Digital Image Processing Laboratory: Introduction to Colorimetry

March 2, 2021

1 Introduction

In order to understand natural images, one needs to understand how objects reflect light and how humans perceive that reflected light as color. The objective of this laboratory is to introduce the basic ideas behind colorimetry, the quantitative measurement and manipulation of color. After completing this laboratory, you should have a better understanding of color principles and how to accurately produce a color image on a calibrated monitor.

There are three major components to understanding color:

- Understanding how light is reflected from a natural scene.
- Understanding how humans perceive reflected light.
- Understanding how a rendering device, such as a monitor, functions.

The following sections will introduce some theoretical background for each of these components, and this will be followed by several related exercises.

In Section 3, you will reproduce a chromaticity diagram from color matching functions, including two standard sets of RGB primaries and two white points. In Section 4, you will reproduce an image given a spatial array of reflectance coefficients, the radiant energy from an illuminant, and the color matching functions. Finally in Section 5, you will create a color-filled chromaticity diagram using the set of Rec. 709 RGB primaries. Each exercise will be performed in Python.

1.1 Reflection of Light in a Natural Scene

It is important to note that the color that one sees is a function of both the reflectance of an object's surface, and the spectral distribution of the light source, or *illuminant.* More specifically, let $S(\lambda)$ be the spectral energy density of an illuminant as a function of the spectral wavelength λ, and let $R(\lambda)$ be the fraction of the illuminant energy that is reflected toward the viewer. The spectral distribution of light that the viewer sees is given by

$$I(\lambda) = R(\lambda)S(\lambda) \ . \tag{1}$$

The viewer's perception of color depends on the distribution of the reflected light $I(\lambda)$. While the spectral distribution of light energy in a natural scene can be very complex, human

Questions or comments concerning this laboratory should be directed to Prof. Charles A. Bouman, School of Electrical and Computer Engineering, Purdue University, West Lafayette IN 47907; (765) 494-0340; bouman@ecn.purdue.edu

beings are only sensitive to a limited range of spectral wavelengths. We will be principally concerned with wavelengths between 400 and 700 nanometers (nm) since this is the range of wavelengths that are most visible. Wavelengths longer then 700nm are known as infrared, and those shorter then 400nm are known as ultraviolet.

Importantly, $I(\lambda)$ is a function of both the object being viewed and the illuminant $S(\lambda)$. In practice this means that the perceived color of an object can depend on the spectral distribution of the light illuminating it. Real illuminants can vary widely in their spectral distribution, but there are a few standard models that are commonly used. One simple model is the equal energy white illuminant given by

$$S_{EE}(\lambda) = 1 \ .$$

This illuminant has equal spectral energy at each wavelength. In practice few illuminants have such flat spectral distributions. A more common model is the standard D_{65} illuminant, which approximates the distribution of noon sunlight. The distribution is given by a tabulated set of measurements which we will use in this laboratory. In our exercises, all spectral distributions will be given at 31 discrete wavelengths uniformly spaced between 400 and 700 nanometers (nm), i.e. $\lambda = 400nm, 410nm, \ldots, 700nm$.

1.2 Human Perception of Color

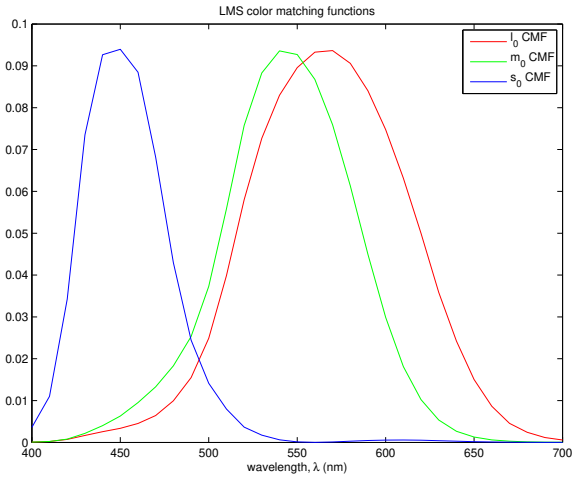

Figure 1: LMS Color Matching Functions

Humans have three distinct types of color receptors, or *cones*, in their retinas. These are referred to as *long*, *medium*, and *short* cones because each type has a particular sensitivity to long, medium and short wavelengths of light. The relative response of each cone as a function of light wavelength, λ, has been measured experimentally, and is represented by the three functions $l_0(\lambda)$, $m_0(\lambda)$, and $s_0(\lambda)$, illustrated in Figure 1. These are referred to as the cones' *color matching functions.* The total response of each cone can then be determined by

$$L \ = \ \int_{\lambda} I(\lambda) l_0(\lambda) d\lambda$$

$$\begin{aligned} M &= \int_\lambda I(\lambda) m_0(\lambda) d\lambda \\ S &= \int_\lambda I(\lambda) s_0(\lambda) d\lambda \ . \end{aligned} \tag{2}$$

This implies that, while the spectrum of incoming light may be very complex, the response of the visual system is limited to three dimensions. This three dimensional representation of color is known as the *tristimulus* model of color vision and is widely accepted and known to be quite accurate.

In practice, the above integrals are approximated by summing over a finite number of wavelengths. Typically, the spectral energy is measured at 31 discrete wavelengths uniformly spaced between 400 and 700 nanometers (nm), $\lambda = 400nm, 410nm, \ldots, 700nm$. The tristimulus values of (2) are then approximated by three vector inner products.

$$\begin{aligned} L &= l_0 I \\ M &= m_0 I \\ S &= s_0 I \end{aligned} \tag{3}$$

where l_0, m_0, and s_0 are 1×31 row vectors containing properly normalized samples of the color matching functions, and I is a 31×1 column vector representing the spectral energy of the incoming light. This relationship is more compactly expressed in matrix form.

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} l_0 \\ m_0 \\ s_0 \end{bmatrix} I$$

1.3 The XYZ Color Space and Chromaticity

In fact, the LMS system is not commonly used for color specification. The most standard representation of tristimulus values is the CIE 1931 standard XYZ color coordinate system. The coordinates in the XYZ system are related to LMS through the following transformation,

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = A \begin{bmatrix} L \\ M \\ S \end{bmatrix}$$

where the inverse of A is approximately given by,

$$A^{-1} = \begin{bmatrix} 0.2430 & 0.8560 & -0.0440 \\ -0.3910 & 1.1650 & 0.0870 \\ 0.0100 & -0.0080 & 0.5630 \end{bmatrix} .$$

Color matching functions $x_0(\lambda)$, $y_0(\lambda)$, and $z_0(\lambda)$ for the XYZ color system can then be derived from $l_0(\lambda)$, $m_0(\lambda)$, and $s_0(\lambda)$ using the same transformation.

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = A \begin{bmatrix} l_0 \\ m_0 \\ s_0 \end{bmatrix}$$

Now given a radiant light distribution, $I(\lambda)$, the color in the XYZ coordinate system can be computed using these new color matching functions.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} I$$

The XYZ color matching functions x_0, y_0, and z_0 are chosen to have some important special properties. First, they are positive for all wavelengths, λ. Second, for an equal energy spectral distribution with $I_{EE} = [1, 1, \cdots, 1]^t$, the XYZ coordinates have unit values.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} I_{EE} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Another important property is that the Y component is closely associated with the perceived "brightness" of the color. Y is referred to as the *luminance*.

A set of tristimulus values (X, Y, Z) can be thought of as a vector in a 3-D color space. The color of a point in the space is then dependent on both the direction and magnitude of the vector. Intuitively, changes in the vector's magnitude will change the brightness of the color while changes in the vector's direction will change the color's hue and saturation. These latter properties are generally referred to as a color's *chromaticity*. The chromaticity, (x, y, z), of a color is defined as the coordinates (X, Y, Z) scaled by their sum.

$$\begin{aligned} x &= \frac{X}{X+Y+Z} \\ y &= \frac{Y}{X+Y+Z} \\ z &= \frac{Z}{X+Y+Z} \end{aligned} \tag{4}$$

Geometrically, the chromaticity of a color is the intersection of the vector (X, Y, Z) with the unit plane $X + Y + Z = 1$. Notice from (4) that the chromaticities are independent of the length of the (X, Y, Z) vector, and that $x + y + z = 1$. Often, only (x, y) is specified for a color's chromaticity since z may be computed by $z = 1 - x - y$.

1.4 Color Rendering on a Monitor

The XYZ tristimulus values specify precisely what a color should look like. A significant challenge is then how to accurately produce those colors on a display or other rendering device. In this section, we will cover the transformation from XYZ to the color coordinates of a typical electronic display.

1.4.1 Color Primary Transformations

Most display devices produce colors by linearly adding three primary colors such as red, green, and blue. These three primary colors may be represented as basis vectors $\overrightarrow{R}$, $\overrightarrow{G}$, and

$\overrightarrow{B}$ in the XYZ space,

$$\overrightarrow{R} = \begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix} \quad \overrightarrow{G} = \begin{bmatrix} X_g \\ Y_g \\ Z_g \end{bmatrix} \quad \overrightarrow{B} = \begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix}$$

where each vector contains the measured XYZ values for the corresponding primary.

Using this additive convention, a color (X, Y, Z) can be produced by adding scaled amounts of the RGB primaries,

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = r\,\overrightarrow{R} + g\,\overrightarrow{G} + b\,\overrightarrow{B} = \begin{bmatrix} \overrightarrow{R} & \overrightarrow{G} & \overrightarrow{B} \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix} \tag{5}$$

From this, we see that (X, Y, Z) is determined from the (r, g, b) values by the transformation,

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M \begin{bmatrix} r \\ g \\ b \end{bmatrix} \tag{6}$$

where

$$M = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix} . \tag{7}$$

Similarly, the required (r, g, b) coefficients to represent a particular color (X, Y, Z) is determined by,

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = M^{-1} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} . \tag{8}$$

Note that for almost all RGB primary systems, there are some visible colors (X, Y, Z) that result in a negative component among (r, g, b). Since a monitor can only produce positive quantities of each primary, an important implication is that any such display device will be unable to produce all visible colors.

1.4.2 Chromaticity and White Point for RGB Primaries

It is common to specify a set of color primaries $\overrightarrow{R}$, $\overrightarrow{G}$, and $\overrightarrow{B}$ indirectly through their chromaticities and a *white point*. Note that the chromaticities alone are not enough to specify a given point in XYZ since chromaticities are always scaled such that $x + y + z = 1$. In effect they only specify the "direction" of the color vector. The white point specification fills in the remaining information.

Generally, a white point specifies the set of chromaticities (x, y, z) associated with a "white light" spectrum. For example, the white point associated with a D65 illuminant is

$$(x_{wp}, y_{wp}, z_{wp})_{D65} = (0.3127, 0.3290, 0.3583) \tag{9}$$

and an *equal energy* white point is given by

$$(x_{wp}, y_{wp}, z_{wp})_{EE} = (0.3333, 0.3333, 0.3333) \ . \tag{10}$$

From the perspective of a display device, the white point specifies the color (chromaticity) produced when adding maximal amounts of the three primaries, i.e. when $(r, g, b) = (1, 1, 1)$, or $(r, g, b) = (255, 255, 255)$ if using a single-byte encoding. (Note this definition of white point differs slightly from the one given above, but unfortunately the same term is used for both concepts.)

Now given the chromaticities for a set of primaries,

$$\begin{aligned} \overrightarrow{R} &= (x_r, y_r, z_r) \\ \overrightarrow{G} &= (x_g, y_g, z_g) \\ \overrightarrow{B} &= (x_b, y_b, z_b) \end{aligned} \tag{11}$$

the XYZ tristimulus values for the primaries will be related to the chromaticities through a set of scaling constants κ_r, κ_g, and κ_b.

$$(X_r, Y_r, Z_r) = \kappa_r(x_r, y_r, z_r) \tag{12}$$
$$(X_g, Y_g, Z_g) = \kappa_g(x_g, y_g, z_g) \tag{13}$$
$$(X_b, Y_b, Z_b) = \kappa_b(x_b, y_b, z_b) \tag{14}$$

In matrix form, this conversion is represented as,

$$M = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix} = \begin{bmatrix} x_r & x_g & x_b \\ y_r & y_g & y_b \\ z_r & z_g & z_b \end{bmatrix} \begin{bmatrix} \kappa_r & 0 & 0 \\ 0 & \kappa_g & 0 \\ 0 & 0 & \kappa_b \end{bmatrix} \ . \tag{15}$$

Note the above matrix, M, is the transformation from (r, g, b) to (X, Y, Z) as specified in Equation (6).

Our problem now is to find the set of scaling constants κ_r, κ_g, and κ_b which will determine the true primary colors, given the white point specification. From our previous definition, the white point specifies the color (X_{wp}, Y_{wp}, Z_{wp}) produced when $(r, g, b) = (1, 1, 1)$. So from Equation (6),

$$\begin{bmatrix} X_{wp} \\ Y_{wp} \\ Z_{wp} \end{bmatrix} = \begin{bmatrix} x_r & x_g & x_b \\ y_r & y_g & y_b \\ z_r & z_g & z_b \end{bmatrix} \begin{bmatrix} \kappa_r & 0 & 0 \\ 0 & \kappa_g & 0 \\ 0 & 0 & \kappa_b \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{16}$$

Since the white point is usually specified by chromaticity, (x_{wp}, y_{wp}, z_{wp}), we need to scale appropriately to get (X_{wp}, Y_{wp}, Z_{wp}). Since the Y component of the color (X, Y, Z) is closely associated with the perceived "brightness", we will assume the brightest output of the display device, $(r, g, b) = (1, 1, 1)$, produces $Y = 1$. The XYZ coordinates of the white point are then given by,

$$\begin{bmatrix} X_{wp} \\ Y_{wp} \\ Z_{wp} \end{bmatrix} = \begin{bmatrix} x_{wp}/y_{wp} \\ 1 \\ z_{wp}/y_{wp} \end{bmatrix} \ .$$

and finally the scaling coefficients,

$$\begin{bmatrix} \kappa_r \\ \kappa_g \\ \kappa_b \end{bmatrix} = \begin{bmatrix} x_r & x_g & x_b \\ y_r & y_g & y_b \\ z_r & z_g & z_b \end{bmatrix}^{-1} \begin{bmatrix} x_{wp}/y_{wp} \\ 1 \\ z_{wp}/y_{wp} \end{bmatrix} . \tag{17}$$

1.4.3 Standard RGB Primaries

One important set of RGB primaries is specified in the CIE 1931 color space standard. The chromaticities are given by,

$$\begin{aligned} \overrightarrow{R}_{CIE\_1931} &= (x_r, y_r, z_r) = (0.73467, 0.26533, 0.0) \\ \overrightarrow{G}_{CIE\_1931} &= (x_g, y_g, z_g) = (0.27376, 0.71741, 0.00883) \\ \overrightarrow{B}_{CIE\_1931} &= (x_b, y_b, z_b) = (0.16658, 0.00886, 0.82456) \end{aligned} \tag{18}$$

These primaries are very useful for laboratory measurements, but are not practical for use in devices such as monitors. A more common set of primaries for display devices are the recommended standard Rec. 709 RGB primaries.

$$\begin{aligned} \overrightarrow{R}_{709} &= (x_r, y_r, z_r) = (0.640, 0.330, 0.030) \\ \overrightarrow{G}_{709} &= (x_g, y_g, z_g) = (0.300, 0.600, 0.100) \\ \overrightarrow{B}_{709} &= (x_b, y_b, z_b) = (0.150, 0.060, 0.790) \end{aligned} \tag{19}$$

2 Plotting Color Matching Functions and Illuminants

In this section, you will plot the color matching functions and illuminants used in the laboratory. The relevant data is contained in the file *data.npy*, which is available at the lab web site. It contains the XYZ color matching functions at the discrete set of wavelengths [400:10:700] nanometers. It also includes two vectors *illum1* and *illum2* containing radiant light energy from two illuminant sources. The *illum1* vector is a D65 source (noon daylight), and *illum2* is from a fluorescent light.

1. Load the file *data.npy* into Python.

```
import numpy as np
# Load data.npy
data = np.load('data.npy', allow_pickle=True)[()]
# List keys of dataset
data.keys()
```

2. Plot the three $x_0(\lambda)$, $y_0(\lambda)$, and $z_0(\lambda)$ color matching functions versus wavelength, λ. Plot all three on the same axes, and use the [matplotlib.pyplot.legend] function to identify the graphs.

3. Compute the three $l_0(\lambda)$, $m_0(\lambda)$, and $s_0(\lambda)$ color matching functions corresponding to the long medium and short cones. You will need to use the transformation given in Section 1.3. Plot these three functions versus wavelength, λ.

4. Plot the spectrum of the D_{65} and fluorescent illuminants versus wavelength, λ.

Section 2 Report:
Hand in:

- The plot of the $x_0(\lambda)$, $y_0(\lambda)$, and $z_0(\lambda)$ color matching functions.
- The plot of the $l_0(\lambda)$, $m_0(\lambda)$, and $s_0(\lambda)$ color matching functions.
- The plot of the D_{65} and fluorescent illuminants.

3 Chromaticity Diagrams

A chromaticity diagram is a graphical representation of colors according to their position in (x, y) chromaticity coordinates. Chromaticity coordinates have an important property that combinations of any two colors always fall along a straight line between the two points. This property will be useful in visualizing the structure of a color space.

The most saturated colors are produced from spectral distributions that have energy only at a single wavelength. Such a pure spectrum can only be generated by using a very narrow optical filter, or from a narrow band source such as a laser. The XYZ coordinates of a pure spectral source at wavelength λ_0 is given by

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \int_\lambda \delta(\lambda - \lambda_0) x_0(\lambda) d\lambda \\ \int_\lambda \delta(\lambda - \lambda_0) y_0(\lambda) d\lambda \\ \int_\lambda \delta(\lambda - \lambda_0) z_0(\lambda) d\lambda \end{bmatrix} = \begin{bmatrix} x_0(\lambda_0) \\ y_0(\lambda_0) \\ z_0(\lambda_0) \end{bmatrix} . \tag{20}$$

Since the chromaticities of these pure spectral colors are highly saturated, they represent a boundary for all visible colors. A plot of these (x, y) chromaticities as a function of λ is called a *chromaticity diagram.*

1. Load the data file `data.npy` into the Python.

2. Using equations (20) and (4), plot the chromaticities (x, y) of a pure spectral source as a parametric function of λ. Use a solid line type.

3. In your diagram, also plot the three CIE 1931 standard RGB primaries using the chromaticities given in equation (18). Connect the three points with straight lines to illustrate the range of colors that can be generated by these three primaries. Label each of the three points using the [matplotlib.pyplot.text].

4. Repeat the previous step for the Rec. 709 RGB primaries given in equation (19).
5. Plot and label a point on your diagram for the chromaticity of a D_{65} white point, given in equation (9).
6. Repeat the previous step for an equal energy white point, given in equation (10).
7. Print or export your final diagram.

Section 3 Report:
Hand in your labeled chromaticity diagram.

4 Rendering an Image from Illuminant, Reflectance, and Color Matching Functions

The objective of this section will be to display a calibrated color image from a known illuminant spectrum and the reflectance coefficients at each point in the image. You will use the XYZ color matching functions and illuminant data from the previous section, as well as an $m \times n \times 31$ array, `R`, that contains reflectance coefficients at 31 wavelengths for each point in an $m \times n$ image. The indexing of this array has the order *R(row,column,wavelength)*. The reflectance data can be found in the file *reflect.mat*, which is available from the lab web site. Note this file is approximately 11 Mbytes.

1. Load *data.npy* and *reflect.npy* into your Python workspace.
2. Using the D_{65} light source (*illum1*) and R, compute the reflected light energy at each given wavelength, λ, using equation (1). Do this for each pixel in the image, producing an $m \times n \times 31$ array, called I.
3. Compute the XYZ tristimulus values for each pixel by applying the color matching functions to the spectral energy in I. Call this $m \times n \times 3$ array `XYZ`.
4. Assuming your computer monitor uses the Rec. 709 RGB primaries, and has a D_{65} white point, compute the transformation matrix M_{709_D65} that will transform from XYZ to this RGB color space. Print out this matrix.
5. Use M_{709_D65} to transform each pixel in your `XYZ` array into RGB coordinates.
6. Some RGB component values may fall outside the range [0,1], which are colors your monitor will be unable to display. To fix this, appropriately clip any components outside this range to 0 or 1.
7. Now γ-correct your image so that it will be displayed properly on your monitor. Assume here that your monitor has a γ of 2.2.

8. Display the result using the *Image* from pillow, and print or export the result for you report. In this case you will first need to scale the pixel values by 255, and convert the array to type *uint8*.

```
import numpy as np
from PIL import Image
im = Image.open('img.tif')
x = np.array(im)
# Some calculation
...
im = Image.fromarray(x)
im.save(fname,'tiff')
```

9. Repeat the entire exercise using the fluorescent source in `illum2`. Note that the only change here will be the light source–you should use the same M_{709_D65} transformation as before for the XYZ to RGB conversion.

10. Compare the two results.

Section 4 Report:
Hand in the following:

1. The matrix M_{709_D65}.
2. The two images obtained from D_{65} and fluorescent light sources.
3. A qualitative description of the differences between the two images.

5 Color Chromaticity Diagram

In this exercise you will create a chromaticity diagram similar to Section 3, but that will also display a range of colors available from your monitor. Recall the chromaticity (x, y, z) of a color (X, Y, Z) is the intersection of the color vector with the plane $X + Y + Z = 1$. Here, you will display the colors for all the points within the plane $X + Y + Z = 1$ that your monitor is capable of reproducing.

1. Use `meshgrid` to create two coordinate matrices, x and y, of chromaticity values ranging from 0 to 1, with a spacing of 0.005. Then compute the corresponding z matrix. For each location (i, j), these matrices give a particular set of chromaticity coordinates (x_{ij}, y_{ij}, z_{ij}), where x increases with j (column), and y increases with i (row).

2. For each location (i, j) in the chromaticity matrices, compute the corresponding display (r, g, b) values for the color $(X, Y, Z) = (x_{ij}, y_{ij}, z_{ij})$. Here, assume your monitor produces the Rec. 709 RGB primaries, and the primaries fall right on the plane

$X + Y + Z = 1$. In other words, the transformation matrix, M_{709}, is given exactly by equations (19) and (15), with $\kappa_r = \kappa_g = \kappa_b = 1$. Place the three computed RGB values in a 3-D color image array.

3. The locations where r, g, **or** b are negative represent colors that cannot be reproduced by this set of primaries. In each location where the r, g, or b component is negative, set *all three components* to 1 (white).

4. Gamma-correct each color plane, assuming a γ value of 2.2.

5. Use the Python command `matplotlib.pyplot.imshow` to display your color diagram. Label and scale the x and y axes appropriately. Hint: The scaling can be achieved easily with a command similar to `imshow(img,extent=[0,1,0,1])`.

6. Insert the same chromaticity plot of a pure spectral source that you produced in Section 3. Note that you **must have** the correct scaling of the x and y axes in the previous step for this plot to align properly.

Your final result should be a color-filled triangle contained inside the pure spectral source outline, and the vertices of the triangle should be the chromaticities of the Rec. 709 RGB primaries.

Section 5 Report:
Hand in your color diagram.