The Effect of the Aging Lens on Color Vision

Glenn Davis <gdavis@gluonics.com>
November 18, 2018

The goal of this colorSpec vignette is to simulate the effect of age on human color vision. The colored figures are best viewed by an observer with age of 32 years, and on a display calibrated for sRGB.

**The Human Lens**

It is well known that the lens in the human eye does not transmit all wavelengths equally; the transmission at short wavelengths is less, which means the lens is yellowish. In the UV there is very little transmission; which is a good thing since the lens protects the retina from UV damage. It is also well known that the lens gets yellower with age. When making these colored images it is appropriate to use the CIE 1931 color matching functions (CMFs). Unfortunately I could not find the average age of the observers used to establish the 1931 standard observer (there were 17 of them). But it is known that the average age of the observers used to create the CIE 1964 standard observer is 32 years, see [Pok87]. So we’ll take 32 years for the CIE 1931 standard observer as well. Featured functions in this vignette are: linearize(), lensAbsorbance(), RGBfromXYZ(), extradata(), applyspec(), calibrate(), and PlotPatchesRGB().

Start the R session and load the colorSpec package,

```r
library( colorSpec )
```

Compute and plot lens transmittance at 32 and 64 years using the model in [Pok87].

```r
lens.trans = linearize( lensAbsorbance( c(32,64), wave=380:780 ) )
par( omi=c(0,0,0,0), mai=c(0.6,0.7,0.3,0.2) )
plot( lens.trans, color='black', lty=1:2, main=FALSE, legend='topleft' )
```
To compare the color appearance at age 64 to that at age 32, we need the transmittance at age 64 relative to that at age 32. We know that object `lens.trans` is a matrix, so use the standard R matrix subset operation to extract each spectrum. Then perform the division and plot the ratio.

```r
lens.64 = lens.trans[,2] / lens.trans[,1]
lens.64 = colorSpec( lens.64, wavelength(lens.trans), 'transmittance' )
specnames(lens.64) = "trans.64 / trans.32"
par( omi=c(0,0,0,0), mai=c(0.6,0.7,0.3,0.2) )
plot( lens.64, main=TRUE, legend=FALSE, ylab='Relative Transmittance', col='black' )
```

Figure 1: Human Lens Transmittance at age=32 and age=64
Figure 2: Human Lens Transmittance at age 64 relative to age 32

Think of this curve as defining a pair of glasses with a yellowish tint. In this vignette, going from an age of 32 years to 64 years is equivalent to putting on these tinted glasses.

The Macbeth ColorChecker with Observer Age 32

We first read the spectra of the ColorChecker target. This data has been kindly provided in CGATS format by Pas. ColorChecker is a Registered Trademark of X-Rite, and X-Rite is a Trademark.

```r
path = system.file( 'extdata/targets/CC_Avg30_spectrum_CGATS.txt', package='colorSpec' )
MacbethCC = readSpectra( path, wave=wavelength(lens.64) )
MacbethCC = MacbethCC[ order(MacbethCC$SAMPLE_ID), ]
print( extradata(MacbethCC), row.names=F )
```

<table>
<thead>
<tr>
<th>SAMPLE_ID</th>
<th>SAMPLE_NAME</th>
<th>Munsell</th>
<th>ISCC-NBS_Name</th>
<th>LEFT</th>
<th>TOP</th>
<th>WIDTH</th>
<th>HEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>dark skin</td>
<td>3YR 3.7/3.2</td>
<td>moderate brown</td>
<td>7</td>
<td>9</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>2</td>
<td>light skin</td>
<td>2.2YR 6.47/4.1</td>
<td>light reddish brown</td>
<td>40</td>
<td>9</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td>blue sky</td>
<td>4.3PB 4.95/5.5</td>
<td>moderate blue</td>
<td>73</td>
<td>9</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>4</td>
<td>foliage</td>
<td>6.7GY 4.2/4.1</td>
<td>moderate olive green</td>
<td>106</td>
<td>9</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>5</td>
<td>blue flower</td>
<td>9.7PB 5.47/6.7</td>
<td>light violet</td>
<td>139</td>
<td>9</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>6</td>
<td>bluish green</td>
<td>2.5BG 7/6</td>
<td>light bluish green</td>
<td>172</td>
<td>9</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>7</td>
<td>orange</td>
<td>5YR 6/11</td>
<td>strong orange</td>
<td>7</td>
<td>42</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>8</td>
<td>purplish blue</td>
<td>7.5PB 4/10.7</td>
<td>strong purplish blue</td>
<td>40</td>
<td>42</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>9</td>
<td>moderate red</td>
<td>2.5R 5/10</td>
<td>moderate red</td>
<td>73</td>
<td>42</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>10</td>
<td>purple</td>
<td>5P 3/7</td>
<td>deep purple</td>
<td>106</td>
<td>42</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>11</td>
<td>yellow green</td>
<td>5GY 7.1/9.1</td>
<td>strong yellow green</td>
<td>139</td>
<td>42</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>12</td>
<td>orange yellow</td>
<td>10YR 7/10.5</td>
<td>strong orange yellow</td>
<td>172</td>
<td>42</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>13</td>
<td>Blue 7.5PB 2.9/12.7</td>
<td>vivid purplish blue</td>
<td>7</td>
<td>75</td>
<td>29</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Green 0.25G 5.4/8.65</td>
<td>strong yellowish green</td>
<td>40</td>
<td>75</td>
<td>29</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Red 5R 4/12</td>
<td>strong red</td>
<td></td>
<td>73</td>
<td>75</td>
<td>29</td>
<td>29</td>
</tr>
</tbody>
</table>
Note that MacbethCC is organized as 'df.row' and contains extra data for each spectrum, notably the coordinates of the patch rectangle.

Now build the "material responder" from Illuminant D65 and the 1931 CMFs:

```r
D65.eye = product( D65.1nm, "artwork", xyz1931.1nm, wave=wavelength(lens.64) )
# calibrate so the perfect-reflecting-diffuser is the 'official XYZ'
# scale XYZ independently
prd = neutralMaterial( 1, wavelength(lens.64) )
D65.eye = calibrate( D65.eye, stimulus=prd, response=officialXYZ('D65'), method='scaling' )
```

Calculate XYZ and then RGB:

```r
XYZ = product( MacbethCC, D65.eye, wave=wavelength(lens.64) )
RGB = RGBfromXYZ( XYZ, 'sRGB' )  # this is *linear* sRGB
# add the rectangle data to RGB, so they can be plotted in proper places
patches = extradata(MacbethCC)
patches$RGB = RGB
patches.first = patches  # save this reference object for later
# display in proper location, and use the sRGB display transfer function
par( omi=c(0,0,0), mai=c(0.2,0.2,0.2,0.2) )
plotPatchesRGB( patches, gamma='sRGB', back='gray20', labels=FALSE )
```
This figure has the colors as perceived by the 1931 standard observer.

**The Macbeth ColorChecker with Observer Age 64**

Make new responder by inserting the hypothetical pair of tinted glasses (defined by `lens.64` in Figure 2) between target and the eye, and then recalculate RGBs.

```r
D65.eye.64 = applyspec( D65.eye, function(y) {lens.64 * y} )
XYZ = product( MacbethCC, D65.eye.64, wave=wavelength(lens.64) )
patches = extradata(MacbethCC)
patches$RGB = RGBfromXYZ( XYZ, 'sRGB' )  # this is *linear* sRGB
par( omi=c(0,0,0,0), mai=c(0.2,0.2,0.2,0.2) )
plotPatchesRGB( patches, gamma='sRGB', back='gray20', labels=FALSE )
```
As expected, the result has a yellow tint. Now make a plot that compares the effective responsivities.

```r
# the effective responsivities for age=32
par( omi=c(0,0,0,0), mai=c(0.6,0.7,0.3,0.2) )
specnames(D65.eye) = sprintf( "%s.32", c('x','y','z') )
plot(D65.eye, lty=1, legend='top' )
# the effective responsivities for age=64
specnames(D65.eye.64) = sprintf( "%s.64", c('x','y','z') )
plot(D65.eye.64, lty=2, add=TRUE, legend='topright' )
```
But these figures are only appropriate for the instant in time that the change was made, and before the eye and brain have had the time to adapt. In electronic camera terms, there is no "white balance" yet.

So now calibrate and adapt to D65 using the Bradford Method. This method is regarded as being a good model for the way that the human eye and brain achieve color constancy, see [Lin].

```r
prd = neutralMaterial( 1, wavelength(lens.64) )
XYZ.D65 = officialXYZ('D65')
D65.eye.64 = calibrate( D65.eye.64, stimulus=prd, response=XYZ.D65, method='Bradford' )
XYZ = product( MacbethCC, D65.eye.64, wave=wavelength(lens.64) )
patches = extradata(MacbethCC)
patches$RGB = RGBfromXYZ( XYZ, 'sRGB' )  # this is *linear* sRGB
par( omi=c(0,0,0,0), mai=c(0.2,0.2,0.2,0.2) )
plotPatchesRGB( patches, gamma='sRGB', back='gray20', labels=FALSE )
```
Figure 6: Rendering with Illuminant D65 and xyz1931.1nm, at age=64 after chromatic adaption

The tint is now gone. But it hard to compare colors in this figure with the ones way back in Figure 3. So combine the original age=32 rendering with the age=64 rendering by splitting each square into 2 triangles.

```r
par( omi=c(0,0,0,0), mai=c(0.2,0.2,0.2,0.2) )
# draw full squares from Figure 3
plotPatchesRGB( patches.first, gamma='sRGB', back='gray20', labels=F )
# overwrite the squares with triangles by setting shape= and add=
plotPatchesRGB( patches, gamma='sRGB', labels=F, shape='bottomright', add=T )
```
The top-left triangle has the color from Figure 3 and the bottom-right triangle has the color from Figure 6. There are minor differences in the Red and Magenta patches, and some smaller differences in a few others.

Here are the responsivity functions after adaption:

```r
par( omi=c(0,0,0,0), mai=c(0.6,0.7,0.3,0.2) )
plot( D65.eye, lty=1, legend='top', main=FALSE )
plot( D65.eye.64, lty=2, add=TRUE, legend='topright' )
```
So despite the fairly radical yellowing of the lens with age (in Figure 2), this adaption model shows that the perceived colors are not all that different. Great!

**References**


**Appendix**

This document was prepared November 18, 2018 with the following configuration:

- R version 3.5.1 (2018-07-02), i386-w64-mingw32
- Running under: Windows 7 (build 7601) Service Pack 1
- Matrix products: default
• Base packages: base, datasets, grDevices, graphics, methods, stats, utils

• Other packages: colorSpec 0.7-5, knitr 1.20

• Loaded via a namespace (and not attached): MASS 7.3-50, Rcpp 0.12.19, backports 1.1.2, compiler 3.5.1, digest 0.6.17, evaluate 0.11, highr 0.7, htmltools 0.3.6, magrittr 1.5, microbenchmark 1.4-4, minpack.lm 1.2-1, rmarkdown 1.10, rootSolve 1.7, rprojroot 1.3-2, stringi 1.1.7, stringr 1.3.1, tools 3.5.1, yaml 2.2.0