using visible light spectrum
to cluster
relative distributed, ordered
collections of datapoints

If I told you that it is possible to make relative distributed, ordered data of differing number of elements comparable.

Just like that. Curious?

# Who am I?

Dirk Biesinger

Bachelor in Business Information Systems

Associate Industrial Mechanical Technician

Sr. Data Science Consultant

Instructor University of Washington

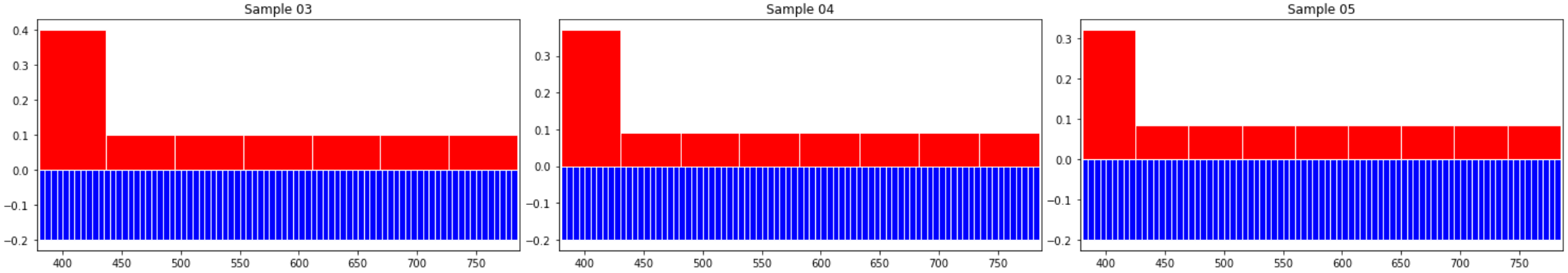
saravjishut.org

In an ever more data driven and data conscious world, collecting and storing data is no longer a problem. The area of concern has shifted to connecting the data and using the data to glean insights out of it.

One of the general problems in connecting data is varying formats of the data on many levels.

# A different challenge lies in how to present and codify what is a simple matter for humans but is in fact is relatively complex on a data basis.

Take the following illustrations as an example:




The characteristic of the representation looks very similar or comparable.

But: these are vastly different from a data point of view: All have a different number of items in the red data. There are 7, 8 and 9 in “Sample 03”, “Sample 04” and “Sample 05” respectively.

These differences pose a challenge when trying to use the data together.

# Some Examples where this is applicable:

- data related to sentiment analysis where demographic information is available and the analysis spans multiple data sets based on different rating scales, e.g. 5-star rating and 7 or 10 step satisfaction rating
- packaging size in the food industry
- garment size in clothing
- relative performance in technology products (e.g iPhone 5 had 9 variants, iPhone 12 has 12 variants, iPhone 13 has 7. Or same example on a time basis: in 2013 there were 5, 5, 5, 6 variants available (per quarter), This changed to 20, 20, 20, 17 in 2021 (per quarter).
- quality rating e.g. in meat products

It occurred to me that there is another area that has similar characteristics:

Converting a visible light spectrum curve to a color code.

This is also a n:1 relationship: Many different light spectrum profiles can represent the same color code.

Further, the light spectrum profile curve can be broken down into a composition of three distinct curves for red, green and blue.

# IDEA

Could the visible light spectrum and its methods for transformation be used to make relative distributed, ordered data comparable?

# I set goals for evaluating this idea:

1. Explore if it is possible to convert relative distributed ordered data to visible light spectrum and if there is value in doing so.
2. Explore if this information could be used to create clusters. This included creating centroids and assigning a distinct color code based on this.
3. Explore if there is a way to reverse from the color information assigned to a centroid to a visible light spectrum curve and use this to create relative distributions for bucket qty n, where n is determined by the requirements from the incoming data

*This evaluation limits its scope to exploring the possibility to use visible light spectrum to compare relative distribution over different number of buckets. As the visible light spectrum to color code is a n:1 relationship, this evaluation also attempts to find one solution to generate a visible light spectrum out of a color code.*

*This evaluation will not assess how much of an influence these results have in predictive models.*

# SOLUTION

- convert the individual subsets of the data belonging to one relative distribution into visible light spectrum
- into a color code and x-y location on the CIE standard chromaticity diagram

# SOLUTION

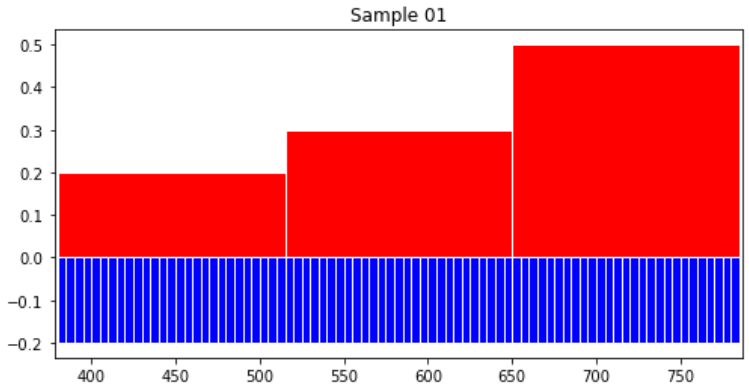
- Convert relative distribution with few buckets into a 5nm buckets in visible light spectrum [340-780]nm | 81 buckets
- Convert this approximated spectrum into color code and x-y coordinates in the CIE standard chromaticity diagram
- Use these x-y coordinates for clustering and analysis.
- Find centroids in these clusters
- Convert back to spectrum
- Convert back to few buckets

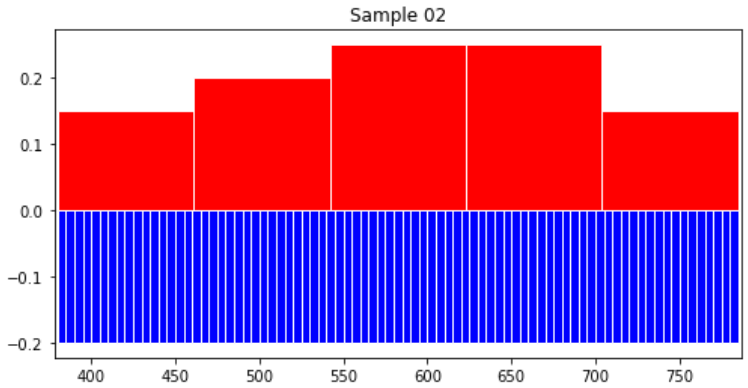
# Simple, right?

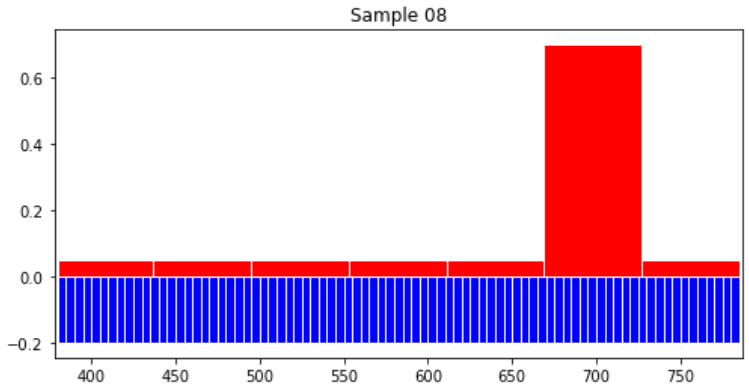
# Let’s start with a toy data set

li_01 = [0.2, 0.3, 0.5]
li_02 = [0.15, 0.2, 0.25, 0.25, 0.15]
li_03 = [0.4, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
li_04 = [0.1, 0.1, 0.1, 0.4, 0.1, 0.1, 0.1]
li_05 = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.4]
li_06 = [0.7, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05]
li_07 = [0.05, 0.05, 0.05, 0.7, 0.05, 0.05, 0.05]
li_08 = [0.05, 0.05, 0.05, 0.05, 0.05, 0.7, 0.05]
li_09 = [0.22, 0.32, 0.46]

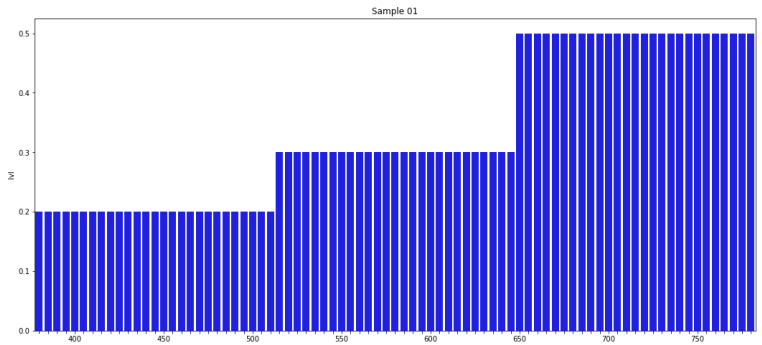
# Toy Data | few buckets and 5nm buckets

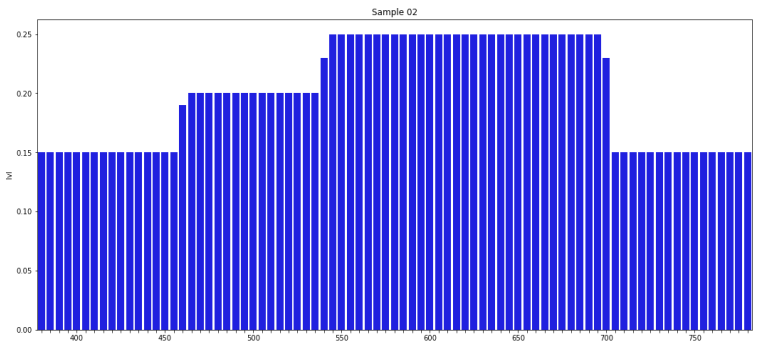


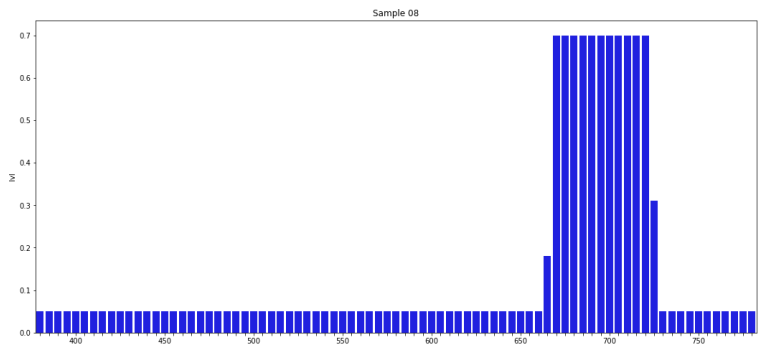



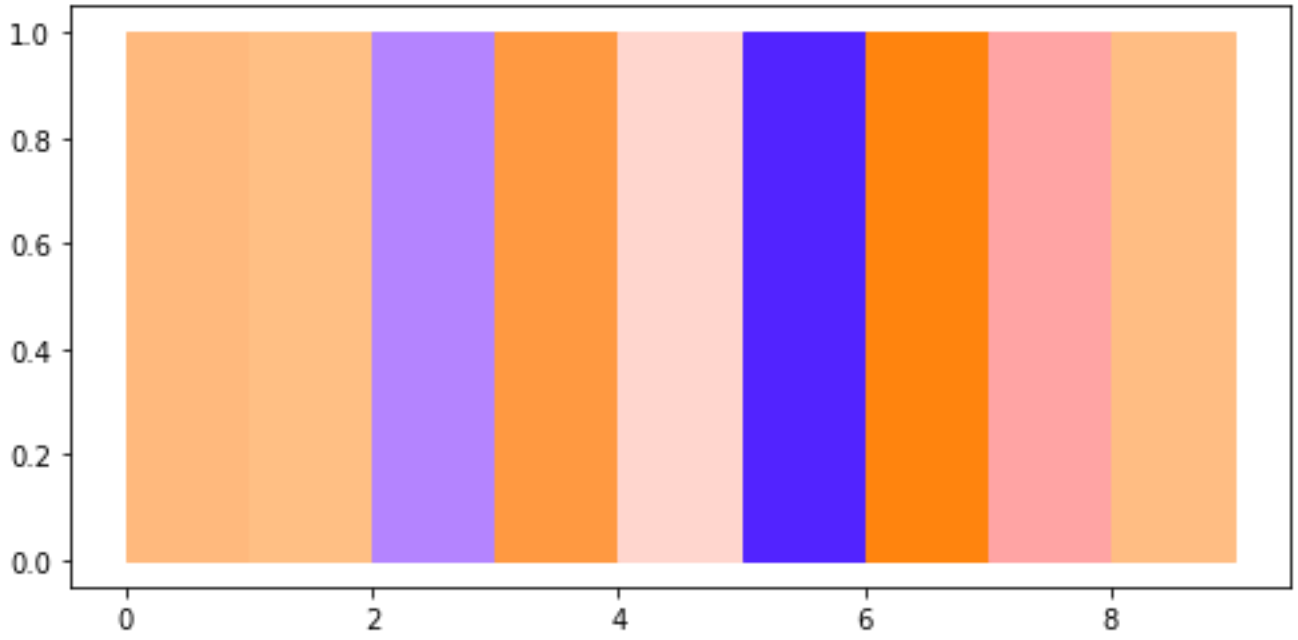
# Toy Data | converted to 5nm buckets



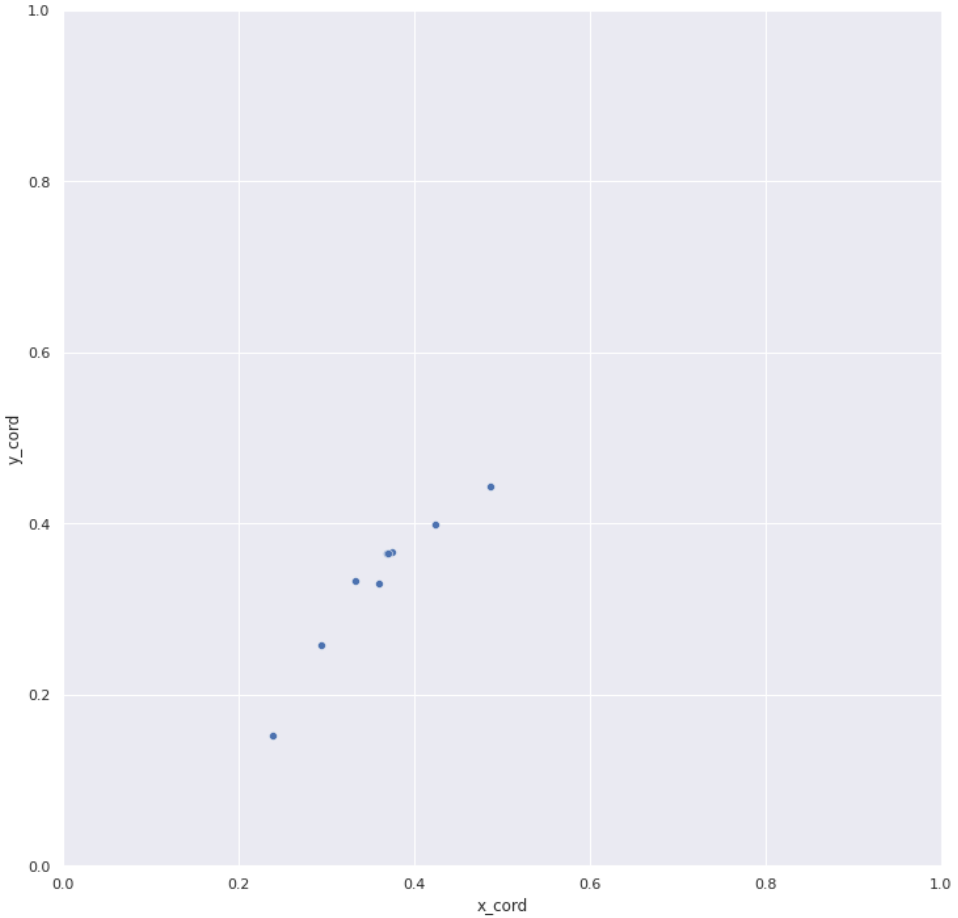



# Toy Data | Show me the colors



# Toy Data | CIE chromaticity diagram




# MORE Data! | Need input

From this point forward a larger and richer dataset is used.

This dataset, based on actual sales data, it is cleaned, anonymized, and preprocessed to combine the individual items into an ordered relative distributed array.

The incoming data size was 2.1M observations in numerous relative distribution bucket configurations.

# The DATA

- 103 distinct locations,
- 9 distinct departments (with varying coverage between locations),
- 11,519 distinct articles
- total of 512,126 observations.

# The DATA

This dataset is not only a real-world example, but also allows, due to its composition, for multiple dimensions of grouping.

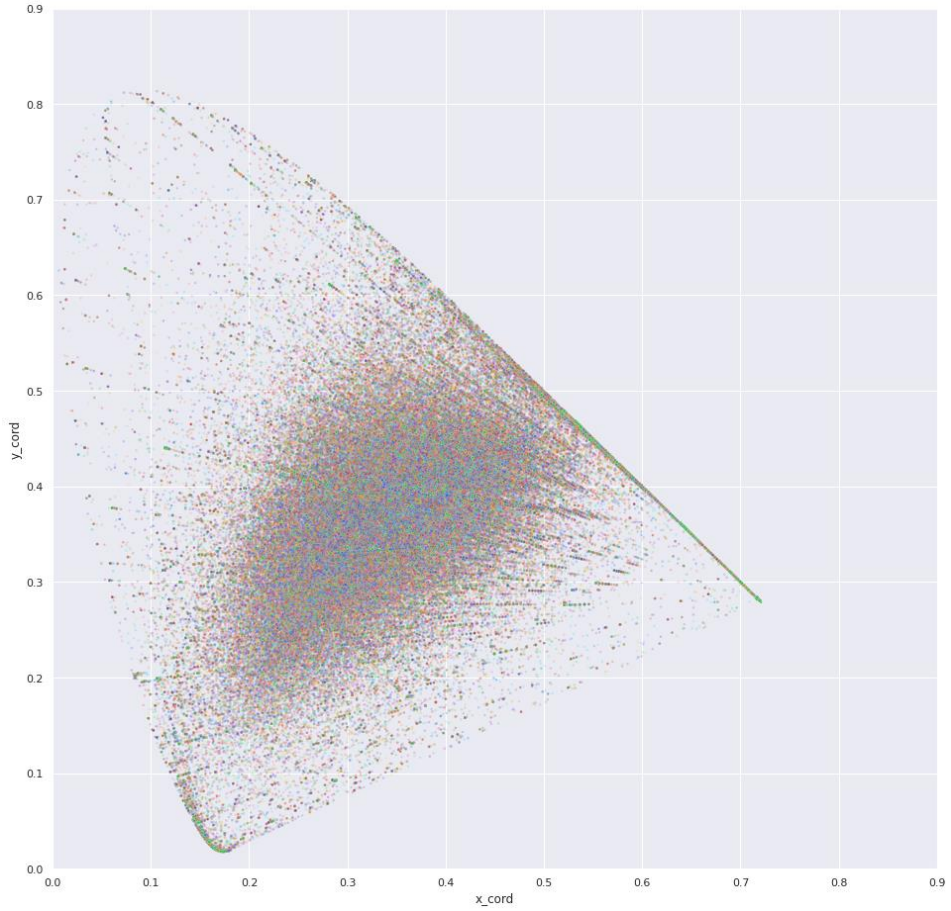
This dataset also contains the actual (absolute) sales quantities of the items.

# The DATA

This can be used to create a weighted evaluation.

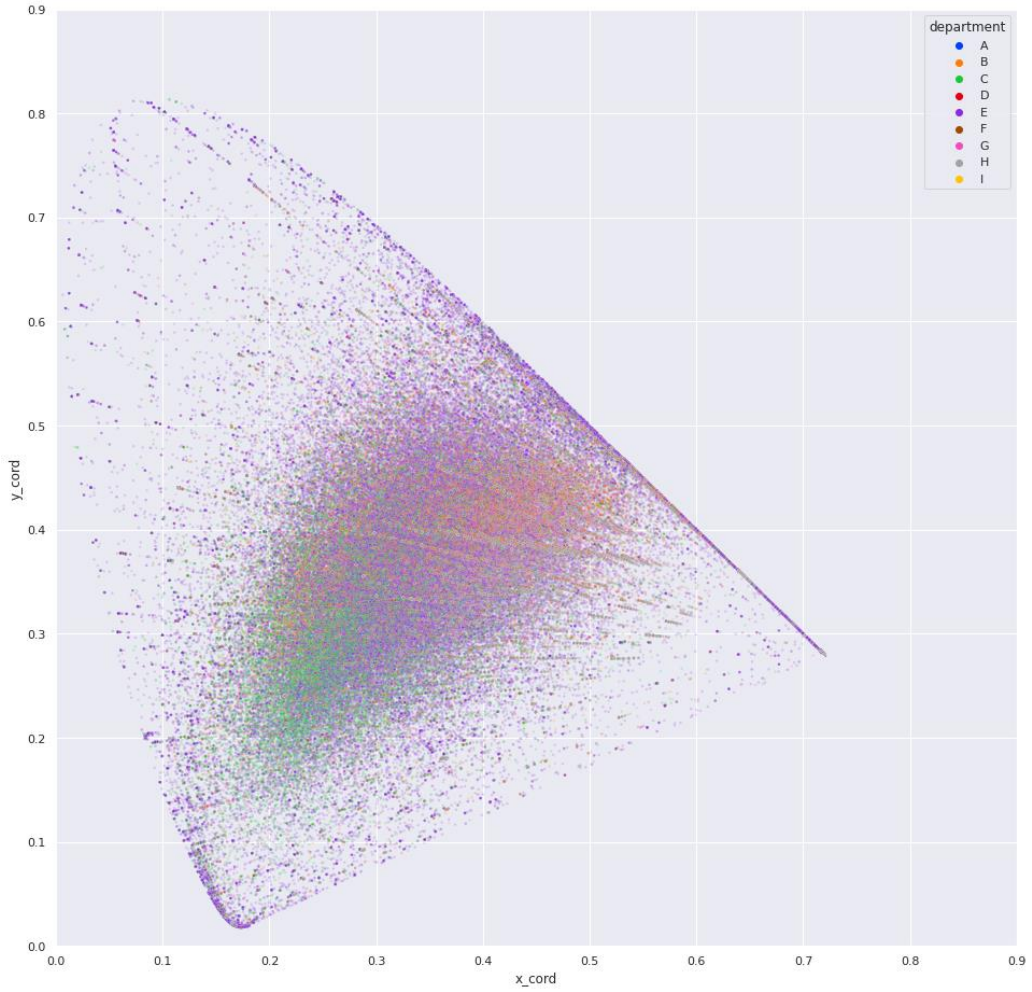
- mean sales quantity is 12.32 items,
- max is 1,184
- median is 7

# The DATA | colored by location



# The DATA | color by department

First clusters appear

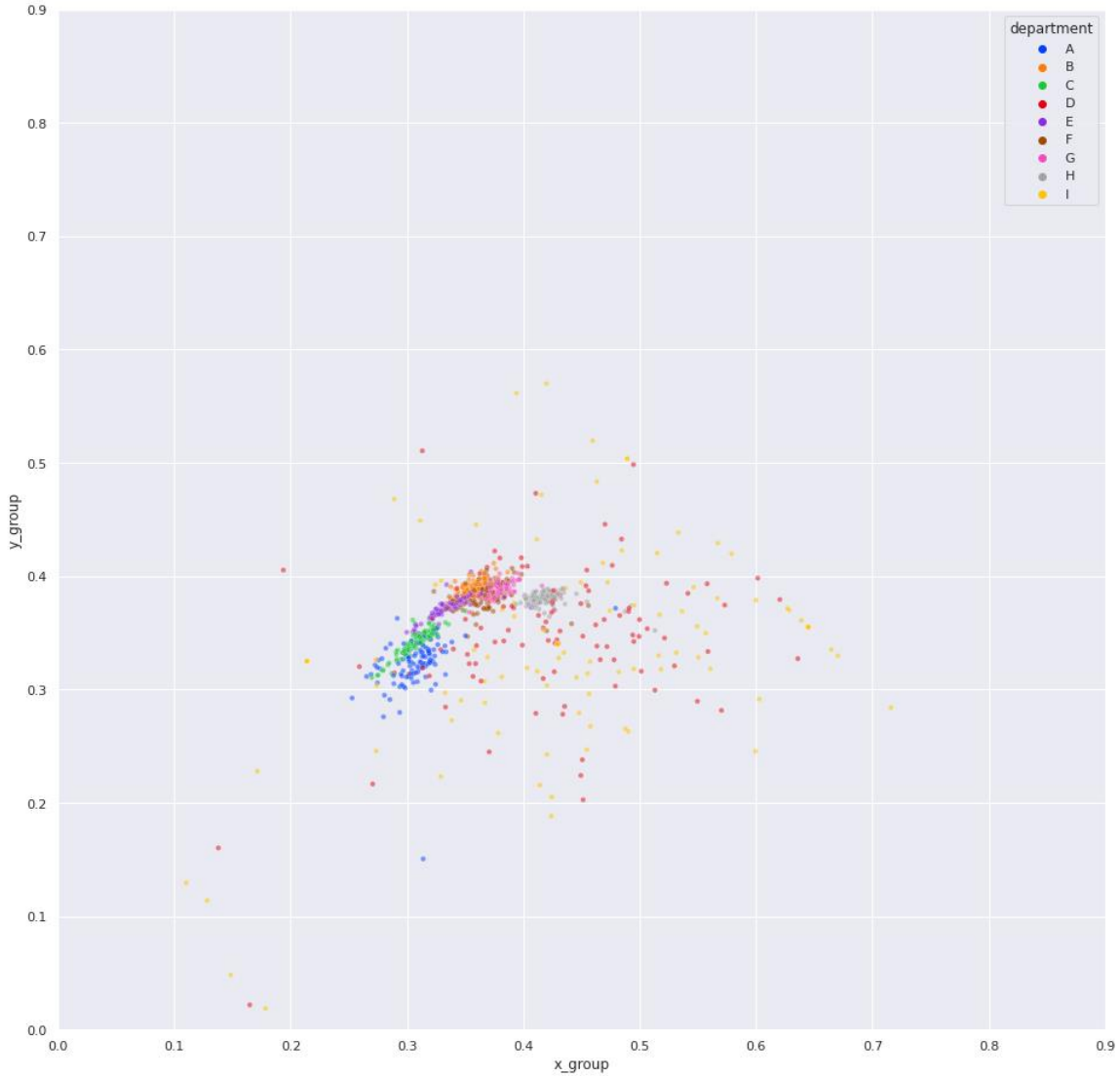


# Potential for clustering on Location

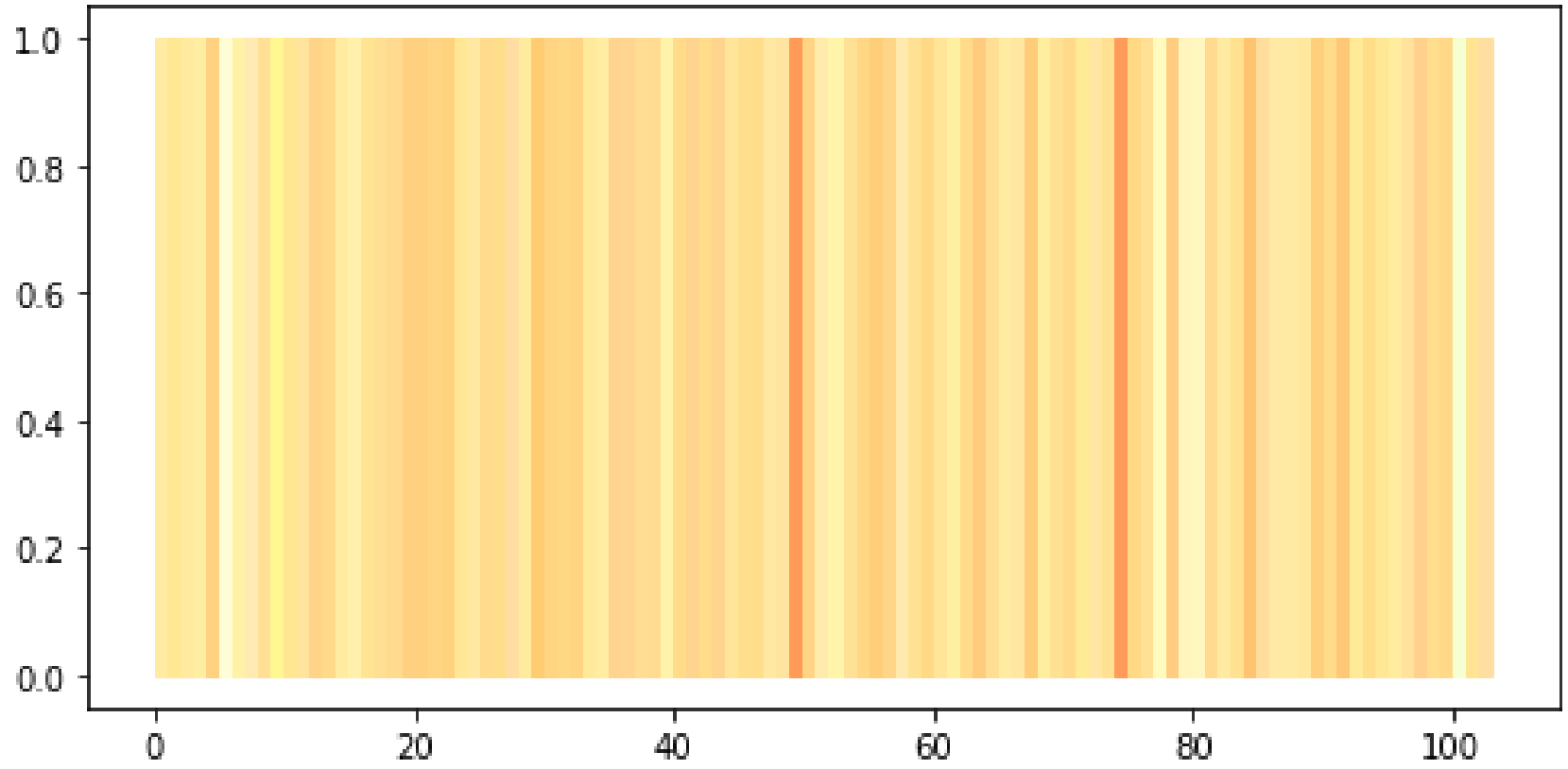
Data grouped by Location.

Color by department.

Clear clustering happening.

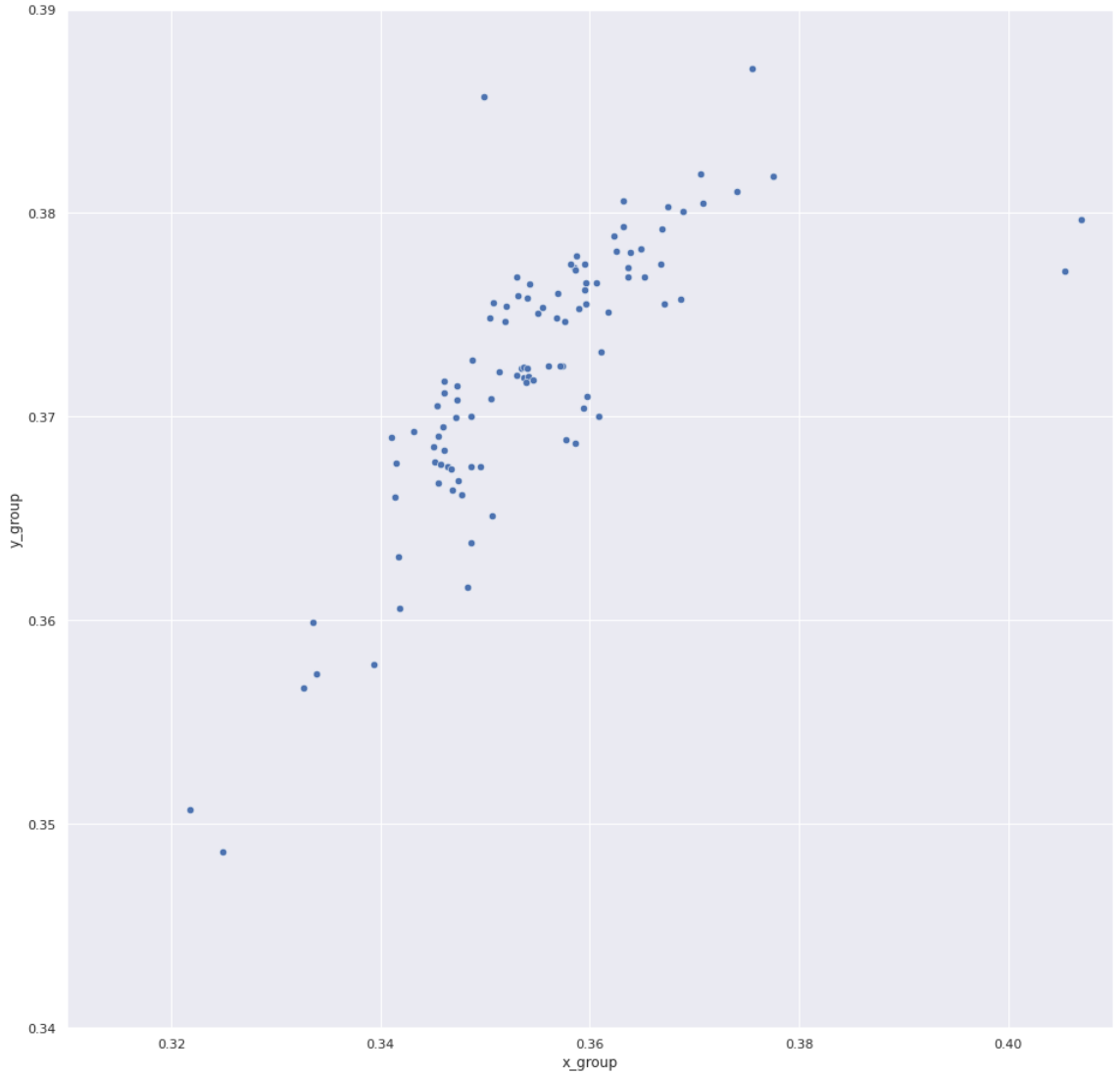


# Color code for all 103 locations



# Can we use clustering?

Data grouped by Location.

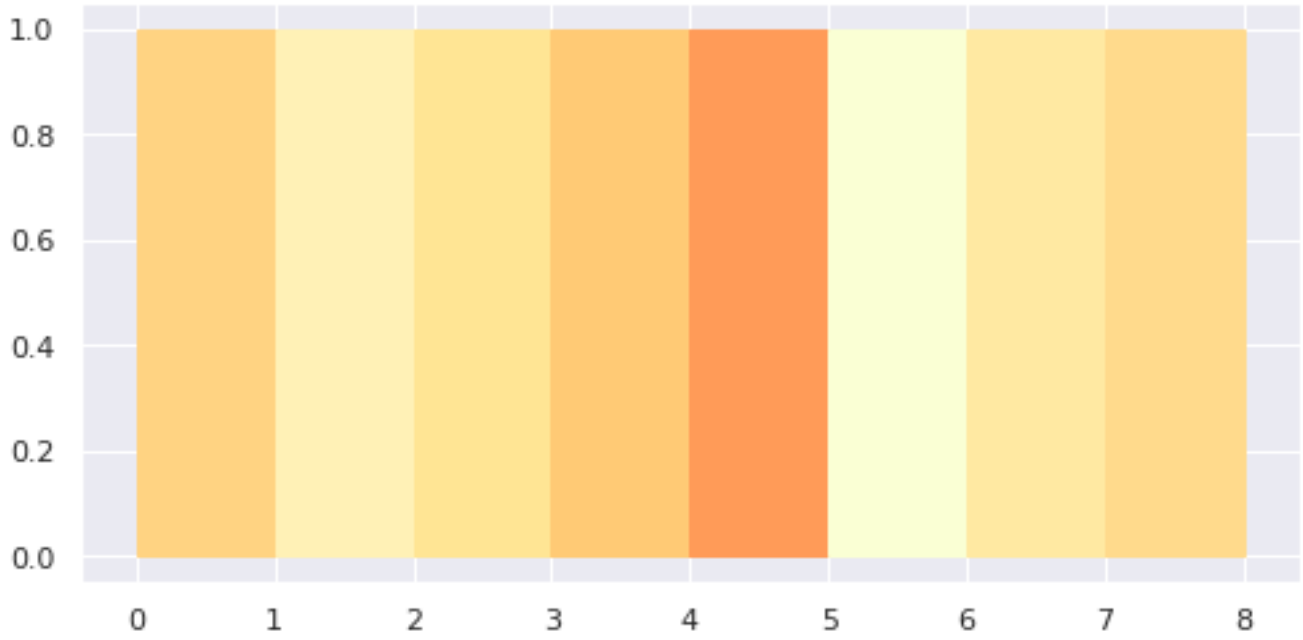


# Cluster Algo output

## Gaussian Mixture with 7 clusters


## KMeans with 8 clusters

# Centroids from Kmeans | color code



# Reversing color code to spectrum

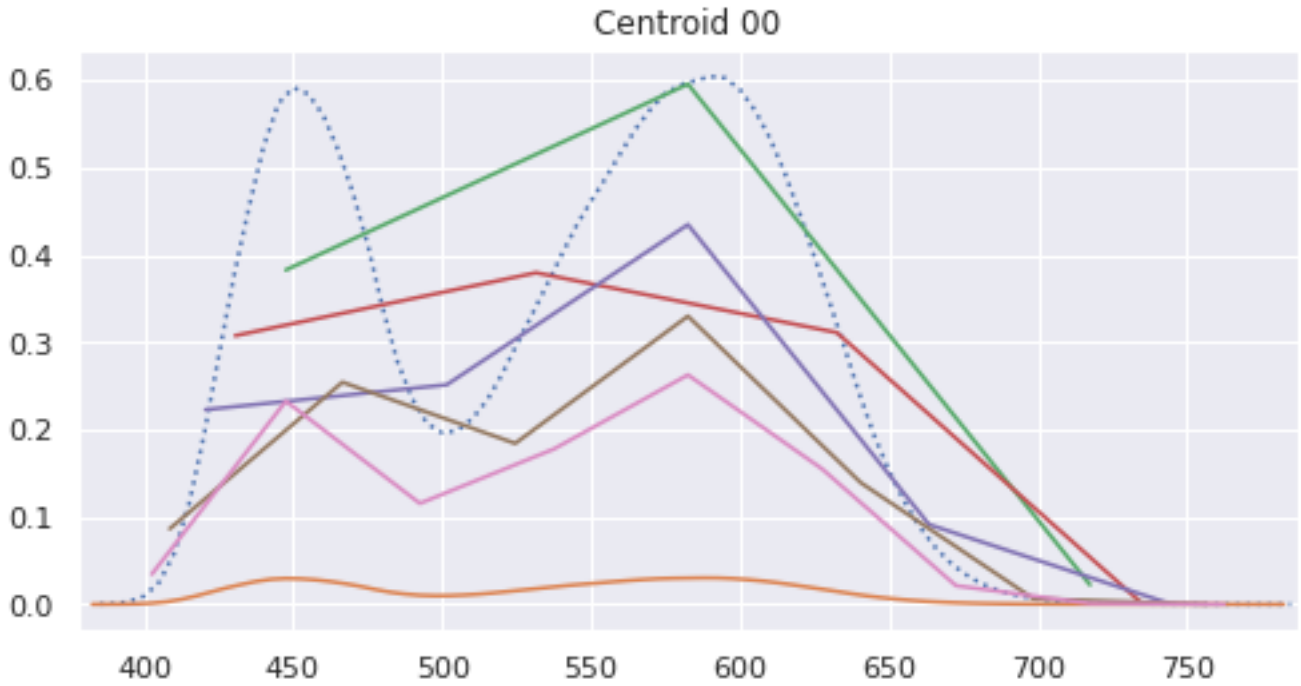
- Now, to extract a color spectrum curve is not a simple conversion, as there is a n:1 relationship between color curves to color code.
- To get a color spectrum curve, some baseline needed to be used to anchor the n:1 relationship to a 1:1 relationship.
- in this case, the Illuminnat D65 calibration is utilized.
- Based on this, a "curve" based on the 81 points on the 5nm edges is created.

# Reversing color code to spectrum

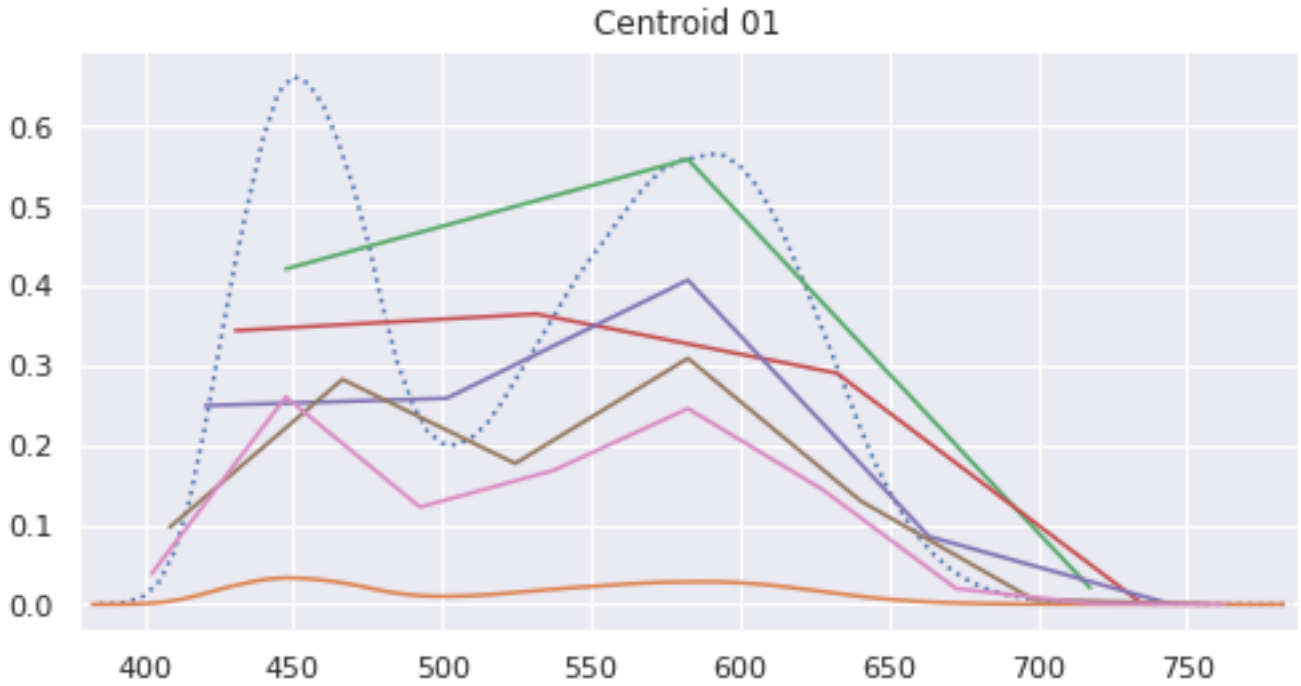
- The last step is to reverse the attribution of the 5nm buckets into the buckets of the observation distributions.

- Please note: in the following graphs, the blue dotted line, is a magnified (by factor 20) curve of the 81 bin representation. The orange line at the bottom is the actual scale. All other lines are the results when applying 3, 4, 5, 7, 9 bucket distributions.

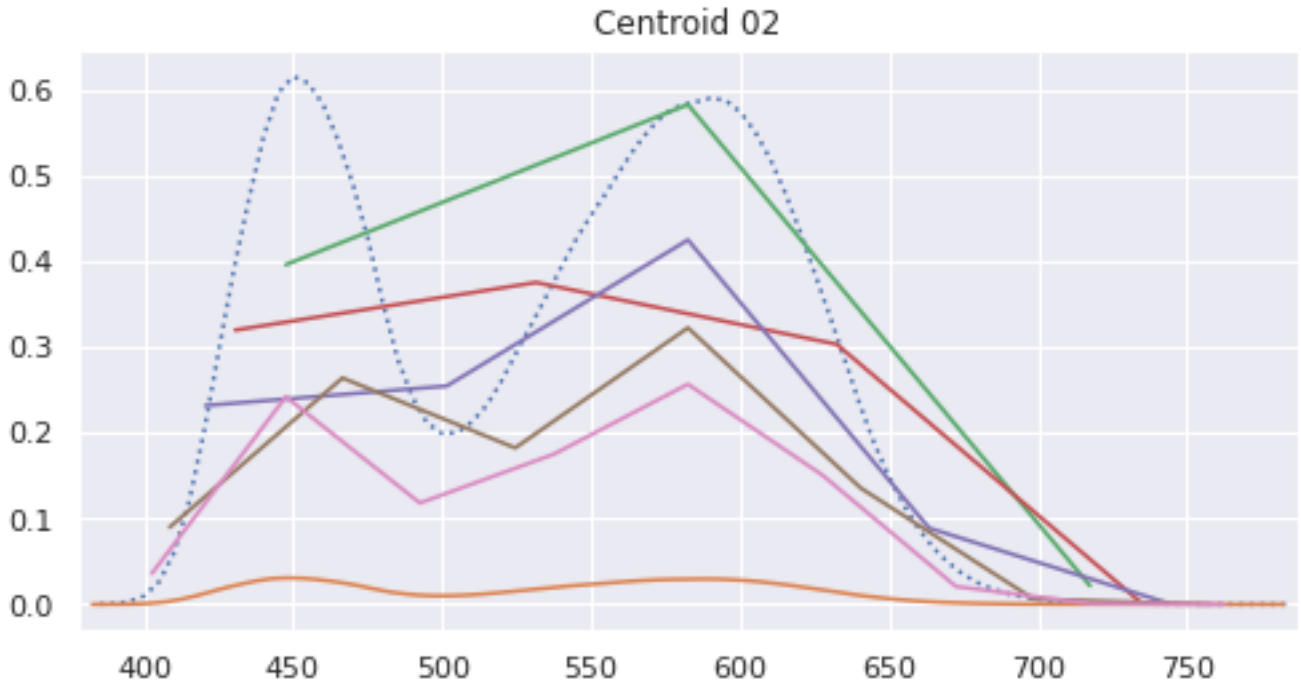
# Reversing color code to spectrum to distribution



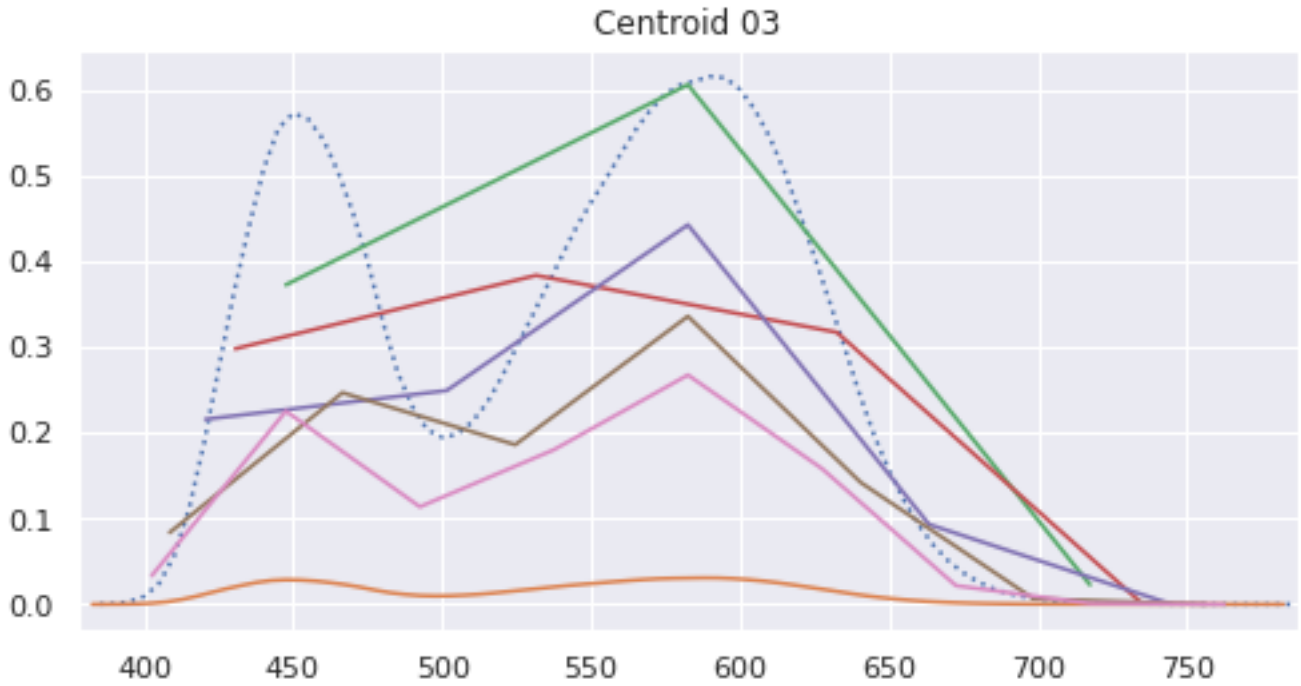
# Reversing color code to spectrum to distribution



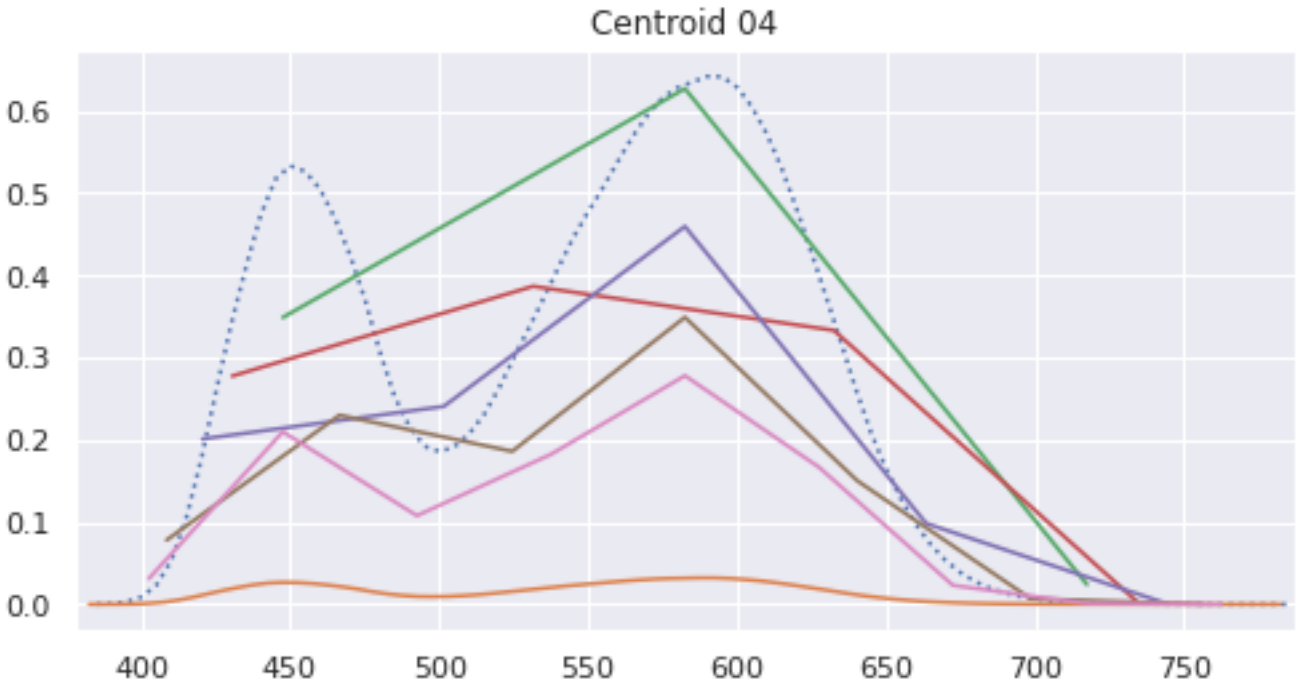
# Reversing color code to spectrum to distribution



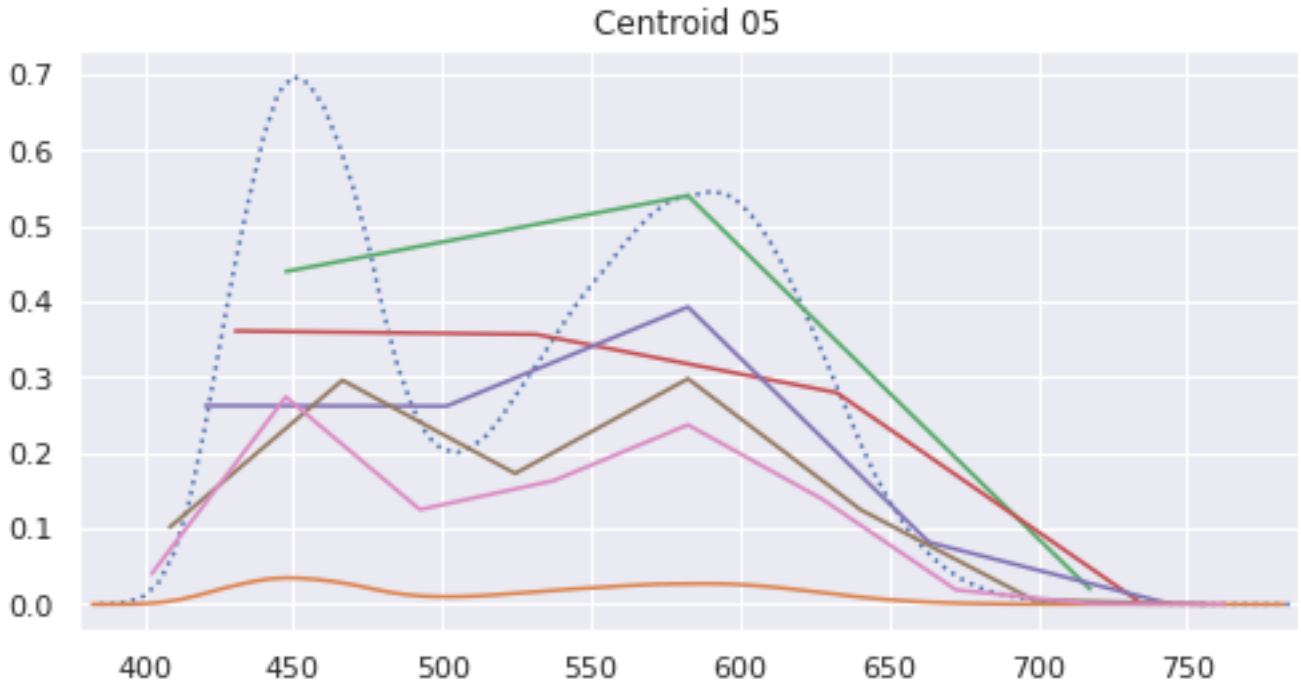
# Reversing color code to spectrum to distribution



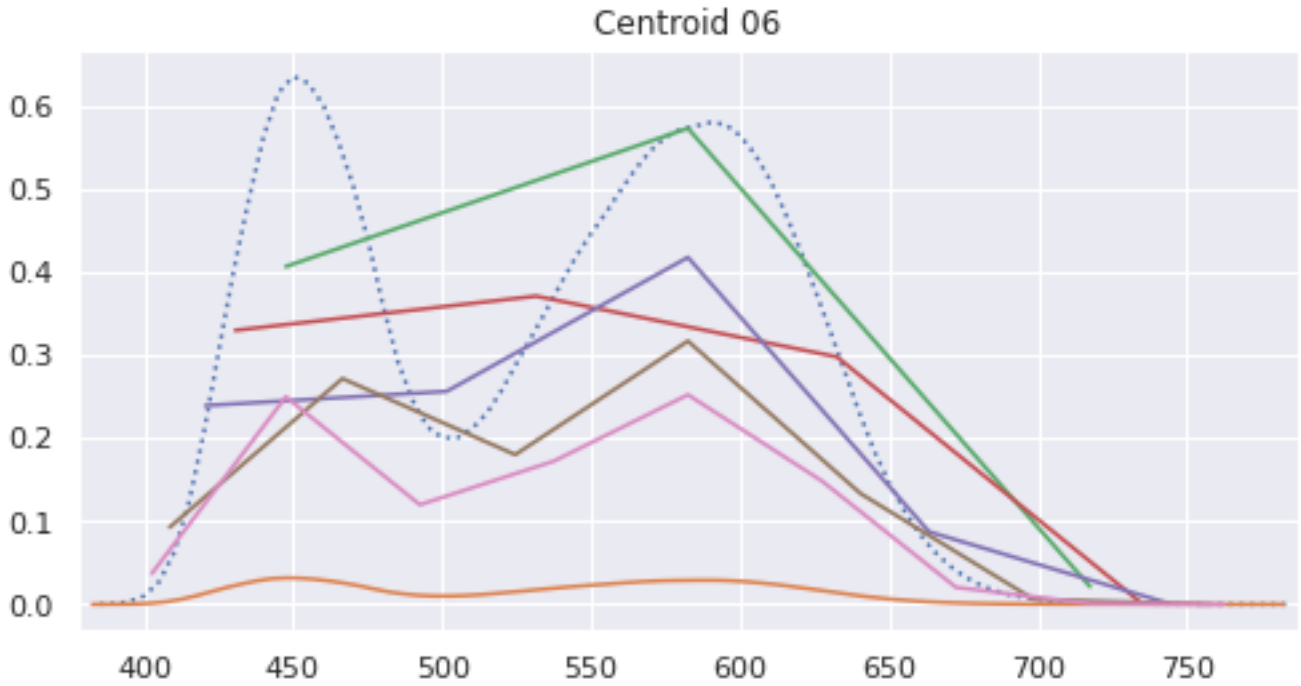
# Reversing color code to spectrum to distribution



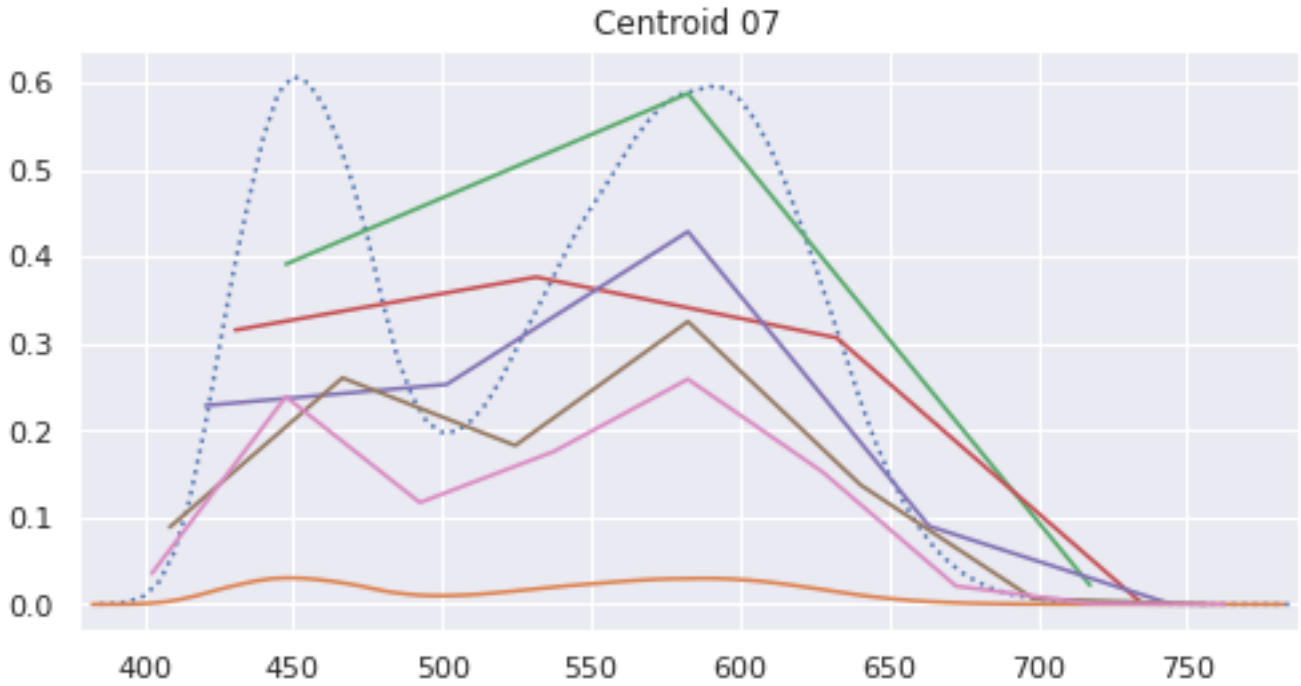
# Reversing color code to spectrum to distribution



# Reversing color code to spectrum to distribution



# Reversing color code to spectrum to distribution



I told you.

That it is possible to make relative distributed, ordered data of differing number of elements comparable.

# Questions?

## References:

All illustrations but CIE standard chromaticity diagram by the author.
CIE standard chromaticity diagram: https://en.wikipedia.org/wiki/File:Cie_Chart_with_sRGB_gamut_by_spigget.png Original image by user Spigget, licensed under Creative Commons Attribution-Share Alike 3.0 Unported.
Christian Hill: Converting a spectrum to a color, 2016, https://scipython.com/blog/converting-a-spectrum-to-a-colour/
Demo notebook, source code, ref materials and sample data set can be found at: github.com/Saravji/cluster_by_color

# This is 42.