assignment_color_for_PDF.R

acate

Sun Nov 6 12:48:56 2016

```
# assignment_color.R
# begun 2016-09-05 (Labor Day) by adc

library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.2.4
```

```
library(colorspace)

allDataManuallyFixed <- read.csv("~/Google Drive/teaching/CogPsych_4114_2016/in-class_assignments/assig

df <- allDataManuallyFixed

attach(df)

# Make copies of the HSV variables that are normalized to range 0,1
# Some color coordinates require values in that range, or at least as they are implemented in R
# Keep the original variables, too.
df$h <- HUE/360;
df$s <- SAT/100;
df$v <- VAL/100;

attach(df)
```

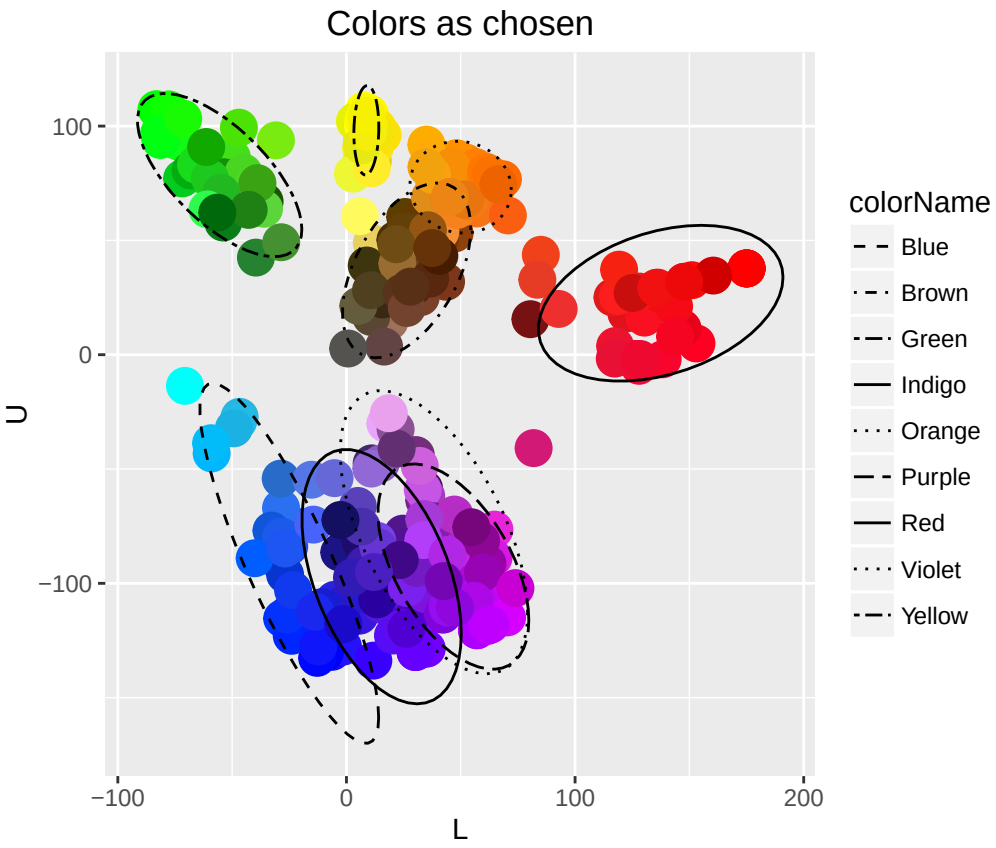
```
## The following objects are masked from df (pos = 3):
##
##     colorName, HUE, id, SAT, VAL
```

```
# Super kludgy code to convert HSV values to L*u*v coordinates:

x <- HSV(cbind(HUE,s,v))

# There seems to be a weird problem with the colorspace package's HSV object class
# Other classes from this package (e.g. RGB ) can be converted to the LUV class,
# but not HSV for some reason.
#
# Consider the following error:
# > as(x,"LUV")
# Error in cbind(L, if (missing(U)) NULL else U, if (missing(V)) NULL else V) :
#   Ambiguous conversion
#
# So Anthony worked around this by converting HSV to RGB (which works) first.

y <- as(as(x,"RGB"),"LUV")

# Another problem: can't directly coerce LUV class to data frame:
```

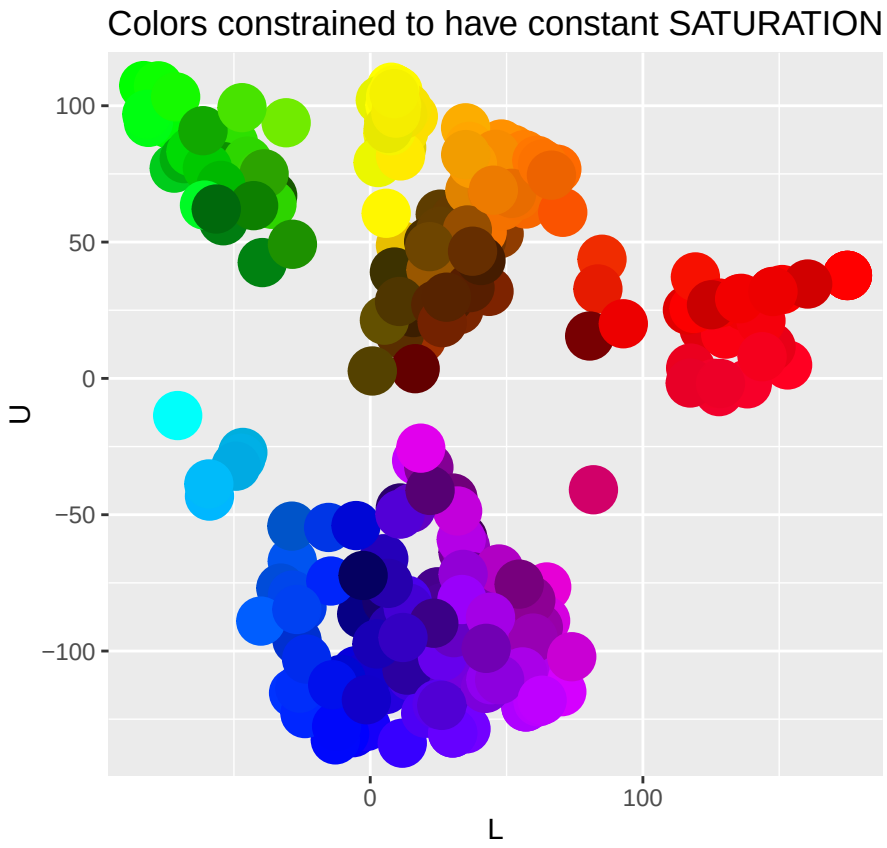
```
# > n <- as.data.frame(y)
# Error in as.data.frame.default(y) :
#   cannot coerce class "structure("LUV", package = "colorspace")" to a data.frame
#
# So, first convert to matrix, then to data frame:

m <- coords(y);
mdf <- as.data.frame(m);

# Calculate means for each category (color name)
meanColors <- rbind(tapply(m[,2],colorName,mean),tapply(m[,3],colorName,mean))
```

```
# Draw the plots
```

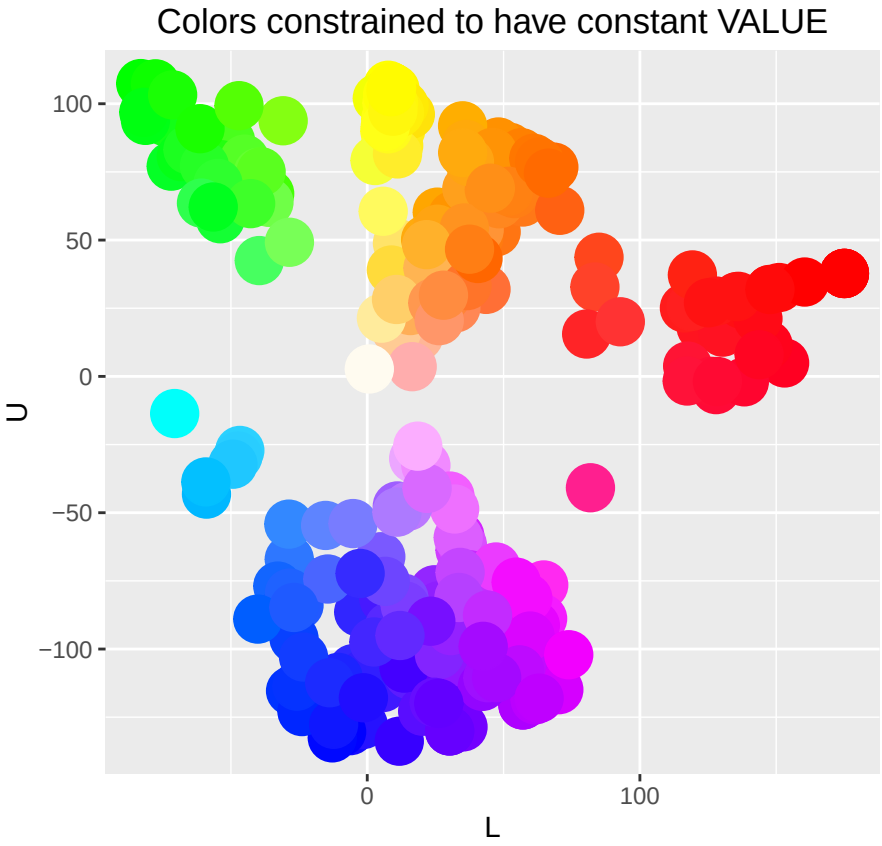
```
# The second and third dimensions of the LUV color space are the ones related to hue and saturation.  T
cp <- ggplot(mdf,aes(mdf[,2],mdf[,3],color=hsv(h,s,v),group=colorName,label=colorName))

# For comparison, make plots where the colors are displayed not as the students chose them, but with a
cpKSat <- ggplot(mdf,aes(mdf[,2],mdf[,3],color=hsv(h,1,v),group=colorName,label=colorName))
cpKVal <- ggplot(mdf,aes(mdf[,2],mdf[,3],color=hsv(h,s,1),group=colorName,label=colorName))


# Plot the colors as they appeared when students chose them:
# cp + geom_point(size=8) + scale_color_identity(guide="none") + coord_fixed() + labs(list(title = "Col

# Version with 95% confidence ellipses
# cp + geom_point(size=8) + scale_color_identity(guide="none") + coord_fixed() + stat_ellipse()


# Plot the colors as they appeared when students chose them:
# With linetype to distinguish ellipses, which also makes a legend
# (AC couldn't figure out how to make line colors different easily)
cp + geom_point(size=6) + scale_color_identity() + coord_fixed() + stat_ellipse(aes(linetype=colorName)
```




```
# Plot the colors as they would appear if they all had the same SATURATION
cpKSat + geom_point(size=8) + scale_color_identity(guide="none") + coord_fixed() + labs(list(title = "C
```




```
# Plot the colors as they would appear if they all had the same VALUE
cpKVal + geom_point(size=8) + scale_color_identity(guide="none") + coord_fixed() + labs(list(title = "C
```



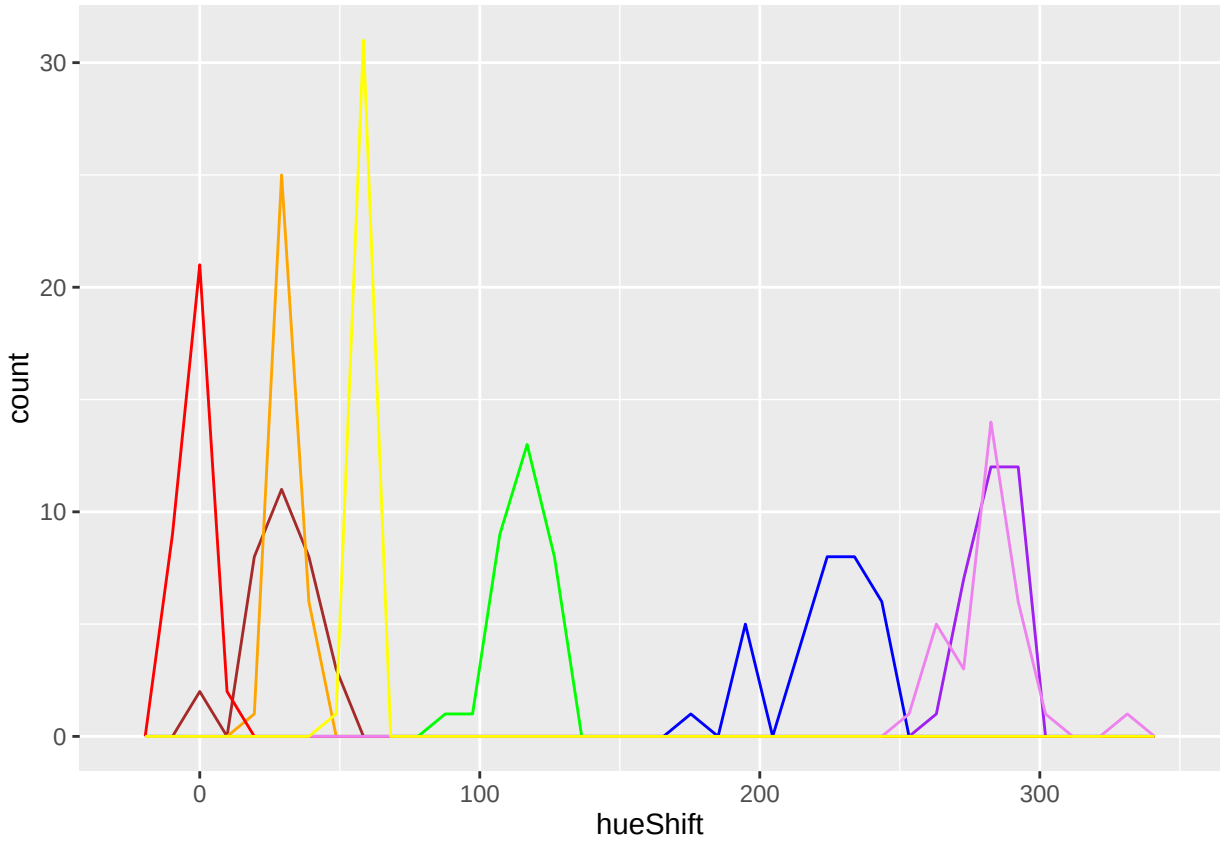

```
# Funny all-text versions:
# cp + scale_color_identity(guide="none") + coord_fixed() + stat_ellipse() + geom_text(fontface="bold")
# cp + scale_color_identity(guide="none") + coord_fixed() + stat_ellipse() + geom_label(fill=hsv(h,s,v)
```

```
# Histograms of HSV variables
```

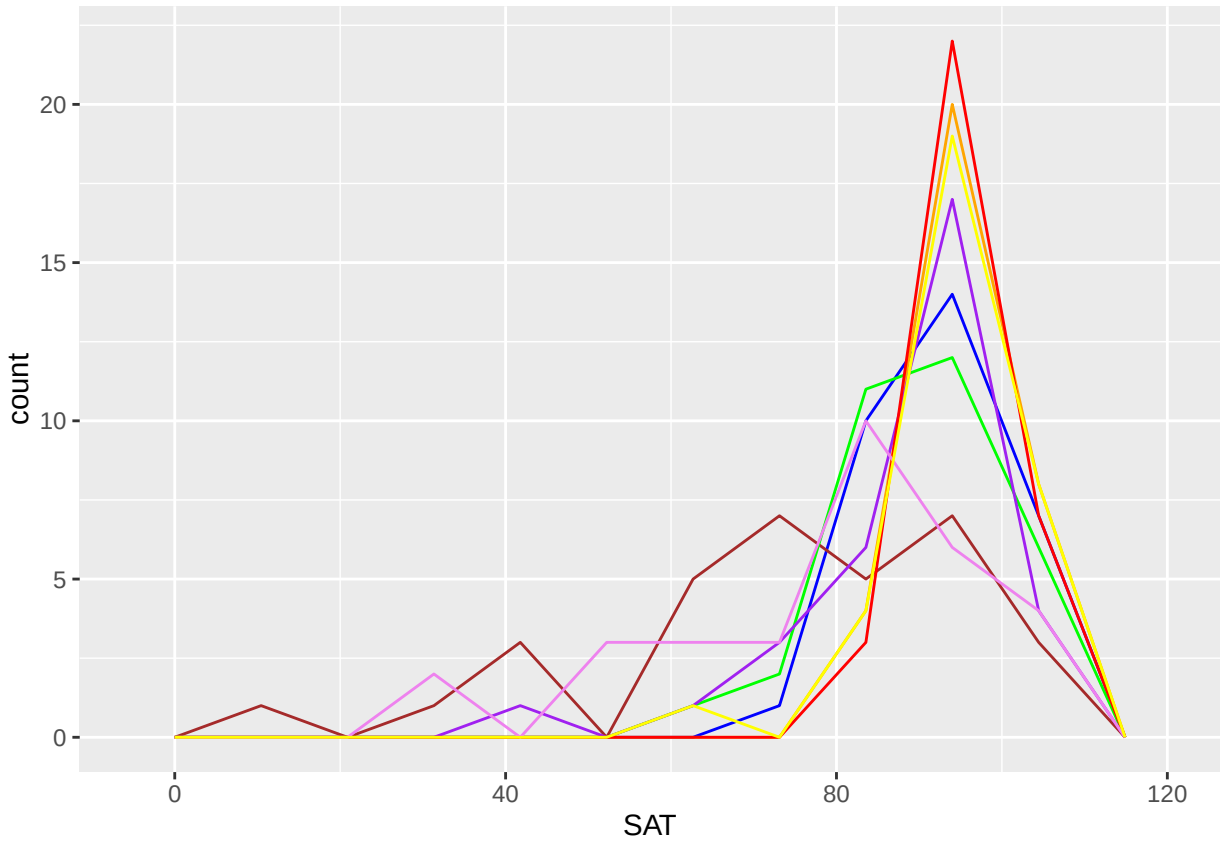
```
hueShift <- HUE;
hueShift[hueShift > 340] <- hueShift[hueShift > 340] - 360;

df$hueShift <- hueShift;

attach(df)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     hueShift

## The following objects are masked from df (pos = 3):
##
##     colorName, h, HUE, id, s, SAT, v, VAL

## The following objects are masked from df (pos = 4):
##
##     colorName, HUE, id, SAT, VAL
```

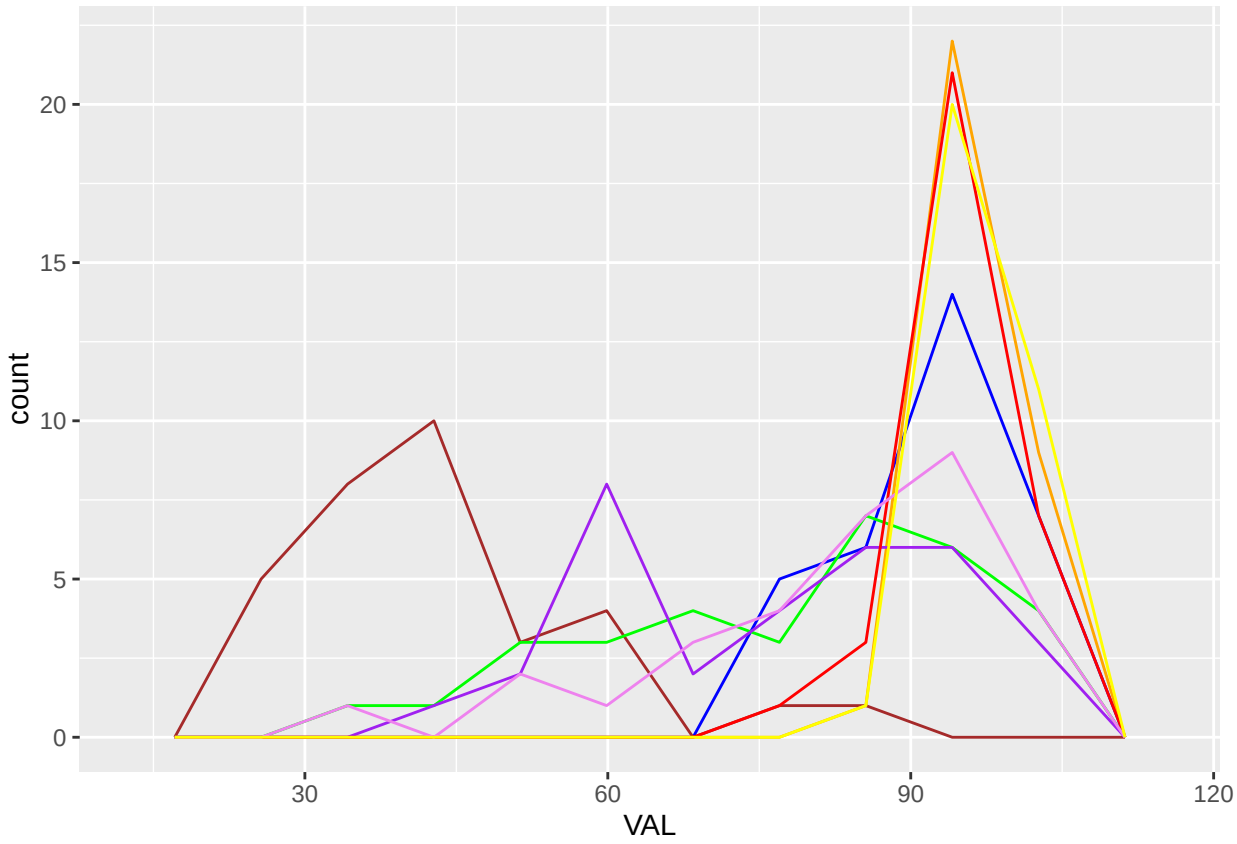
```
# Plot histogram of HUE numbers
ggplot(df[colorName!="Indigo",],aes(hueShift,group=colorName,color=colorName)) + geom_freqpoly(bins=36)
```




```
# Plot histogram of SATURATION numbers
ggplot(df[colorName!="Indigo",],aes(SAT,group=colorName,color=colorName)) + geom_freqpoly(bins=10) + sc
```




```
# Plot histogram of VALUE numbers
ggplot(df[colorName!="Indigo",],aes(VAL,group=colorName,color=colorName)) + geom_freqpoly(bins=10) + sc
```


count
20
15
10
5
0
30
60
90
120
VAL