

A short course in color, and some other aspects of our image analysis

Color vision by tricolor stimulus

The cones of the human eye have three distinct pigments (in different cones), with peak sensitivities in the red, yellow, and blue. Light reaching our eye from any scene is a complicated mixture of light of all colors, from red to blue, at all wavelengths inbetween. However, to reproduce the sensation of light of any color, we need only mix three primary colors in the proper amounts: red, green, and blue. These are the primary additive colors; mixing *light* of these colors reproduces any color. Primary additive colors are used in light-emitting devices, such as color monitors. Any complicated mixture of light at different wavelengths only represents varied degrees of stimuli of our three types of cones, so mixing only three primary colors suffices to reproduce any color. There are a few tricky colors, given that our cones each respond over extended wavelength ranges. See, for example, Lloyd Kaufman's book *Sight and Mind* for some interesting details. In any event, the average human can discriminate about 20,000 different colors in matching tests. Note that various color schemes claim up to 16.7 million colors (24-bit digital color). This high resolution is only noticeable in certain wavelength ranges. For example, we are relatively insensitive to small changes in amounts of red, but we are sensitive to small changes in amounts of blue. Color monitors resolve fine changes in all three primary colors; it doesn't cost much more, and it makes life simpler.

In producing a surface, such as a photographic print, that reproduces any given color, we can't add color, since dyes or inks don't *produce* light (fluorescent dyes are a partial exception). Rather, colored pigments (dyes and inks) only subtract colors from incident light. For example, yellow pigments subtract blue light, leaving green and red light. As with adding light, in adding pigments we need only three primary colors: cyan (also called "minus red"), magenta ("minus green") and yellow ("minus blue"). These are the primary subtractive colors.

Thus, we have the two common schemes for color rendition: RGB = red, green, blue, for additive colors, and CMY or CMYK for subtractive colors. The RGB scheme is used in color monitors and the like; the CMYK scheme is used in printing, including offset presses and our own computer's printer. The extra color, K, is simply black. The pigments used for cyan, magenta, and yellow don't cover the spectrum perfectly, so that mixing all three produces a muddy brown rather than a true black. (Pigments that are perfect subtractive primaries lack other key properties, such as long-term stability to light, ability to wet paper properly, etc.) Thus, printers also add a small amount of black ink as needed to make true blacks. Computing the exact amount to add that gives such blacks without muddying the non-black parts of the scene is both tricky and inexact; you'll notice that our prints from on-screen images can often be a bit muddy. Such is life.

The quantitative description of color

First, let's briefly note the qualitative description of color. You may know of color wheels and color chips (Munsell chips, for example). For any given color (any part of a scene), we can match it to a Munsell chip. The chips can be arranged in a semi-quantitative way in three dimensions, with general degree of "lightness" on a vertical axis, the different "hues" (redness blending into yellowness into greenness...) distributed at various angles of rotation around this axis, and

the degree of "saturation" (how much white is blended with a pure color) represented by the radial distance from the axis. This presentation summarizes a great deal about how we perceive color, e.g., that the spectrum "wraps around," making extreme blues or indigos appear close to deep reds at the other extreme of wavelength. However, this scheme is not quantitative. It doesn't let us describe or predict how a change in the color balance of light illuminating a print will change the final color we perceive (as when we illuminate a page with incandescent light rather than sunlight). We turn, then, to the RGB or CMY schemes, in which the quantities of each primary color are specified.

In specifying color digitally with the RGB scheme, each component indicates how much of that color is present, *relative to* the maximum intensity of that color that the device (color monitor) can put out. An outdoor scene might have flux densities of 150, 130, and 200 micromoles per square meter per second in the R, G, and B wavelength ranges, respectively. Suppose there's a maximum in R or G or B, say, $800 \mu\text{mol m}^{-2}\text{s}^{-1}$. The sensor in the camera will have a certain voltage or current output at this light flux density, and we will make the digitizer convert this to the number 255 (digital values will run from 0 to 255, not 1 to 256). Another camera, designed for low light use indoors, might "max out" at $150 \mu\text{mol m}^{-2}\text{s}^{-1}$; we will use this real flux density as equivalent to 255 as a digital value.

Equal amounts of R, G, and B do not correspond to equal photon flux densities in the R, G, and B wavelength bands. Our eyes are more sensitive to green than to red or blue. Our peak sensitivity is in the yellow, in fact. Photon fluxes in the R, G, and B bands are converted to digital R, G, and B values by multiplying with the relative sensitivity of the eye to each color. Thus, it may take 2.4 times the flux density in the blue to achieve the same digital value as in the green.

There is clearly a direct correspondence from the RGB mixture used on a computer screen to the CMY(K) values used to direct the printer to reproduce the scene. If the CMY colors were perfect complementary colors to RGB, then R, G, and B all at their maximum corresponds to white, and *no* pigments should be put on the page (C, M, and Y are all 0). Suppose, as we will use later, R, G, and B each range from 0 to 255. For a pure green, $R = 0$ and $B = 0$; perhaps $G = 120$ (it's a dark green). This means that all the red and blue should be taken out, and all but 120 of the green. This might be achieved with $C = 255$, $M = 135$, and $K = 255$.

Hue, saturation, contrast, and all that

Specifying the amounts of R, G, and B light in a single element of a picture ("pixel" = *picture element*) is fine if we want to tell a color monitor how much power to apply to each electron gun to make a given color. How does the RGB scheme relate to our cogent ideas of hue, saturation, etc., which describe how we react to a scene, how we discriminate features, etc.? Let's define some terms:

Description of a single pixel: hue, saturation, and value.

Hue is the dominant color, equivalent to which wavelength dominates. A green hue is one in which green light is the weighted average color. We might relate hue to wavelength, and say that green is equivalent to 520 nm. A yellow hue is one in which green and red are mixed, or the average wavelength is about 550 nm. We can resolve hue fairly finely, though not evenly across the spectrum. Let's credit ourselves with being able to distinguish a few hundred hues. Going to the quantitative RGB scheme, we might say hue is something like mean wavelength, so we could compute it as, say, $450*B + 520*G + 650*R$, where R, G,

and B are each on a scale from 0 to 1 (max). In practice, the location of the mean wavelength is mapped onto a circle, from 0 to 360 (degrees). A pure red is around 0, greens are around 150, blues are around 250...and reds with some blue are mapped to over 300.

Saturation is the amount of dilution of hue by overall whiteness. Consider a scene in 24-bit color that has RGB=(70,120,70). The dominant color is green, so the hue is green. This color is a mixture of pure green at a modest intensity (G=50) with white at a higher intensity (R=G=B=70). Note that white is equal amounts of R, G, and B, at any absolute amount; different amounts correspond to different brightnesses, or graynesses. We could then compute saturation as (total R+G+B after subtracting the white part)/(total R+G+B). Actually, saturation is calculated as $100 * [(maximum\ component\ among\ R,G,B) - (minimum\ component)] / [maximum\ component]$. In the example just given, saturation equals $100 * (120 - 70) / 120 = 41$, after rounding down. This is a modestly washed-out or pastel green.

Value is the overall brightness. One measure is the sum of R + G + B, but your eye does not perceive a blue-green (R,G,B = 100,100,0) as significantly brighter than a simple green (0,100,0). Thus, value is defined as the maximum component among R, G, and B, divided by 2.55 so that it scales from 0 to 100. For example, a pixel with RGB = (34,68,57) has $V = (68/255) * 100 = 27$ (always round down).

Thus, we have a third, alternative way to describe any color, the hue/saturation/value or HSV system. You see this in some texts and in some computer programs, such as the *xv* color editor on UNIX systems. It can also be read out with the L-view editor on our PC, as I recall.

Description of whole scenes: contrast, gamma, and such. We don't see a scene as isolated pixels, but as patterns; we compare parts of the scene to each other, in large part. We also grasp the overall intensity of the scene, or its brightness. Let us define a couple more concepts for a whole scene:

Brightness is the average *value* over the whole scene, that is, the average R+G+B. If we are in a color-editing program such as *Adobe PhotoShop* and we use its ability to adjust brightness, we simply let the program multiply all R, G, and B values in all pixels by a common factor. For example, reducing brightness by 20% is the process of multiplying each R by 0.8, each G by 0.8, and each B by 0.8, in every pixel. Two pixels that used to be (38, 75, 88) and (127, 130, 100) become, respectively, (30, 60, 70) and (102, 104, 80) (after rounding to integers).

Contrast is the relative "brightness" (value, really) of different pixels. A true copy of a *scene* in which one pixel is 2.4 times as bright as another would have the value of pixels in the *image* in the same ratio. Thus, in the original scene the two pixels might have total light flux densities of 100 and 240 micromoles per square meter per second. In the image that preserves the same contrast, the sums R+G+B might be 44 and 106, respectively.

We do not have to leave the brightness or contrast intact to get an effective image. In fact, we rarely do. We adjust overall brightness of a monitor display to overcome distracting ambient light, or to get a high brightness that makes our pupils contract to give us greater depth of field, and so on. We also adjust the intensity of light sources illuminating prints or slides, for the same reasons.

Manipulating contrast is a bit more involved. Suppose we take two elements of the scene that differ by a factor of 2 in "brightness" and make the image represent the brighter pixel as *four*

time brighter. This increases the contrast, or the discernibility of one pixel from others of any different brightness. Color slide film does this, giving images that appear sharper, "crisper," more colorful (one more reason why reality can be disappointing compared to the images). Can we define contrast quantitatively? A simple definition is that contrast is the rate at which the *logarithm* of image brightness increases, compared with the rate at which the logarithm of brightness increased in the original scene. That is, $C = d(\ln I[\text{image}]) / d(\ln I[\text{scene}])$. In the example just given, the difference in $\ln I[\text{scene}]$ was $\ln 2 = 0.693$. In the image, it was $\ln 4 = 1.386$. The contrast is $C = 1.386/0.693 = 2$. Color slide film typically has $C = 1.6$. On the other hand, color print film commonly has C near 0.65, allowing a wide range of brightnesses to be captured on a medium that has a limited range of brightnesses in its images. On a photographic print or any printed image, we can distinguish only about a 16-fold range of brightnesses (deep blacks get diluted by reflections, for one), while in slides we readily see at least a 50-fold or 100-fold range.

Digital image processing to enhance comparisons, remove artifacts

Going directly to the crux of our recent work, let's discuss how manipulations of hue, saturation, and such are done and how they can help us to recognize features (such as species identities) and to quantify vegetative cover.

The control panels of various programs such as *Adobe PhotoShop* or *xv* allow one to manipulate these traits of an image. While I'm not privy to the source code, here are my informed guesses as to what computations these programs make; in some cases, I was able to verify them by comparing RGB values of initial and transformed images.

To adjust *brightness*, the programs simply multiply all R, G, and B values in every pixel by the same constant. Values that exceed the limit (digital value 255) are simply truncated at 255. Adjusting brightness alone does very little to help us recognize features, other than keeping shadow features from being so dark that low-brightness pixels merge into a common minimum value, or, analogously, keeping the highlights from "blocking up" near one pure white.

To adjust *saturation*, let's consider the extreme case of going to full saturation. the programs treat each pixel as a separate case. They find the minimum among R, G, and B in that pixel and subtract that value from R, G, and B, thus removing any dilution by white. A pixel that had initial RGB = (89, 37, 92) becomes (52, 0, 55) (though we're not done yet). A pixel that had RGB = (89,65,92) becomes (24, 0, 27); thus, pixels are *not* treated alike by a transformation independent of their original RGB values. To finish up, the programs scale all RGB values so that the largest nonzero number among R, G, and B becomes the maximum, 255. Thus, the pixel that started as (89, 37, 92) and then became (52, 0, 55) must have 55 converted to 255, by multiplying by $255/55 = 4.64$; the final RGB values are $(52*4.64, 0, 255) = (241, 0, 255)$. For less than complete saturation, each pixel is adjusted fractionally toward this goal. The white removal is at a fraction of completeness, and I believe that the scaling to maximum value is the same fraction (say, 0.42) of the way from original value toward 255. Increasing the saturation has helped us recognize many features, such as shaded soil's difference from shaded vegetation. It helps us develop quantitative classification schemes such as for "vegetation index." We'll discuss vegetation indices soon.

To adjust *contrast*, the programs make a "map" from each initial brightness to a final brightness. Let's measure the brightness with the total value of a pixel, $V = R + G + B$. For example, if the contrast, or "gamma" (γ), is to be set to 2, we might have $V(\text{initial}) = 10$ mapped to $V(\text{final}) = 10$, but $V(\text{initial}) = 20$ mapped to $V(\text{final}) = 40$. Continuing, $V(i) = 30$ would be mapped to $V(f) = 90$, and so on: $V(f) = 10 * [V(i)/10]^{**2}$.

The common contrast controls allow two modes of operation: (1) set gamma itself; (2) adjust the detailed shape of the curve relating $V(f)$ to $V(i)$, by grabbing points on the curve (appropriately displayed) with the cursor and dragging them around. First, what do we mean by changing gamma? Clearly, if $\gamma > 1$, at some $V(i)$, all higher values are jammed into the maximal value, 255. That is a possibility. More often, the programs just set the relation $V(f) = V(i)^{**\gamma}$ at low $V(i)$, with a smooth transition to a curve wherein $V(i)$ and $V(f)$ just meet at value 255. This is a "bowed up" curve of $V(f)$ vs. $V(i)$, such as we use in scanning our slides on Halena. The relation $V(f) = V(i)^\gamma$ only holds toward the low end of the curve. The effect is more pleasing than letting $V(f)$ hit 255 early, and it retains more information in the image.

To adjust *hue*, there are many options. We are free to map any color into any other color - say, blue-green to tan, red to orange, yellow to indigo, etc. Indeed, *xv* and other image editors allow color maps to be changed at will. This is practical only for a modest number of colors, so these editors allow it only in 8-bit color mode, in which there are 256 different colors. False-color images can be very useful and are commonly used in remote sensing. Infrared slide film is sensitive to IR, R, and G (and B, so we filter it out); these are developed as R, G, and B, respectively. Vegetation stands out typically as red or pink, soils are more neutral colors (grays, browns).

More interesting and useful are the continuous transformations, in which a single smooth function shows how each hue of the original image maps into a new, final hue. For example, we could move all hues up by 30 nm, so that 550 nm (yellow) becomes 520 nm (green), 450 nm (blue) becomes 420 nm (deep blue or indigo),...and 400 nm (violet) wraps back to $700 - 30 = 670$ nm (red). There is such a control in the *xv* editor. Since white has no hue, the *xv* editor has a separate control to alter whites. This control has as its resting position a dot in the center of a circle. One can use the mouse to drag the point off center toward any particular color, say, blue-green. The farther off center toward the pure color, the greater the degree of conversion of whites (and grays) toward this color. A third way of hue adjustment mixes it with adjustment of value, in part. One can change the γ value for R or G or B, individually.

What affects color in the original image?

This is clearly a consideration in photographing at different times of day, with varying cloudiness and haze. Color slide film is "objective," recording the flux densities in the light reflected from the scene, without regard to the "color" of light that illuminated the scene. Our eye, in contrast, compensates for much of the difference in reflected light when we view the same object under, say, sunlight vs. incandescent light, even though the relative flux densities in blue vs. red may drop 3-fold between the two cases. (There are fascinating presentations on this ability of the eye to adjust colors...and even to see nearly full color with only two primary colors in the illuminating light. I'll find a reference later.)

Let's examine, first, the effect of the color quality of light that illuminates a scene. It has a certain flux density at each wavelength, λ . Let's call this flux density $\Phi(\lambda)$. A given piece of the scene reflects a fraction of this flux at this wavelength. Call the fraction reflected $r(\lambda)$. Then, the flux density reaching the eye or the camera is the product of these two factors, or $\Phi(\lambda)r(\lambda)$. The red sensation is the integral over the red band of this final flux density times the relative sensitivity of the red cones of the eye at each wavelength. Let's keep things fairly simple. Suppose we had a scene illuminated with light at only three wavelengths, and the total flux density was 180, 200, and 150 at these (red, green, and blue) wavelengths...similarly to daylight. The surface reflects 0.25 of the light in the red, 0.80 of the light in the green, and 0.90 of the light in the blue. (It's a blue-green.) The final light flux densities reaching the eye (or the camera) in the R, G, and B bands are, respectively, $180 \cdot 0.25$, $200 \cdot 0.80$, and $150 \cdot 0.90$, or RGB = (45, 160, 135) - a blue-green. If we illuminated the scene with incandescent light, the flux densities in the red, green, and blue might be 300, 180, 50, respectively. The RGB value at the eye or camera will now be $(300 \cdot 0.25, 180 \cdot 0.80, 50 \cdot 0.90) = (75, 144, 45)$ - virtually green, having lost a lot of blue flux.

How can we correct for this? We can filter either the light reaching the camera or the light used to illuminate the image. In daylight, the relative proportions of R, G, and B were 0.9, 1.0, and 0.75 (normalizing them to 200, the highest value). In incandescent light, the relative proportions were 1.0, 0.6, and 0.17. We can convert these to the daylight values by multiplying the R, G, and B light, respectively, by 0.9, 1.67, and 4.5. We can't multiply light readily, and never with simple absorbing filters, so we actually might use a filter with relative transmissions of $0.9/4.5$, $1.67/4.5$, and $4.5/4.5$, or 0.2, 0.37, and 1.0 - a bluish filter. The total amount of light is reduced, and we have to accept that.

Most light sources we use have flux densities that vary smoothly with wavelength, making filtration simple. In fact, the distribution of light flux density with wavelength is often close to that of an ideal blackbody at a certain temperature (you can look up the blackbody law in various texts on physics, optics, photography, remote sensing, etc., if you're interested). Commonly, then, you hear a light source described by its color temperature - about 5800K for sunlight as it reaches the earth's surface through clear air, or 3200K for incandescent lights. Fluorescent lights are a bit different; they have peaks and troughs in their light output vs., wavelength, and every type of fluorescent light is a special case for filtering to approximate daylight color quality.

This scheme of correcting color rendition with filters has some limitations. First, we assumed that each part of the scene only reflects light incident from the source (the sun, a lamp, etc.). In fact, parts of the scene reflect light to each other. We see light from any part of the scene that was came from source to this part to eye, plus some that went from source to other part to this part to eye, etc. - we see light that has been "scattered" once, twice, or more. Color corrections for changes in the color balance of the source light only correct the first reflections properly. Fortunately, most important features of typical scenes are dominated by single scatterings.

Second, we can get light from schmutz in the light path, not part of the scene. This is strikingly evident when we look down from an airplane to the ground. High enough up, and with modest to heavy concentrations of aerosols in the air, the scene on the ground looks mostly like shades of blue. *No* filter can remove this "path radiance:" the extra light coming from scatterings off the aerosols is not proportional to the light coming from the scene, in any waveband. In fact, the path radiance has almost nothing to do with the scene. The only way to remove the

effects of path radiance is to compute the path radiance and subtract it digitally from the image. At the same time, one can correct for path absorbance of light going to and from the scene; this absorbance is wavelength-dependent. The challenge in making such corrections is that one needs to know just what kind of aerosol is in the way and how much of it is there. This is almost never known; one needs an instrument on the ground that measures transmission of the atmosphere at various wavelengths. Most remote-sensing dudes disdain to use ground information, or the expense and difficulty of placing sensors it too high, or both. An alternative is to deduce the aerosol scattering effects from the appearance (RGB balance and their absolute intensities) of a part of the scene of known color (set of reflectivities at various wavelengths). Even with much hard work to locate such parts, such as White Sands National Monument, only about 10% of the earth's surface can ever have its satellite views corrected reasonably accurately (Kaufmann, 1989). It's good that all our image analysis to date has been on aerial photographs no higher than 800 m above the ground, and in our normally clear air.

Some of our particular goals in image analysis

Our main goal is to quantify the amount of vegetation on each area of our transect, as leaf area index. We want LAI both as the average over a region and as its average over a given shrub. A second goal is to determine the species of each shrub or each grass patch we see in our photos. Species identity is useful in scaling up ET and assimilation, because the different species have different absolute rates of transpiration (E) and assimilation (A) per leaf area. We hope that community-wide averages of E and A are conserved between different communities, based on the ecological principles of water use, but we first need to know how the community members perform. Species identity is also a selling point for our methods to be adopted by community ecologists, who currently ID plants by more tedious ground-based surveys.

To accomplish these goals, we need to analyze each image for fraction vegetation in view, and for the colors and special morphological features of each shrub or grass patch. We can convert from fraction vegetation in view (f_{veg}) to leaf area index, using models of light interception in stands of vegetation (though not trivially; more on this, later).

Extracting "fraction vegetation in view" from an image

At the finest level, each part of a scene either has leaf in view, or non-leaf (esp. soil). We have then to classify each pixel, then count up how many are leaf, how many are non-leaf. In false-color IR photos, classification is reasonably clear-cut; for any pixel, one can compute a vegetation index, $VI = (IR-R)/(IR+R)$. If this value exceeds a cutoff, the pixel is classed as vegetation, otherwise as non-leaf. The separation of the two classes is wide, on almost all soils, though its exact center depends on soil type. One can use a conservative estimate of the cutoff and have a high rate of proper classification. In true-color images, the separation is often not quite as dramatic and clear-cut, but it is practical.

Mixed pixels. Our aerial photos are taken from a great enough height that individual leaves are not resolved, except on yuccas. If our pixels correspond to areas that are 5 - 10 cm on a side in the original scene, then for all our plant species any pixel is likely to be only part leaf and part soil (I'm ignoring the stems, an interesting, if usually minor problem). We shouldn't classify any pixel as purely leaf or purely soil. A dichotomous classification is biased against one category or the other. Consider a scene with leaves distributed randomly across all areas, with a true f_{veg} of

0.7 (70% of the scene as viewed is really leaf). If we use pixels larger than individual leaves, there is a certain probability that any pixel is over or under the threshold to be called leaf. Consider dividing the pixel into, say, 10 parts. The chances that a pixel has 1, 10, 20, 30, ... or 100% leaf in view follow the binomial distribution, with the probability of n of the 10 parts being leaf given by

$$P(n) = \text{nth term in the polynomial } (0.7 + 0.3)^n$$

The probabilities are then 0.3^{10} , $10(0.3)^9 0.7$, $10 * 9/2 * (0.3)^8 (0.7)^2$, etc., or 0.00001, 0.00014, 0.00145, 0.00900, 0.03676, 0.10292, 0.20012, 0.26683, 0.23347, 0.12106, and 0.02825. If our cutoff to call a pixel leaf-like is it being 50% or more as leaf, then the sum of the probabilities of 100%, 90%, 80%, 70%, and 60%, and half the probability of 50% cover (it will be counted half the time as qualifying, being on the cusp) is 90.2%, not 70%.

What's the way out of this bind? We can weight each class by the fraction of vegetation it contains. The sum, over these 10 possibilities, of f_{veg} in a pixel times the probability of seeing a pixel with that f_{veg} is $0 * 0.00001 + 0.1 * 0.00014 + 0.2 * 0.00145 + \dots = 0.70$, exactly. We're then counting each pixel not as either leaf or non-leaf, but as having a fractional membership in each. Realistically, we can do this if we can determine how far this pixel's vegetation index (VI) is along the range from the VI in pure soil to the VI in pure leaf. Suppose leaves have $VI = 0.68$ and soil has $VI = 0.12$. A pixel with $VI = 0.44$ is 0.32 units above soil VI, which is $0.32/0.56$ of the way toward pure leaf, or 57% leaf. We would count it as such in our sum over pixels. This is the method of end-member mixing. It works with more than two traits being resolved (leaf vs. nonleaf). Maybe we are resolving sunlit leaf, shaded leaf, sunlit soil, and shaded soil. We need at least three indices, V_1 , V_2 , and V_3 (leafiness, soilness, shaded-soilness?). Then, we find pixels in the scene that are pure end-members (shaded soil, ...). We score each of the end members on the 3 indices. Call the score of the i-th end-member on the j-th index V_{ij} . Any pixel we choose at random will likely be a mixture of all 4 end-members: it is a fraction f_1 of end-member 1 (say, sunlit-leafiness), a fraction f_2 of end-member 2 (shaded-leafiness), and so on. This pixel will, correspondingly, have intermediate scores on all 3 color indices. We can use its scores on the 3 indices to figure out what mix of end-members composes it. Our chosen pixel has a score V_j on the j-th index:

$$V_j = f_1 V_{1j} + f_2 V_{2j} + f_3 V_{3j} + (1 - f_1 - f_2 - f_3) V_{4j}$$

Here, I've used the fact that all the fractions have to sum to 1, so that $f_4 = 1 - f_1 - f_2 - f_3$. We can rewrite the equation as

$$V_j = f_1 (V_{1j} - V_{4j}) + f_2 (V_{2j} - V_{4j}) + f_3 (V_{3j} - V_{4j}) + V_{4j}$$

Let's write the 3 color scores as a vector, and we'll get this set of linear equations:

$$\bar{v} = \tilde{A}\bar{f} + \bar{v}_4$$

To solve this for the vector of components f_i , let's move the vector \bar{v}_4 to the left-hand side:

$$\bar{v} - \bar{v}_4 = \tilde{A}\bar{f}$$

Formally, we can solve this by multiplying each side by the inverse of matrix A:

$$\bar{f} = \tilde{A}^{-1}(\bar{v} - \bar{v}_4).$$

The way we proceed is then: (1) Find pixels with the 4 pure end-members in the scene; (2) devise 3 "vegetation" indices (really, distinctness indices), which vary as widely as possible over the 4 end-members AND which are linear functions of the scene RGB values (more on this, shortly); (3) evaluate the 4 pure end-members on these indices - that is, compute the V_{ij} ; (4) compose the matrix A ; (5) compute its inverse, \tilde{A}^{-1} ; (5) go through every pixel of interest and compute, first, its scores V_j on each index; second, the vector $\bar{v} - \bar{v}_4$; and, finally, the fraction of each end-member present in it, the 4 different f_i , using the last equation.

Here are a few caveats. First, this method assumes that the 4 end-members present in the scene reflect light independently to our camera or other sensor. That is, there are no multiple scatterings. Of course, this is not strictly true. If there is a lot of sunlit leaf in the scene, it means that the shadows on the soil come from strong exclusion of light; the soil may be lit in good part by light that hit a leaf (and is thus greenish) and less (than in lightly shaded areas) by diffuse skylight, which has a reliable color (bluish). The color, and the color scores V_{4j} , for shaded soil, will vary with the fraction of sunlit leaf, instead of being independent of this sunlit-leaf end-member. How serious is the effect? If we're mostly after vegetation, not very, in the case given; the relative error is mostly in computing the shaded-soil fraction. Second, the color indices we choose should be linear functions of the RGB (or IR-R-G) components of the scene radiance. This is because light (the R, G, and B parts, taken separately) mixes purely additively from our end members. So, it isn't really legit to use a nonlinear index like $(G - R)/(G + R)$. Third, the indices should be chosen to make the end members most distinct in each index. If we have only three colors (R, G, B), of course, we have no choice; there are 3 colors and 3 indices, and we might as well choose $V_{1j} = \text{amount of R}$, $V_2 = \text{amount of G}$, and $V_3 = \text{amount of B}$. Any other choices of linear combinations of colors just scramble the indices but do not change the final calculation (proof of this requires some linear algebra; trust me). This is rather unsatisfying, because nonlinear indices may give better separation of end members, the way that NDVI = $(IR - R)/(IR + R)$ separates vegetation from soil nicely. There's a bit more to say on NDVI. The division by $(IR + R)$, the so-called normalization, is largely to remove the effect of differences in incident sunlight intensity when comparing scenes taken on different days. The assumption is that we can't measure the incident sunlight from in-orbit measurements (which is not strictly true), and that the absolute reflectances in IR and R have less information than their relative values. This is not always true. Recall that shaded vegetation still looks green, but darker. If we normalized all our R, G, and B readings (as $R/(R + G + B)$, etc.), we might not distinguish shaded leaves from sunlit leaves, or we might do so poorly. This throws away some valuable information that might distinguish plant species, such as *Rhus*, with its mostly-shaded leaves, from other species. The only cautionary note here is that unnormalized indices can only be used in analyzing pixels / whole images that were all taken under the same lighting conditions - same day, with no differences in cloud shadows in different areas.

How about some good news? Our indices, which are developed from *known* end members (sunlit leaves...shaded soil) are, by their origin, not confused by the particular soil color on which the vegetation is found. This is *not* true for NDVI and several other indices. NDVI assumes that all soils have the same NDVI, near zero. Our red soils, however, often have $IR - R$ negative. A scene (or single pixel) with $NDVI = 0$ is really partly vegetated, and we mistake it for totally

unvegetated. To keep this advantage, we have to analyze separately each part of the scene that is on a different soil type. In slides that show both red and white soils, we should analyze the part on red soil separately from the part on white soil. This makes it more tedious to automate the classification of pixels with a Fortran program, because we now have to tell the program the exact (and complicated) boundaries between red- and white-soil areas.

Extracting LAI from fraction of vegetation in view

The theory of estimating LAI from f_{veg} is straightforward when the vegetation is uniform, such as in a dense row crop, and its distribution of leaf angles (LAD) is known. In this case, light penetrates into the canopy (or exits the canopy) with a probability $\exp(-KL)$, where K is the extinction coefficient for light leaving/entering at a specified view angle and L is the leaf area index from top of canopy to the point of interest inside the canopy. In our case, $L = LAI$, since we're interested in the probability of seeing the soil at the very base of the canopy. The probability of seeing soil is $\exp(-KL)$, so $f_{veg} = 1 - \exp(-KL)$. We can compute K as (average cosine of the angle between leaves and sunlight or view direction) / (cosine of sunlight or view direction). The derivation of this relation is reasonably straightforward; ask me if you're curious, or look in Juha Ross' book (1981). In any event, if we know the LAD of a kind of vegetation, and we know the view angle at which the photo was taken, we can compute K . For example, if leaves are randomly over all zenith angles (measuring how far from vertical is a line normal to the leaf) and over all azimuth angles (measuring how far the normal vector to the leaf is rotated CCW from north), then $K = 0.5/\cos\theta_v$. Here, θ_v is the zenith angle of the view direction, and it's zero for a straight-down (nadir) view. In this simple case, then, we can invert the equation and calculate L :

$$e^{-KL} = 1 - f_{veg}$$

Taking logarithms of both sides, we have

$$L = -(1/K) \ln(1 - f_{veg})$$

Take the case that $K = 0.5$ and $f_{veg} = 0.7$ (70% vegetation in view); we calculate $L = -2 \ln 0.3 = 2.4$.

OK, how complicated do things get in the practical analysis of f_{veg} to estimate LAI? First, it's clear that we have to know the leaf angle distribution of our vegetation. We have to go out in the field and measure it for tarbush, then for *Larrea*, and so on. With no other information, we might assume that the leaf angles are random, and $K = 0.5$ for a nadir view. This is a good estimate for *Larrea* and mesquite (whose leaves are individually at specific angles, but which average to all directions). It's not so good for tarbush, which clusters more about the horizontal. Second, the estimate of LAI is misled if leaves are clumped. Consider the case that Felicia was presenting in her talk at the Biology Symposium. We have two pixels, one with $LAI = 0$ (bare soil) and one with $LAI = 4$ (a dense shrub). The average LAI is 2. Suppose, however, that we had poorer spatial resolution in our slide and lumped these 2 pixels together. We would average f_{veg} in our view, getting $f_{veg} = 0.5*(0 + [1 - \exp(-0.5*4)]) = 0.432$. Thinking that this is uniform vegetation (in the absence of any more information), we'd take the average $f_{veg} = 0.432 = 1 - \exp(-K*L')$, where L' is the presumed average, uniform LAI. We'd compute $L' = -2 \ln 0.568 = 1.13$. We'd estimate only 56% of the true LAI, because we were averaging a nonlinear function of LAI as if it were linear.

We do this kind of mistaken averaging when our pixels are large compared with shrub size,

certainly. One shrub with LAI = 1.13 looks like any number of smaller shrubs with LAI = 4 that, in total, cover half the land area. More insidiously, we do this kind of averaging when our pixel size is big compared to leaf or leaflet sizes, which it is in aerial photos of most shrubs. Here's the "naive" theory: (1) we find a shrub and look for some pixels that have the highest greenness (say, as G-R); we call these pure sunlit leaf area; we also look for "pure" shaded leaf, pure sunlit soil, and pure shaded soil; (2) we then look at interior areas of the shrub and compute the fraction of leaf in each pixel. We do end-member mixing to resolve each pixel into a mix of the 4 categories, as described above; (3) we compute an average f_{veg} over the shrub, or over modest areas, since LAI will vary, tailing off toward the edge of the canopy. What's wrong with this picture? Suppose our pixel is 5 cm on a side and our leaves are 0.5 cm on a side. Each pixel covers 100 leaf areas. Suppose the real f_{veg} is 0.75. If leaflets are really randomly distributed, then we'd expect 75 "hits" on vegetation out of the 100 sub-pixels. However, the distribution gets narrow when there are many bins. In the binomial distribution, if f_{veg} is fixed and n = number of sub-pixel possibilities gets large, the binomial distribution soon approaches the Gaussian distribution, with a mean number of hits nf and a standard deviation of $\sqrt{nf(1-f)}$. The coefficient of variation (CV) gets small - almost all pixels see the same f_{veg} . One can compute CV as $\sqrt{(1-f)/nf}$. If $n = 100$ and $f_{veg} = 0.7$, $\sigma = 0.045$ and $CV = 0.065$. The chance of finding a pixel with $f_{veg} = 0.9$ (which is a bit more than 4 standard deviations out from the mean) is very small, about 1 in 1000. Thus, it's likely that what we picked as 100% leaf is about 85% leaf. All our estimates of f_{veg} are inflated by the factor 100/85. We then think that what is 70% leaf (LAI = 2.4, if $K = 0.5$) is 82% leaf (LAI = 3.4).

Is there a way out of this bind? I think so, but it's non-trivial to do the theory and the practical calculations. Here are my thoughts to date: First, we do have an idea of how big n is, our number of subpixels at the leaf size. We have three ways of knowing n : (a) By physical leaf or leaflet size, compared to known aerial photo pixel size. This is not to be trusted that much, because leaflets of mesquite are clearly not distributed randomly with respect to each other on the same leaflet, though they are pretty randomly placed between leaflets on different leaves at different heights as we look down on both. (b) By the histogram of f_{veg} for all the pixels on the shrub. If n is large, the histogram will have a narrow distribution; if n is small, it will be wider. The value of f_{veg} also enters into determining the standard deviation, but it occurs as $f(1-f)$, which varies only modestly (1.6-fold, from 0.16 to 0.25) even as f varies from 0.2 to 0.8. (c) By direct tests on photos of shrubs with known f_{veg} and LAI. We can take photos about 5 - 7 meters above a shrub and get exact f_{veg} , because our pixels will be smaller than leaflet sizes. We can then mathematically smear out pixels to the size of our aerial photos and see what is the distribution of f_{veg} in various pixels. We thus get an effective value of n that incorporates any nonrandomness in leaflet distribution; we solve two problems at once.

How might we apply the last of the 3 methods above? If we have an aerial photo and we know it's a mesquite in one area, we know the effective value of n in our aerial-photo pixels. We (I'll finish this later. Right now, I don't have an answer.)

Extracting species identities from aerial photos

On the ground, different species of plants can often be discriminated by gross morphology and color, which we can also see in aerial photos - e.g., the splayed branches of *Larrea* vs. more compact branch arrangements in other shrubs. A sizeable fraction of identifications require a

closer look, such as at leaf size, which we can't resolve in aerial photos except for the largest leaves (yucca). Thus, there are only a few à priori traits that are visible in aerial photos and that separate plant species, such as leaf color. Some of these traits are only applicable part of the time, such as size; a young *Rhus*, destined to attain the size of older *Rhus*, will look the size of an average tarbush. Identifying species from photos will always be a heuristic process, developing rules, testing them, and revising them in light of field checks. We aim for the most robust rules, applicable to shrubs and grasses in varied condition - e.g., applicable to tarbush with either low leaf area and high leaf area, despite their great differences in physical appearance. We also want rules that are explicit and objective, not dependent on intuition of any one operator who, however skillful in classification, can't put the rules down verbally.

What aspects of our aerial photos can we use to our greatest advantage? Here are some photographic traits of different species that are candidates for ID'ing species:

- * Color: light green separates mesquite from most other shrubs.
- * Color: *Rhus* is darker than *Larrea*.
- * Saturated color: *Bouteloua* and *Sporobolus* come out distinctly blue-green, while fluff-grass is a plainer green.
- * Percent soil viewed through crown: tarbushes with low leaf area development show much more soil than other shrubs.
- * Patterns of shade on leaves: *Rhus* in July, 1995 mostly had bare top branches shading the inner leaves notably; there's more shaded leaf area (dark green on a saturated-color photo) on a *Rhus* than on other shrubs.
- * Shadow lengths: *Rhus* are taller than they are wide; viewed from above, their shadows are as long as the canopy is wide, or longer; other shrubs have shadows shorter than their canopy is wide.
- * Leaf size: we can see single leaves of yucca. Together with their light-green color, this is diagnostic.

This section will be revised as we go. We've been viewing our slides and the scanned versions and the color-manipulated scans, to try devising rules of classification. Then we'll go to the field, soon, to test the rules and revise them. In our lab work to devise identification rules, we need to (1) make full use of the slide (viewed in detail with a dissecting scope - less onerous on us and on the slide than long-term projection), the scanned image, and color-manipulated scans, and (2) write down the rules as clearly and fully as possible, and as many rules as are defensible from our field knowledge plus the revelations of color manipulations.

BIBLIOGRAPHY - informally presented

xv "man pages": access these by typing "man xv" on either Sun (rhino or bilbo) in any window with the shell prompt, or look for the hard copy I made.

Mees, C. E. K., and T. H. James. 1966. *Theory of the photographic process*. Macmillan, New York. See this tome for discussions of film contrast, grain, resolution, etc., and for some color theory.

Kaufman, L. 1974. *Sight and Mind: An Introduction to Visual Perception*. Oxford University Press, Oxford. There are very nice sections here on how the eye perceives color, on the Munsell color solid, etc.

Encyclopedia Americana, and other good encyclopedias: articles on "color." The qualitative

aspects of color are discussed here, and some of the quantitative aspects, such as color temperature. You won't find digital color discussed here, at least in editions I have access to.