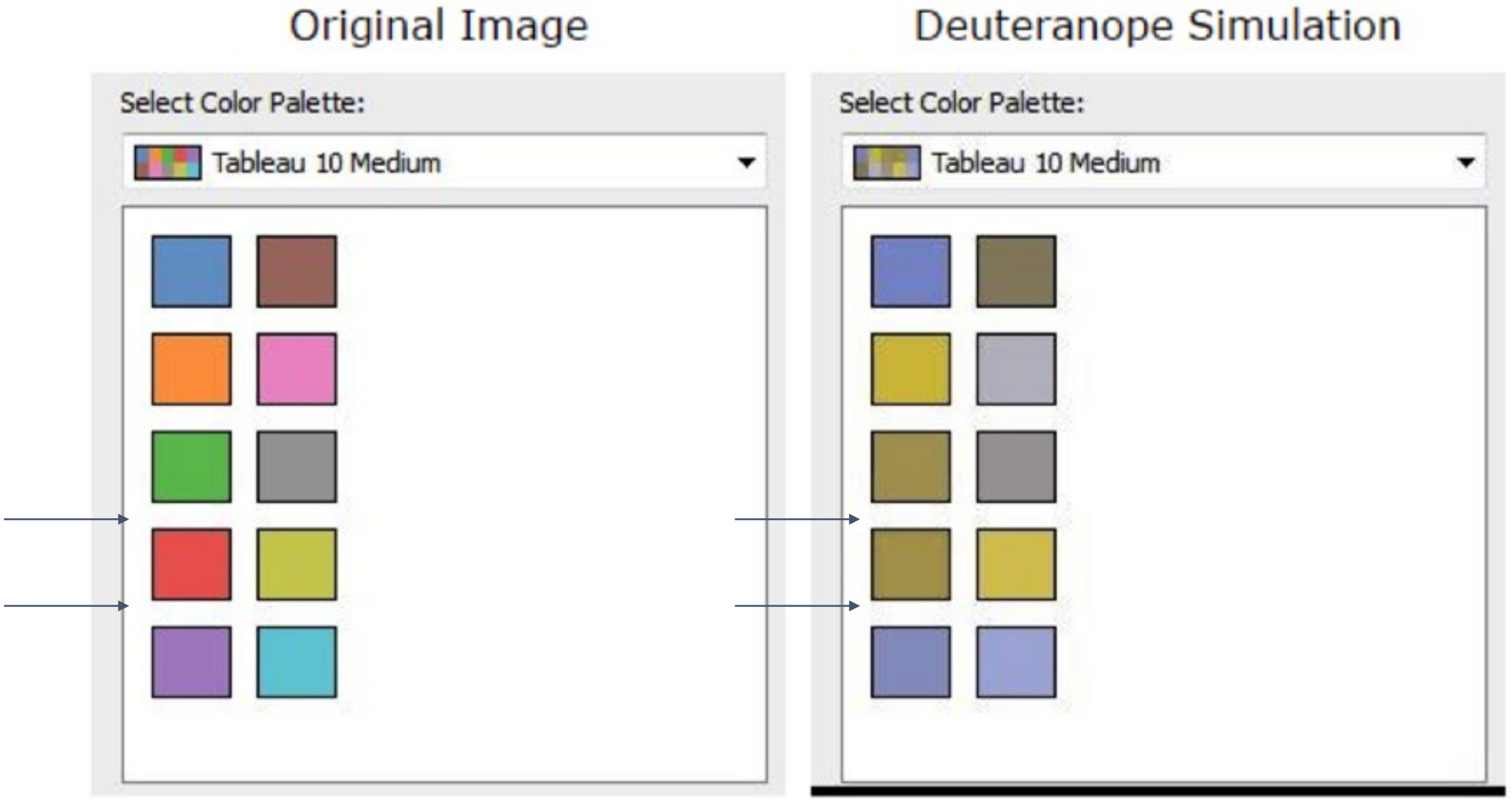
5 tips on designing colorblind-friendly visualizations

GUEST AUTHOR
APRIL 20, 2016

Note: The following is the first installment in a series of guest posts by Tableau Zen Master Jeffrey Shaffer.

Original Image

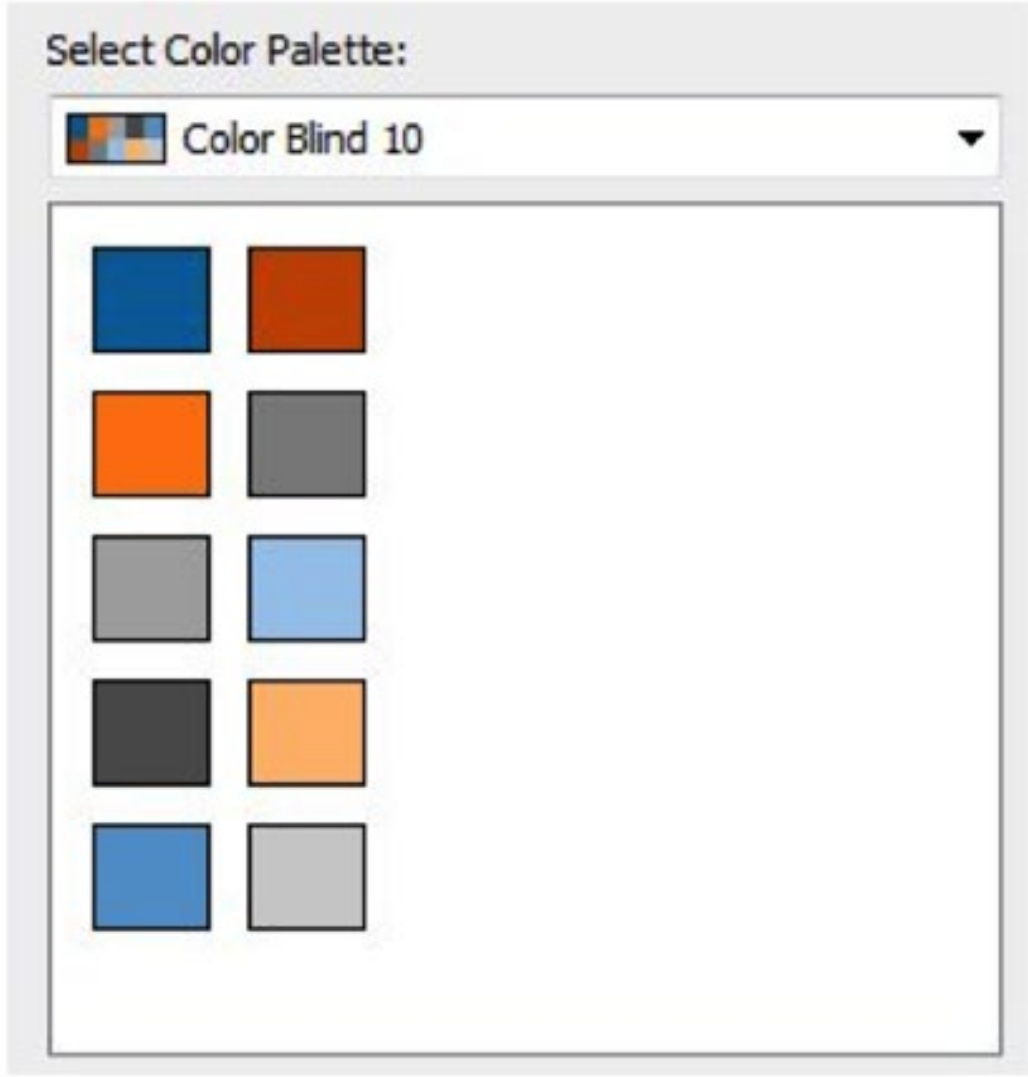
Protanope Simulation



red, green, and brown … blue and purple … pink and gray … and gray and brown (<= Animate this sequence!)

How do you design for colorblind people?

Original Image



Deuteranope Simulation

The world.

How the world looks to a person with a red/green color deficit (deuteranopia).

How the world looks to a person with a blue/yellow color deficit (tritanopia).

Some colorful hats.

As seen by a person with deuteranopia.

As seen by a person with protanopia, another form of red/green deficit.

Use colorblind safe palettes

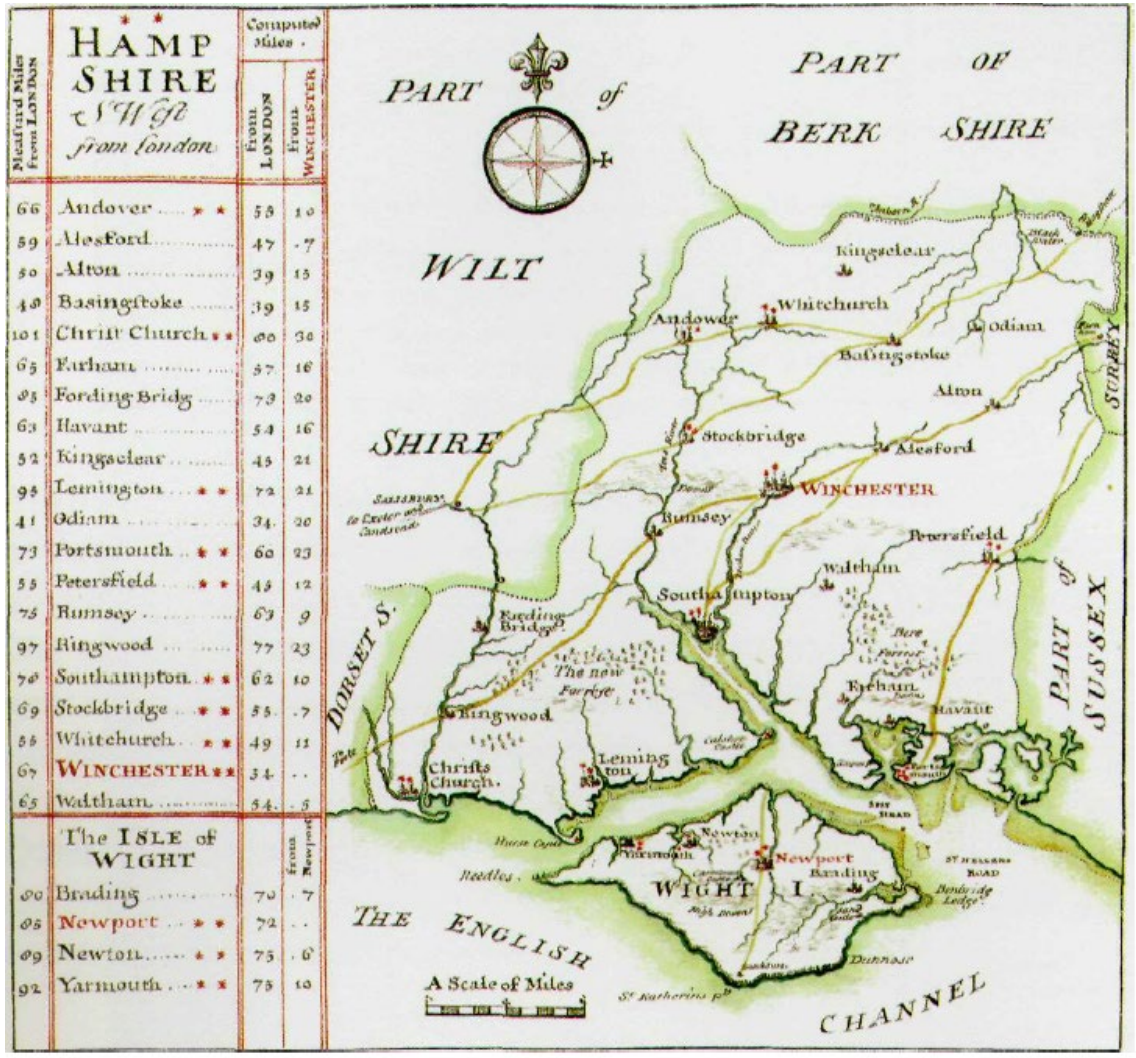
Blue/orange and blue/red normally safe

Consider variations in color intensity

Test design with color blindness simulators

Discussion: Bad Usage of Color

Good Usage of Color



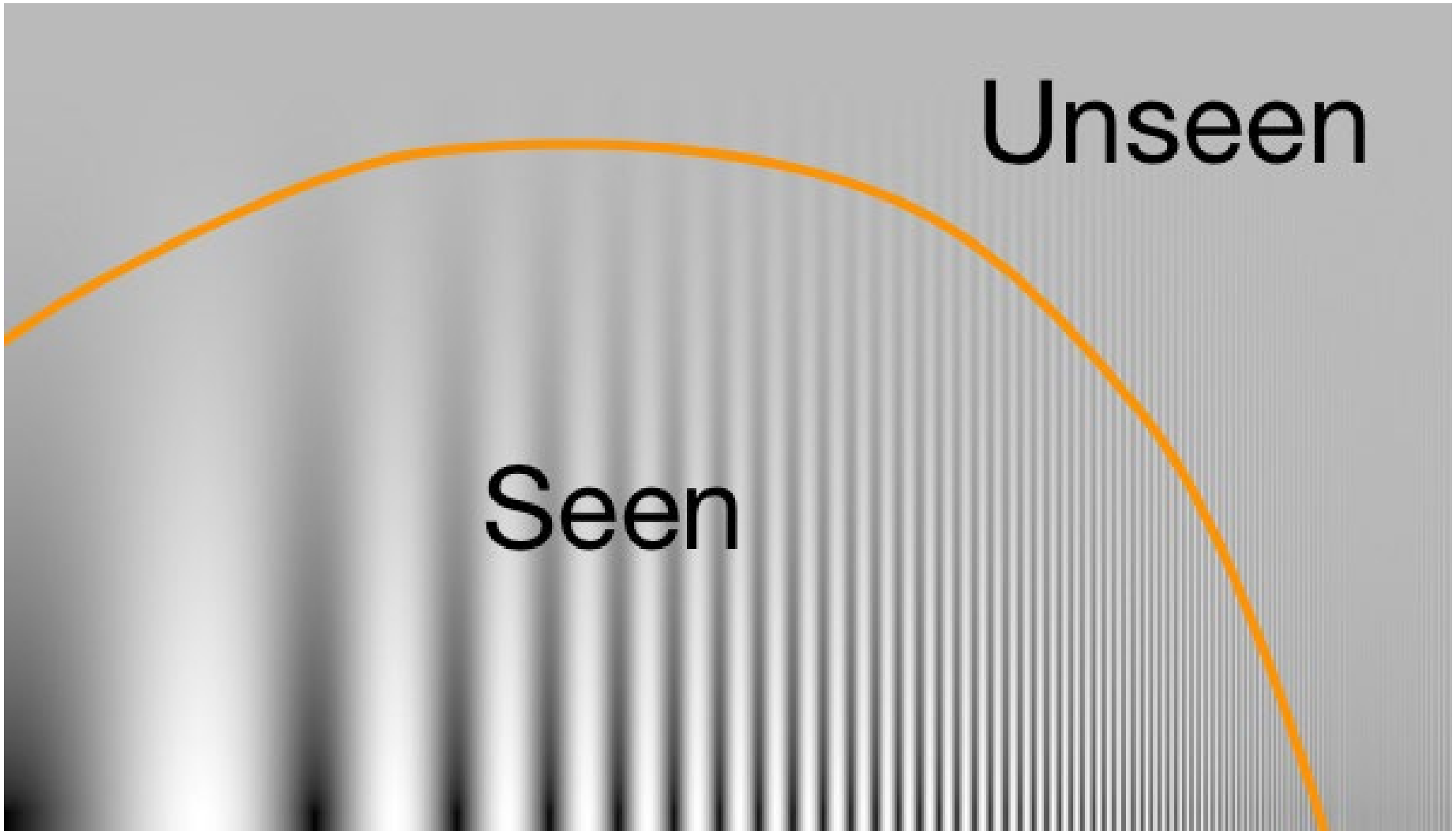
5. Characteristics beyond Color

Contrast

Contrast

- **Weber contrast** $\frac{I - I_\mathrm{b}}{I_\mathrm{b}}$
- **Michelson contrast** $\frac{I_\mathrm{max} - I_\mathrm{min}}{I_\mathrm{max} + I_\mathrm{min}}$
- **RMS contrast** $\sqrt{\frac{1}{MN}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left(I_{ij} - \bar{I}\right)^2}$

Contrast Sensitivity



Why is contrast important in visualization?

Fail

1.67

Contrast

Contrast is the difference in luminance or color that makes an object (or its representation in an image or display) distinguishable. In visual perception of the real world, contrast is determined by the difference in the color and brightness of the object and other objects within the same field of view. Because the human visual system is more sensitive to contrast than absolute luminance, we can perceive the world similarly regardless of the huge changes in illumination over the day or from place to place. The maximum contrast of an image is the contrast ratio or dynamic range.

AAA

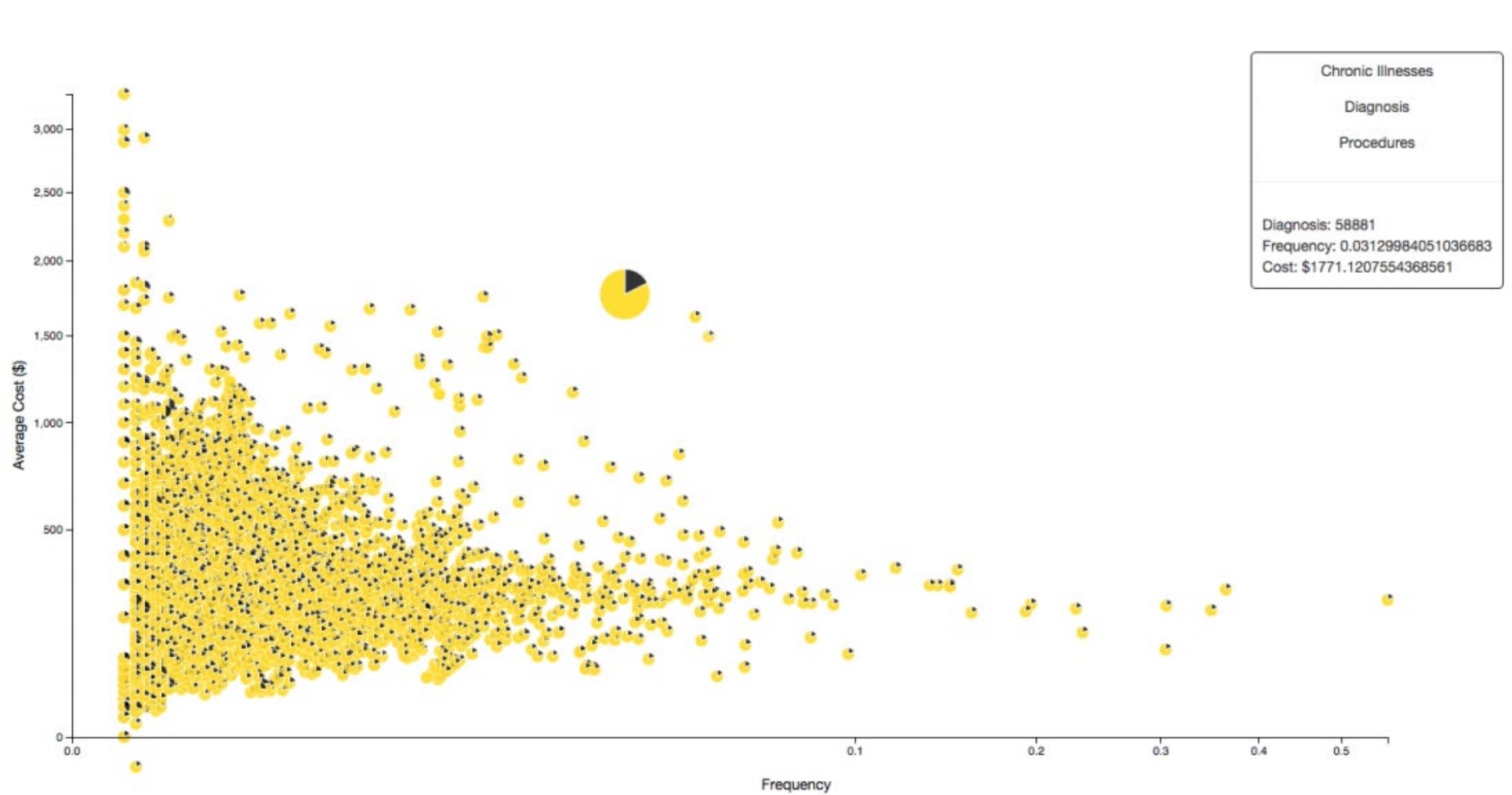
15.62

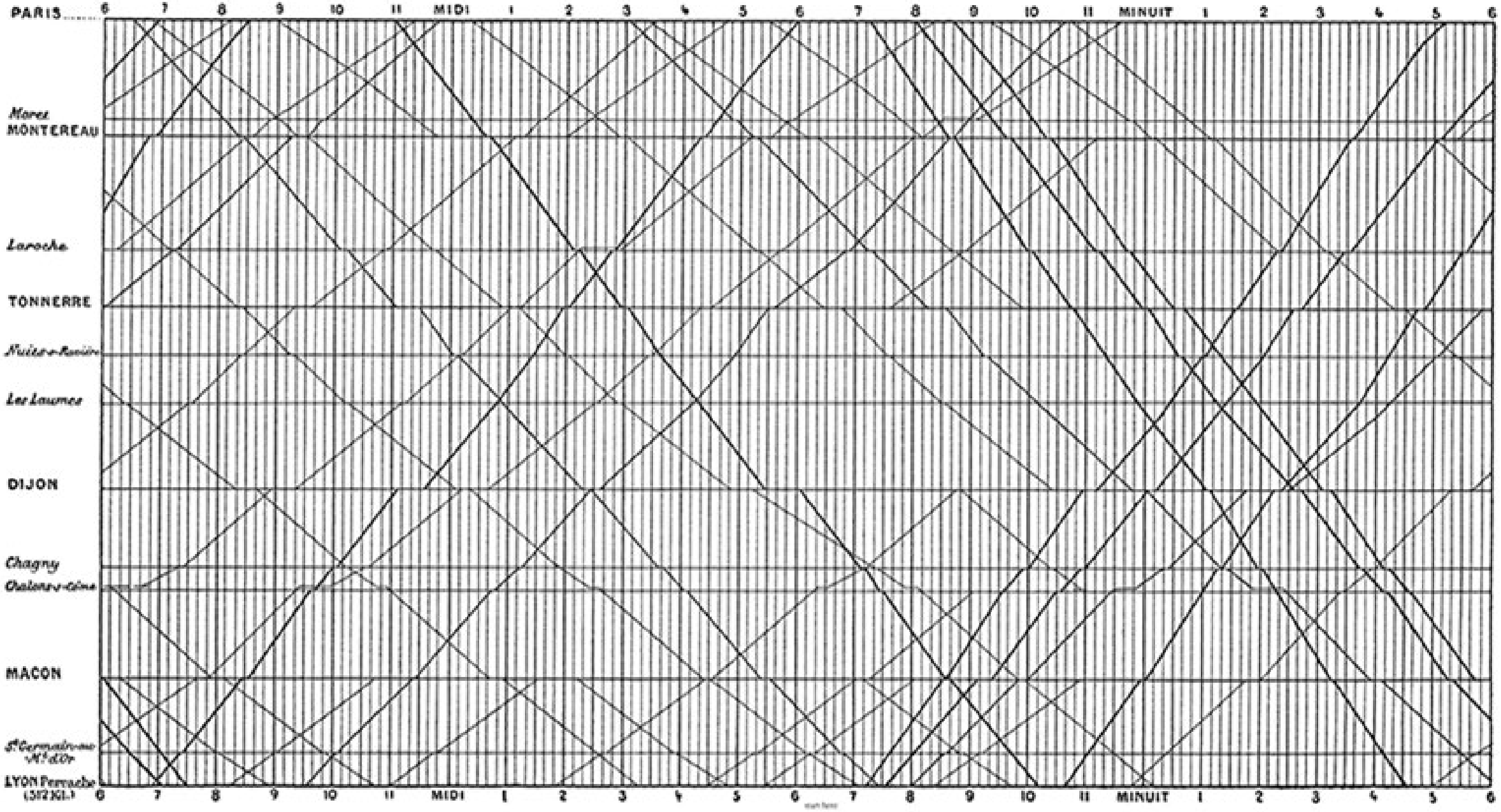
Contrast

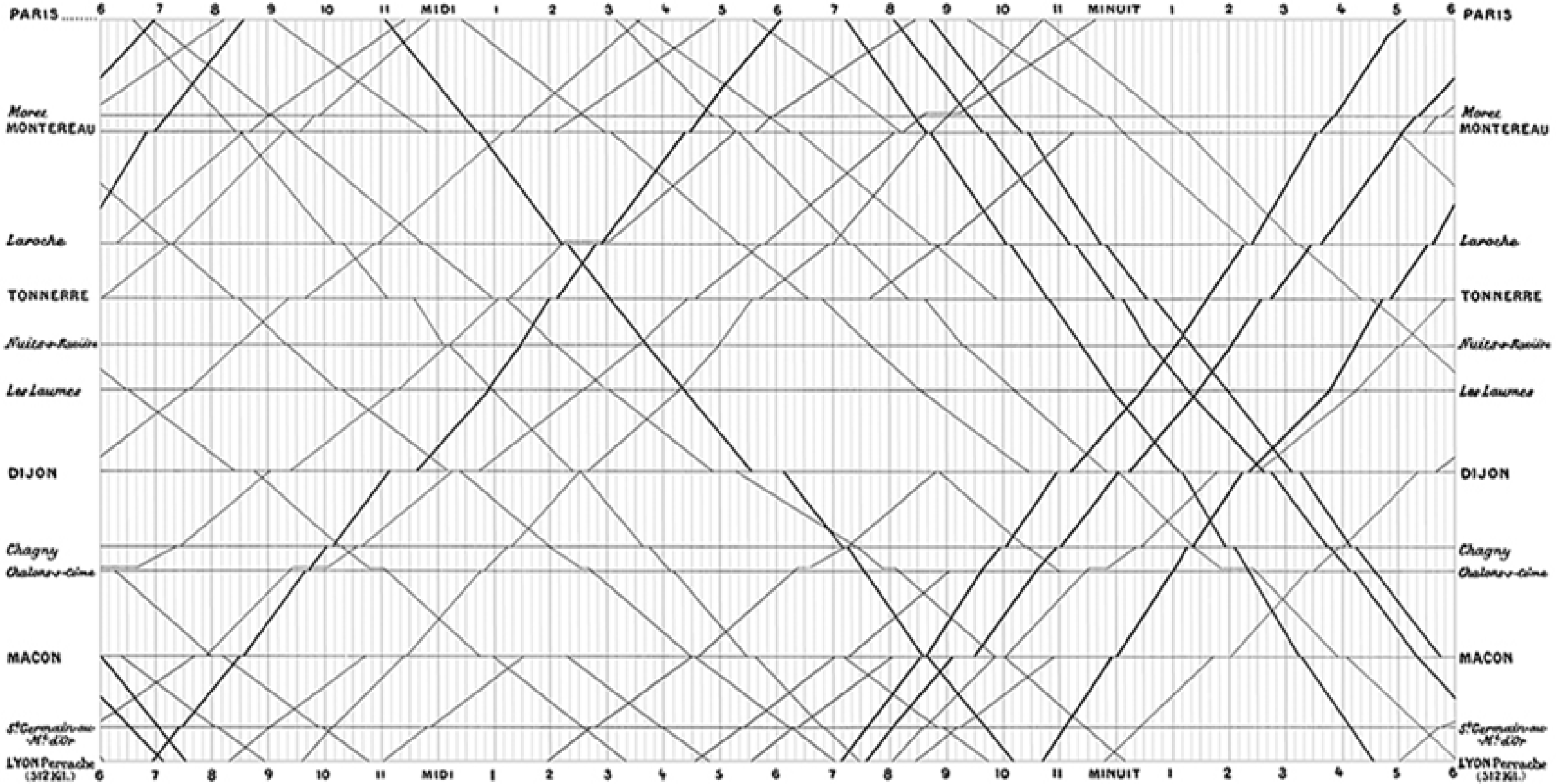
Contrast is the difference in luminance or color that makes an object (or its representation in an image or display) distinguishable. In visual perception of the real world, contrast is determined by the difference in the color and brightness of the object and other objects within the same field of view. Because the human visual system is more sensitive to contrast than absolute luminance, we can perceive the world similarly regardless of the huge changes in illumination over the day or from place to place. The maximum contrast of an image is the contrast ratio or dynamic range.

Visualizations often organized in layers (background color, grids, labels, maps, etc.)

Contrast necessary to create separation


Chronic Illnesses
Diagnosis
Procedures
Diagnosis: 58881
Frequency: 0.03129984051036683
Cost: $1771.1207554368561
Average Cost ($)
3,000
2,500
2,000
1,500
1,000
500
0
0.0
0.1
0.2
0.3
0.4
0.5
Frequency


PARIS
6 7 8 9 10 11 MIDI 1 2 3 4 5 6 7 8 9 10 11 MINUIT 1 2 3 4 5 6
Morez
MONTEREAU
Laroche
TONNERRE
Nuits-s-Ravière
Les Laumes
DIJON
Chagny
Chalons-s-Saône
MACON
St Germain-au-Mt d'Or
LYON Perrache
(512 KL.)
6 7 8 9 10 11 MIDI 1 2 3 4 5 6 7 8 9 10 11 MINUIT 1 2 3 4 5 6


PARIS
6 7 8 9 10 11 MIDI 1 2 3 4 5 6 7 8 9 10 11 MINUIT 1 2 3 4 5 6
PARIS
Morez
MONTEREAU
Morez
MONTEREAU
Laroche
Laroche
TONNERRE
TONNERRE
Nuits-s-Ravière
Nuits-s-Ravière
Les Laumes
Les Laumes
DIJON
DIJON
Chagny
Chalons-s-Saône
Chagny
Chalons-s-Saône
MACON
MACON
St Germain-au-Mt d'Or
St Germain-au-Mt d'Or
LYON Perrache (312 Kil.)
6 7 8 9 10 11 MIDI 1 2 3 4 5 6 7 8 9 10 11 MINUIT 1 2 3 4 5 6
LYON Perrache (312 Kil.)

Matplotlib

- **Matplotlib** is one of the most powerful tools for data visualization in Python.
- **Matplotlib** is an incredibly powerful (and beautiful!) 2-D plotting library.
  - It is easy to use and provides a huge number of examples for tackling unique problems
- In order to get `matplotlib` into your script,
  - first you need to import it, for example:

```
import matplotlib.pyplot as plt
```

- However, if it is not installed, you may need to install it:
  - Easiest way to install **matplotlib** is using `pip`.
  - Type the following command in the command prompt (cmd) or your Linux shell;
    - `pip install matplotlib`
    - *Note that you may need to run the above cmd as an administrator*

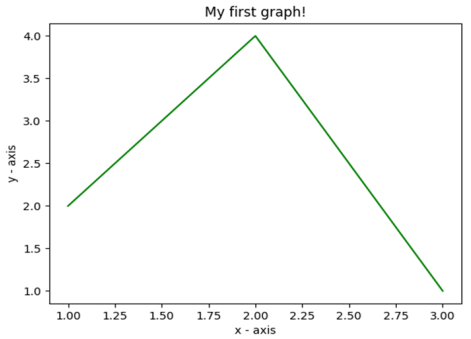
matplotlib

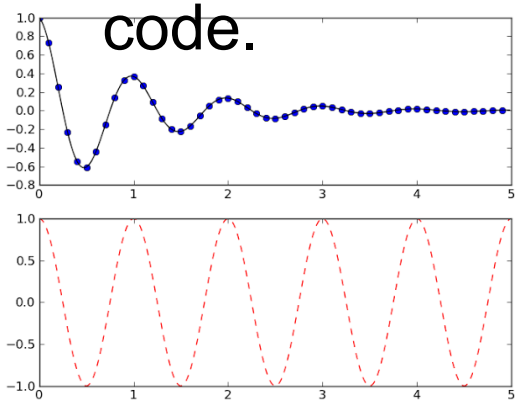
- Strives to emulate MATLAB
  - `matplotlib.pyplot` is a collection of command style functions that make `matplotlib` work like **MATLAB**.
- Each `pyplot` function makes some change to the figure:
  - e.g.,
    - creates a figure,
    - creates a plotting area in the figure,
    - plots some lines in the plotting area,
    - decorates the plot with labels, etc.
- **Note** that **<u>various states</u>** are preserved **<u>across function calls</u>**

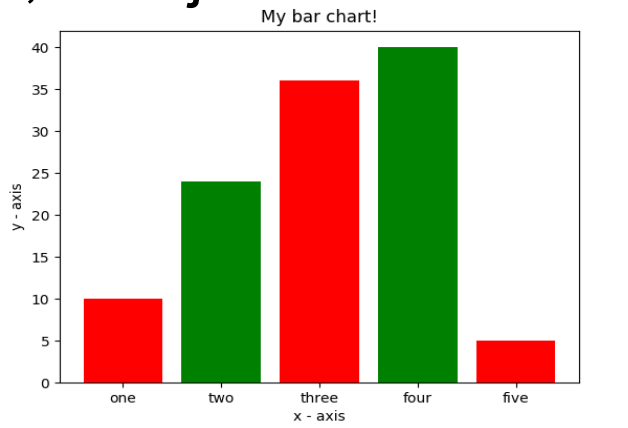
- Whenever you plot with `matplotlib`, the two main code lines should be considered:
  - Type of graph
    - this is where you define a bar chart, line chart, etc.
  - Show the graph
    - this is to display the graph

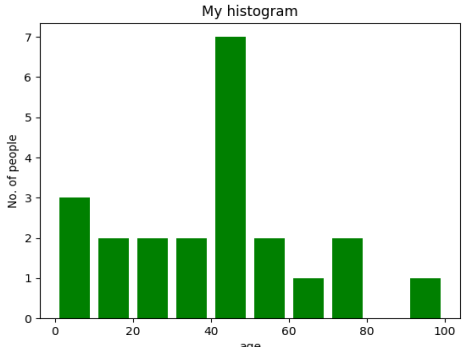
E.g. Matplotlib

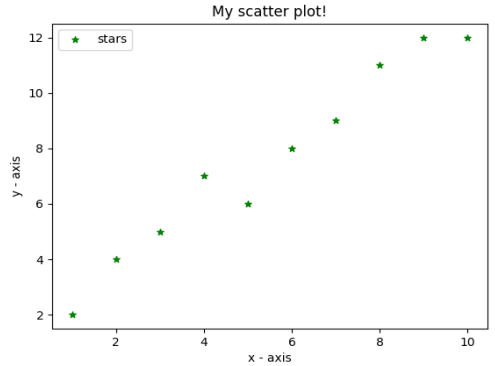
- `Matplotlib` allows you to make easy things
- You can generate **plots**, **histograms**, **power spectra**, **bar charts**, **errorcharts**, **scatterplots**, etc., with just a few lines of code.

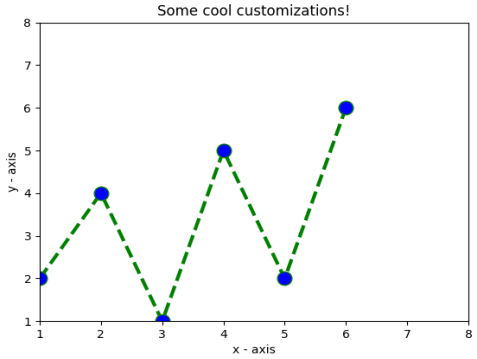


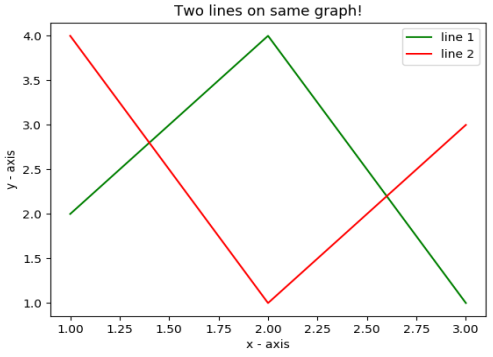






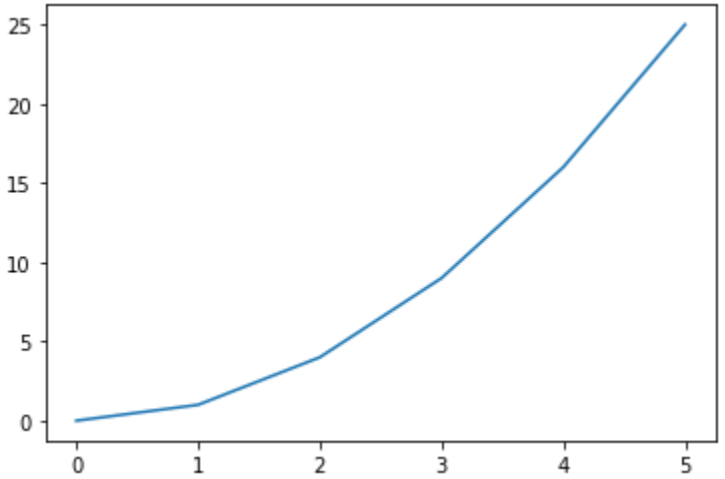






Line Graphs

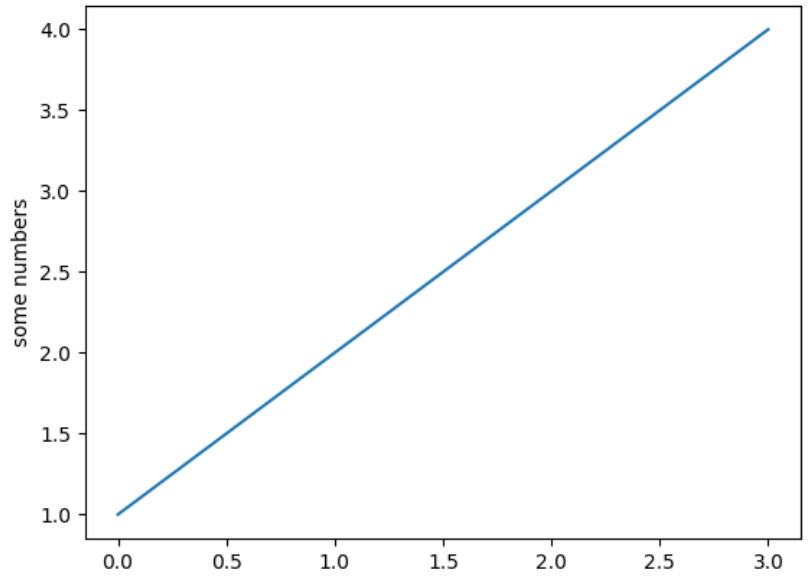
```
import matplotlib.pyplot as plt

#create data for plotting
x_values = [0, 1, 2, 3, 4, 5 ]
y_values = [0, 1, 4, 9, 16,25]

#the default graph style for plot is a line
plt.plot(x_values, y_values)

#display the graph
plt.show()
```



More on Line Graph

- Note: if you provide a single list or array to the `plot()` command,
  - then `matplotlib` assumes it is a sequence of `y values`, and
  - automatically generates the `x values` for you.
- Since python ranges start with `0`, the default `x` vector has the same length as `y` but starts with `0`.
  - Hence the `x` data are `[0, 1, 2, 3]`.

```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()
```

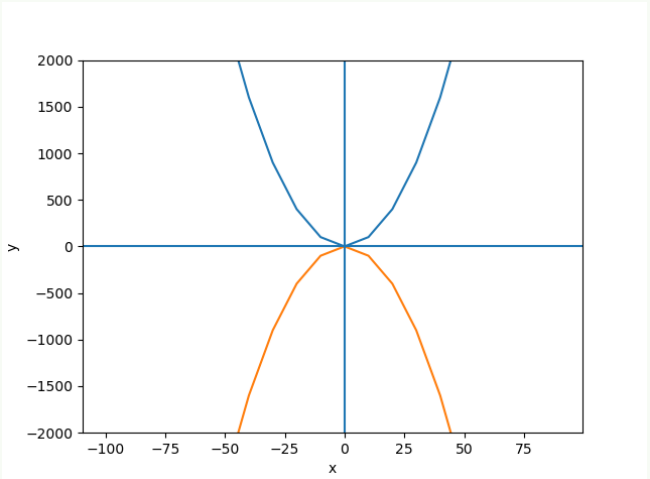


pyplot

- `text()` : adds text in an arbitrary location
- `xlabel()`: adds text to the x-axis
- `ylabel()`: adds text to the y-axis
- `title()` : adds title to the plot
- `clear()` : removes all plots from the axes.
- `savefig()`: saves your figure to a file
- `legend()` : shows a legend on the plot

All methods are available on `pyplot` and on the axes instance generally.

```
import matplotlib.pyplot as plt
y1 =[]
y2 =[]
x = range(-100,100,10)
for i in x: y1.append(i**2)
for i in x: y2.append(-i**2)

plt.plot(x, y1)
plt.plot(x, y2)
plt.xlabel("x")
plt.ylabel("y")
plt.ylim(-2000, 2000)
plt.axhline(0) # horizontal line
plt.axvline(0) # vertical line


plt.savefig("quad.png")
```



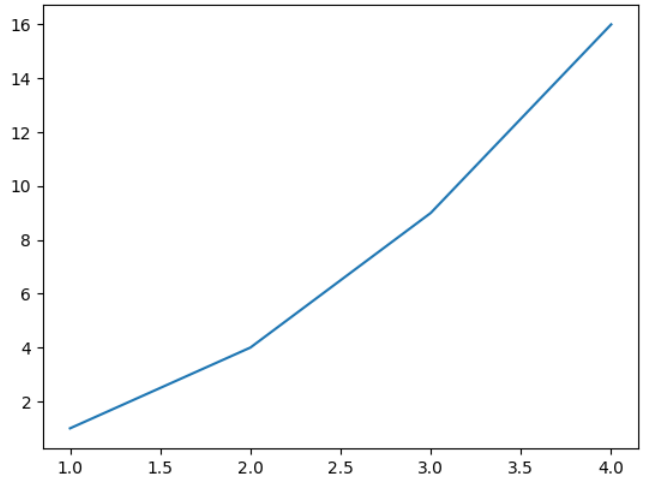
Incrementally modify the figure.

Save your figure to a file

Show it on the screen

Plot

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4]
y = [1, 4, 9, 16]

plt.plot(x, y)
```

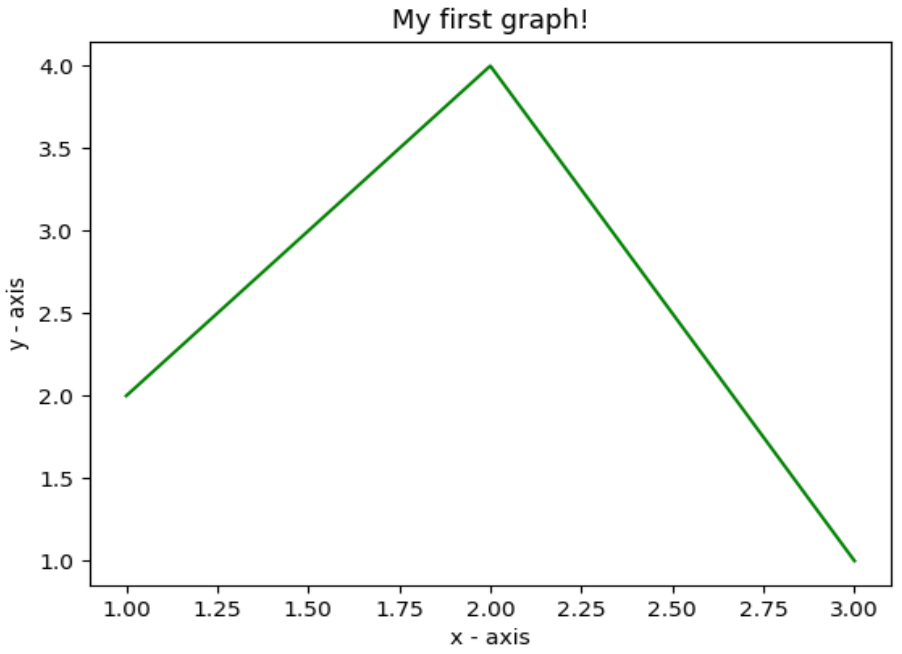


no return value?

- We are operating on a “hidden” variable representing the figure.
- This is a terrible, terrible trick.
- Its only purpose is to pander to MATLAB users.
- I’ll show you how this works in the next lecture

Simple line

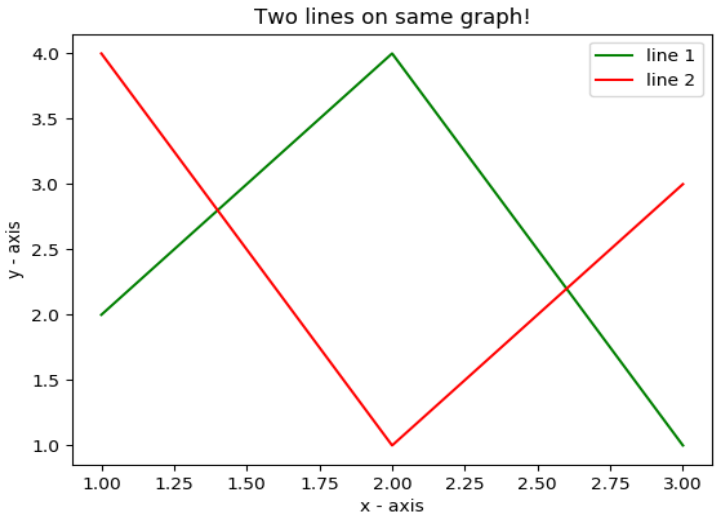
```
# importing the required module
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3]
# corresponding y axis values
y = [2,4,1]

# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')

# giving a title to my graph
plt.title('My first graph!')

# function to show the plot
plt.show()
```



- Define the **`x-axis`** and corresponding **`y-axis`** values as lists.
- Plot them on canvas using **`.plot()`** function.
- Give a name to x-axis and y-axis using **`.xlabel()`** and **`.ylabel()`** functions.
- Give a title to your plot using **`.title()`** function.
- Finally, to view your plot, we use **`.show()`** function.

Simple 2 lines

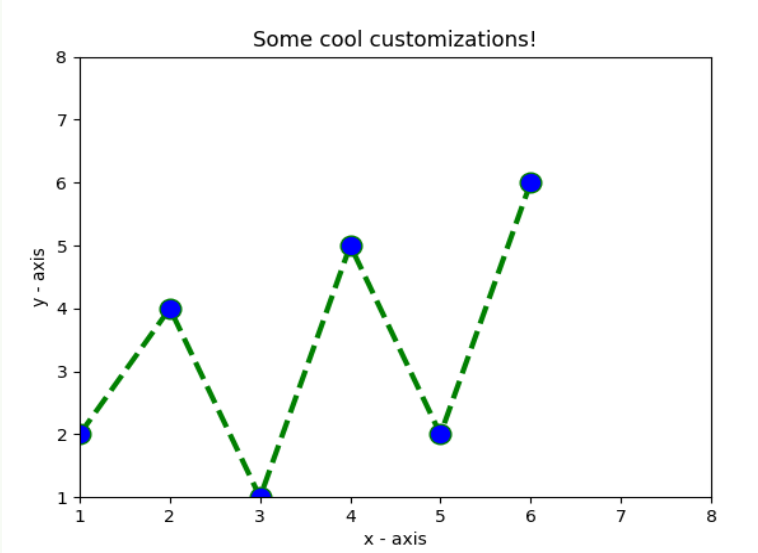
```
import matplotlib.pyplot as plt

# line 1 points
x1 = [1,2,3]
y1 = [2,4,1]
# plotting the line 1 points
plt.plot(x1, y1, label="line 1")

# line 2 points
x2 = [1,2,3]
y2 = [4,1,3]
# plotting the line 2 points
plt.plot(x2, y2, label = "line 2")

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')
# giving a title to my graph
plt.title('Two lines on same graph!')

# show a legend on the plot
plt.legend()

# function to show the plot
plt.show()
```



- Here, we plot two lines on same graph. We differentiate between them by giving them a name(label) which is passed as an argument of `.plot()` function.
- The small rectangular box giving information about type of line and its color is called legend. We can add a legend to our plot using `.legend()` function.

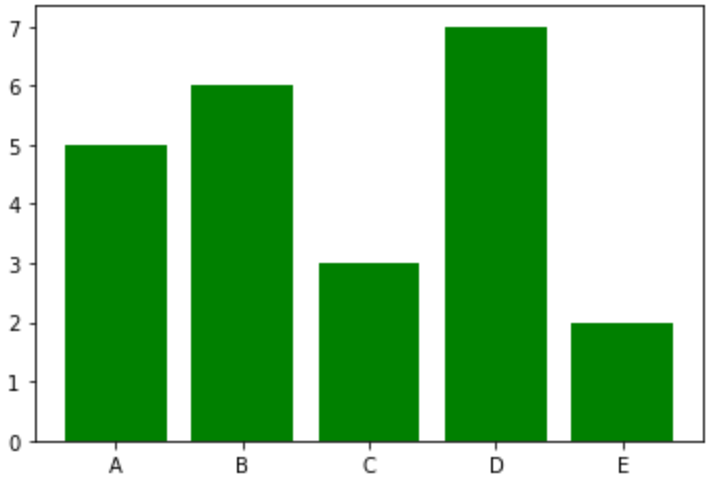
Customization of Plots

```
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3,4,5,6]
# corresponding y axis values
y = [2,4,1,5,2,6]

# plotting the points
plt.plot(x, y, color='green', linestyle='dashed', linewidth = 3,
         marker='o', markerfacecolor='blue', markersize=12)

# setting x and y axis range
plt.ylim(1,8)
plt.xlim(1,8)

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')

# giving a title to my graph
plt.title('Some cool customizations!')

# function to show the plot
plt.show()
```



Bar graphs

```
import matplotlib.pyplot as plt

#Create data for plotting
values = [5, 6, 3, 7, 2]
names  = ["A", "B", "C", "D", "E"]

plt.bar(names, values, color="green")
plt.show()
```

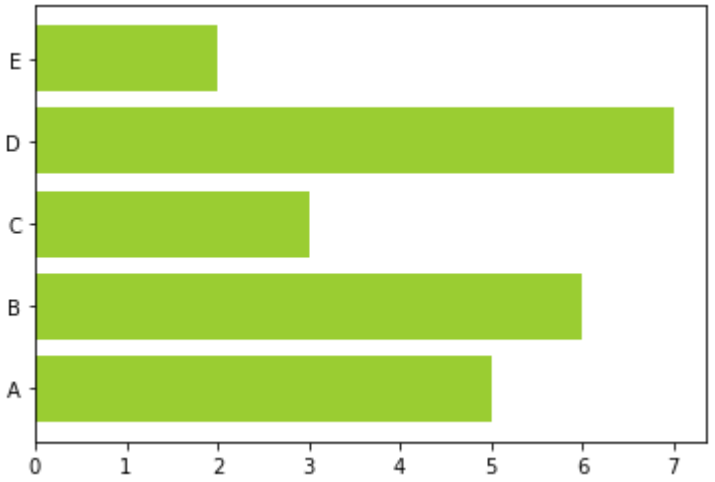


- When using a bar graph, the change in code will be from `plt.plot()` to `plt.bar()` changes it into a bar chart.

Bar graphs

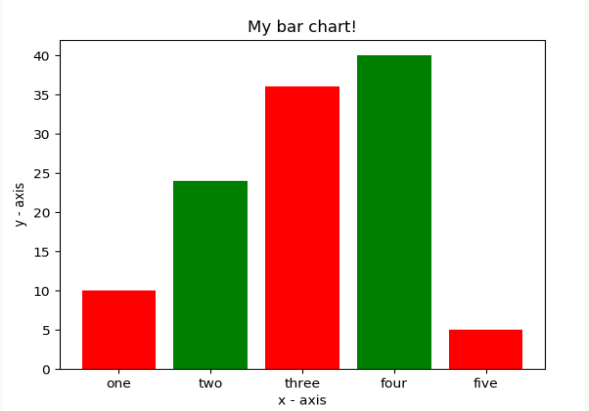
We can also flip the bar graph horizontally with the following

```
import matplotlib.pyplot as plt

#Create data for plotting
values = [5,6,3,7,2]
names  = ["A", "B", "C", "D", "E"]

# Adding an "h" after bar will flip the graph
plt.barh(names, values, color="yellowgreen")
plt.show()
```



Bar Chart

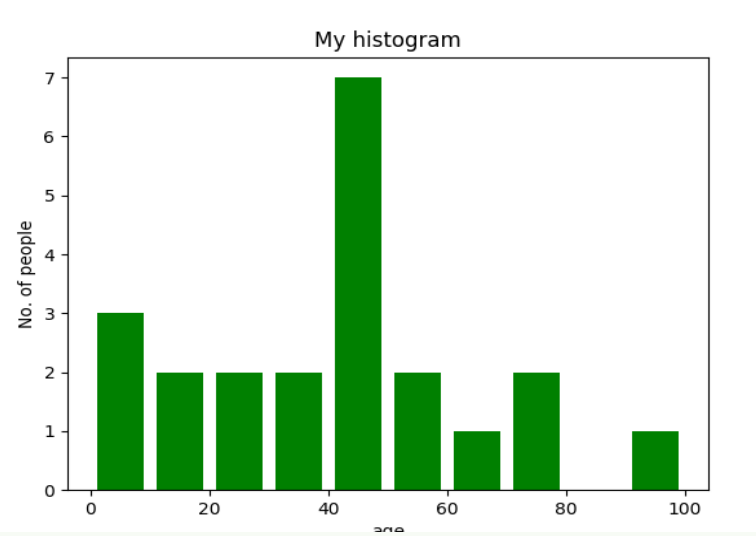
```
import matplotlib.pyplot as plt


# heights of bars
height = [10, 24, 36, 40, 5]

# labels for bars
names = ['one','two','three','four','five']

# plotting a bar chart
c1 =['red', 'green']
c2 =['b', 'g'] # we can use this for color
plt.bar(left, height, width=0.8, color=c1)

# naming the x-axis
plt.xlabel('x - axis')
# naming the y-axis
plt.ylabel('y - axis')
# plot title
plt.title('My bar chart!')

# function to show the plot
plt.show()
```



- Here, we use `plt.bar()` function to plot a bar chart.
- you can also give some name to x-axis coordinates by defining tick_labels

Histogram

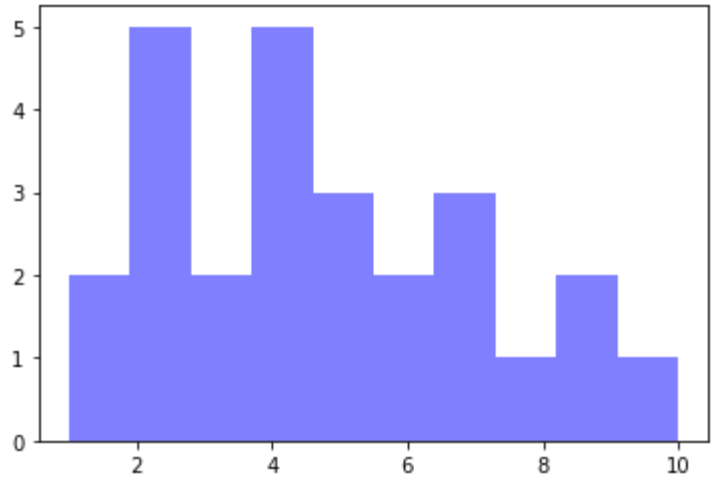
```
import matplotlib.pyplot as plt

# frequencies
ages=[2,5,70,40,30,45,50,45,43,40,44,60,7,13,57,18,90,77,32,21,20,40]

# setting the ranges and no. of intervals
range = (0, 100)
bins = 10

# plotting a histogram
plt.hist(ages, bins, range, color='green',histtype='bar',rwidth=0.8)

# x-axis label
plt.xlabel('age')
# frequency label
plt.ylabel('No. of people')
# plot title
plt.title('My histogram')

# function to show the plot
plt.show()
```



Histograms

```
import matplotlib.pyplot as plt

#generate fake data
x = [2,1,6,4,2,4,8,9,4,2,4,10,6,4,5,7,7,3,2,7,5,3,5,9,2,1]

#plot for a histogram
plt.hist(x, bins = 10, color='blue', alpha=0.5)
plt.show()
```

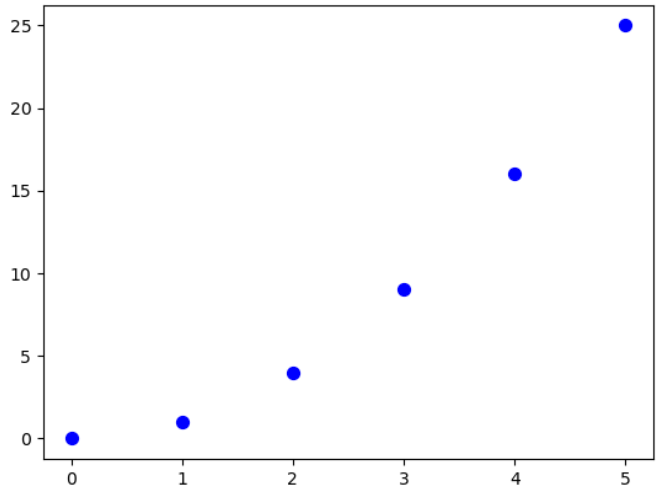
- Looking at the code snippet, I added two new arguments:
  - **Bins** — is an argument specific to a histogram and allows the user to customize how many bins they want.
  - **Alpha** — is an argument that displays the level of transparency of the data points.



Scatter Plots

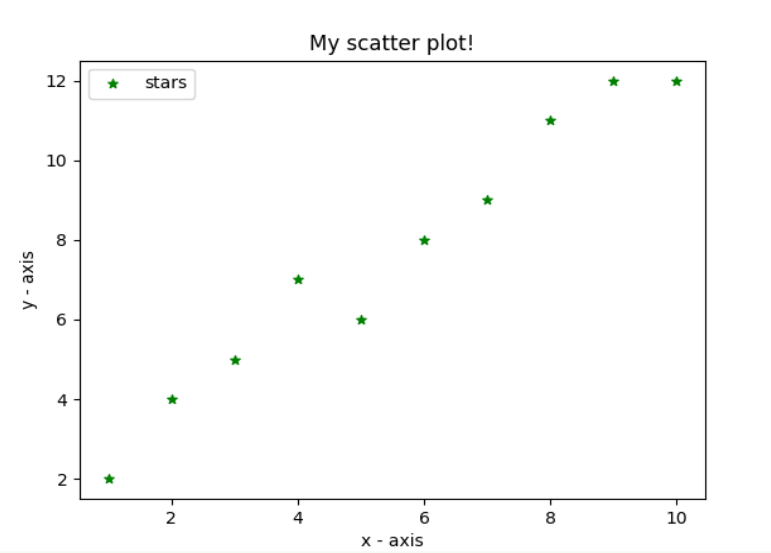
```
import matplotlib.pyplot as plt

#create data for plotting

x_values = [0,1,2,3,4,5]
y_values = [0,1,4,9,16,25]

plt.scatter(x_values, y_values, s=30, color="blue")
plt.show()
```

- Can you see the pattern? Now the code changed from `plt.bar()` to `plt.scatter()`.



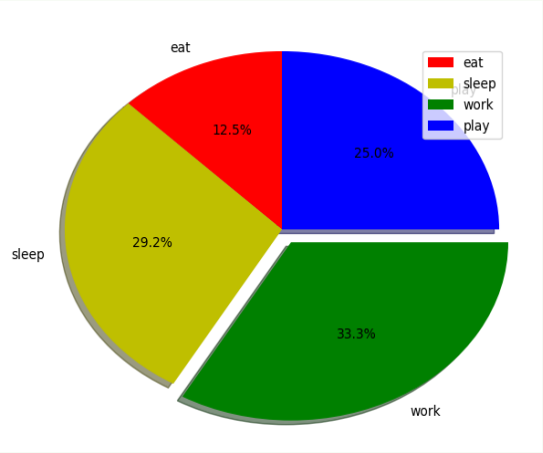
Scatter plot

```
import matplotlib.pyplot as plt

# x-axis values
x = [1,2,3,4,5,6,7,8,9,10]
# y-axis values
y = [2,4,5,7,6,8,9,11,12,12]

# plotting points as a scatter plot
plt.scatter(x, y, label= "stars", color="green", marker="*", s=30)

# x-axis label
plt.xlabel('x - axis')
# frequency label
plt.ylabel('y - axis')
# plot title
plt.title('My scatter plot!')
# showing legend
plt.legend()

# function to show the plot
plt.show()
```



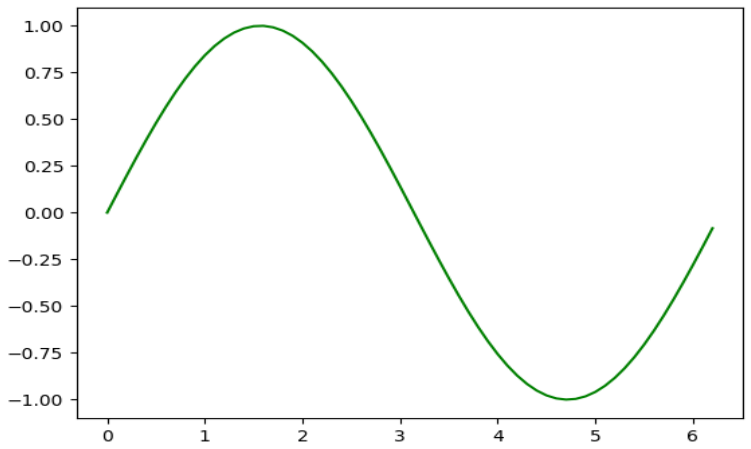
Pie-chart

```
import matplotlib.pyplot as plt

# defining labels
activities = ['eat', 'sleep', 'work', 'play']

# portion covered by each label
slices = [3, 7, 8, 6]

# color for each label
colors = ['r', 'y', 'g', 'b']

# plotting the pie chart
plt.pie(slices, labels = activities, colors=colors,
        startangle=90, shadow = True, explode = (0, 0, 0.1, 0),
        radius = 1.2, autopct = '%1.1f%%')

# plotting legend
plt.legend()

# showing the plot
plt.show()
```



Plotting curves of given equation

```
# importing the required modules
import matplotlib.pyplot as plt
import numpy as np

# setting the x - coordinates
x = np.arange(0, 2*(np.pi), 0.1)
# setting the corresponding y - coordinates
y = np.sin(x)

# potting the points
plt.plot(x, y)

# function to show the plot
plt.show()
```



Examples taken from:
[Graph Plotting in Python | Set 1]