2 Computers and Color

Color by the Numbers

Computers know nothing about color except what we humans tell them. They're just glorified adding machines that juggle ones and zeros to order. One of the many ways we use numbers on the computer is to represent color. To do that, we need some kind of mathematical model of color. Applying mathematical models to reality is always tricky, but it's particularly so when dealing with something as slippery and subjective as color. The great mathematician, Sir Isaac Newton, made many important discoveries about color, but as far as we know he never tried to model it mathematically. Instead he went onto simpler subjects like inventing calculus and discovering the mechanical laws of the universe.

In Chapter 1, we explained that color is really something that only happens in our heads—it's the *sensation* we experience in response to different wavelengths of light. When we talk about measuring color, what we're measuring isn't really color itself, but rather the stimulus that evokes the sensation of color—the makeup of the light hitting our retinas. We can correlate light measurements with the color people experience, but the correlation isn't perfect.

In this chapter, we'll examine the various number systems we use to represent color, explain what these numbers mean, and show how, without color management, the same set of numbers will produce very different colors in different situations.

Color by the Numbers

In the previous chapter, we explained how it's possible to produce all the colors people can see using only red, green, and blue light—the "additive" primary colors. When we reproduce color on a physical device, whether it's a monitor, a piece of transparency film, or a printed page, we do so by manipulating red, green, and blue light.

In the case of true RGB devices such as monitors, scanners, and digital cameras, we work with red, green, and blue light directly. With film and printing, we still manipulate red, green, and blue light, but we do so indirectly, using CMY pigments to subtract these wavelengths from a white background—cyan absorbs red light, magenta absorbs green light, and yellow absorbs blue light—hence the term "subtractive" primary colors. Most digital color is encoded to represent varying amounts of either R, G, and B or C, M, and Y, or, in commercial printing and some (but not all) desktop printers, C, M, Y, and K (for BlacK). (See the sidebar "Why CMYK?")

Unfortunately, these mathematical models of color are quite ambiguous. You can think of an RGB or CMYK file as containing, not color, but rather a *recipe* for color that each device interprets according to its own capabilities. If you give 20 cooks the same recipe, you'll almost certainly get 20 slightly different dishes as a result. Likewise, if you send the same RGB file to 20 different monitors, or the same CMYK file to 20 different presses, you'll get 20 slightly (or in some cases, more than slightly) different images. You can readily see this in any store that sells television sets. You'll see twenty televisions all lined up, of various makes and models, all tuned to the same station, and all producing somewhat different colors. They're receiving the same *recipe* but their different characteristics generate different visible results. This even happens within the same make and model of television.

The RGB and CMYK models originated in the analog rather than the digital world. Neither was designed as an accurate mathematical description of color: they're really *control signals* that we send to our various color devices to make them produce something that we eventually experience as color. So you should always think of RGB or CMYK numbers as tuned for a specific device.

Why CMYK?

Why CMYK rather than CMY? In theory, pure cyan absorbs 100% of red light just as pure magenta and yellow fully absorb green and blue light respectively. A combination of perfectly pure cyan, magenta, and yellow colorants would absorb all light, which people would see as black (if they could see it all—the only perfectly black objects we're aware of in this universe are black holes, which we can't see directly). When one or more colorants aren't 100% pure, some light is reflected instead of being absorbed. This is why many toner-based devices have a greenish three-color black, and why three-color black on a printing press is usually a muddy brown. The colorants are simply not perfect. Photographic dyes come close, but inks and toners, for example, have to satisfy many different physical requirements besides color, such as adhering to the paper and each other, drying in a reasonable length of time, being fade-resistant, and being affordable. This almost invariably involves compromising the color purity. So to get a better black that will absorb as much light as possible, as neutrally as possible, we use black ink. Another good reason for printing with black ink is that black-only objects such as text are a lot easier to print when you don't have to perfectly align the cyan, magenta, and yellow versions.

You may also wonder why it's CMYK rather than CMYB. There's general agreement that referring to black as "B" would lead to confusion with Blue. Press operators often refer to cyan and magenta as blue and red, which is, of course, incorrect and not something we personally encourage. But it's an ingrained, time-honored practice, and trying to change it is, as our friend and colleague Herb Paynter would say, "a hill that ain't worth dyin' on." There are various theories as to why "K" was chosen, but the likeliest, in our view, is that it refers to "key"—the master plate to which the other three colors are registered. Since black is the darkest color, it's usually used as the key because it's the easiest to see. Whatever the reason, we print with CMYK, not with CMYB.

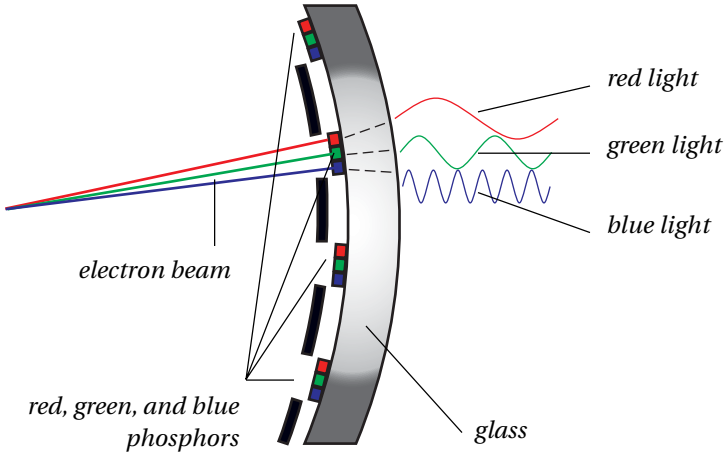
Analog Origins

The numbers in RGB and CMYK files don't really represent color. Instead, they represent the amounts of *colorants*—the things our devices use to make a color. Both RGB and CMYK were used in the analog world long before they were translated to the digital world.

CMYK printing has been around as a mass-market commercial process since the early 1920s, and until pre-press went digital in the 1970s, CMYK separations were made optically by photographing the original art through C, M, Y, and neutral-density (for the black plate) filters. The earliest scanners used analog RGB signals. The scanners' RGB signals were typically converted directly to analog CMYK which was used to expose film from which printing plates were made. When we started making color digitally, we simply used digital RGB and digital CMYK to mimic their analog predecessors. In short, it was the easiest way to make the transition to digital color, but not necessarily the best way.

Monitor RGB. When we display color on a monitor, we do so by spraying streams of electrons that strike *phosphors*. Phosphors are chemical and mineral compounds that emit light when they're struck (the technical term is *excited*) by a beam of electrons. Color monitors use three different phosphors painted on the inside of the faceplate that emit red, green, and blue light, respectively. By varying the strength of the electron beam, we can make the phosphors emit more or less red, green, and blue light, and hence produce different colors (see Figure 2-1).

Figure 2-1
Monitor phosphors



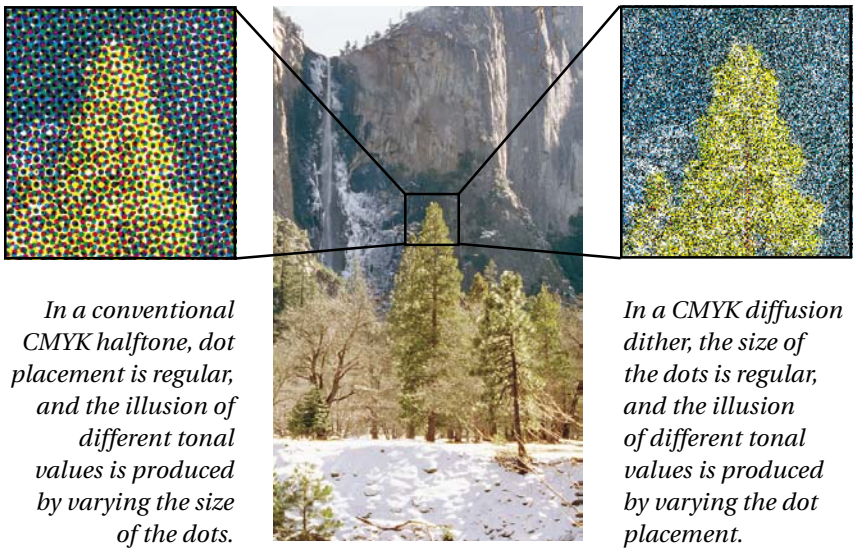
But the precise color that the monitor produces depends on the type of phosphors used, their age, the specific circuitry and other characteristics of the monitor, and even the strength of the magnetic field in which the monitor is located. All monitors' phosphors produce something we recognize as red, green, and blue, but there are at least five quite different phosphor sets in common use, and the phosphors can vary substantially even in a single manufacturing batch. Factor in individual preferences for brightness and contrast settings, and it's highly unlikely that any two monitors will produce the same color from the same signal, even if they're two apparently identical monitors bought on the same day.

Scanner RGB. When we capture color with a scanner or digital camera, we do so using monochromatic light-sensitive sensors and red, green, and blue filters. Each sensor puts out a voltage proportional to the amount of light that reaches it through the filters, and we encode those

analog voltages as digital values of R, G, and B. The precise digital values a scanner or camera creates from a given color sample depend on the makeup of the light source and the transmission characteristics of the filters. As with monitor phosphors, scanner and camera filters vary from vendor to vendor, and they also change with age. Scanner lamps also vary both from vendor to vendor and with age, and the light source in a digital camera capture can range from carefully controlled studio lighting to daylight that varies from exposure to exposure, or even, with scanning-back cameras, over the course of a single exposure. So it's very unlikely that two capture devices will produce the same RGB values from the same color sample.

Printer CMYK. When we print images on paper, we usually do so by laying down dots of cyan, magenta, yellow, and black ink. In traditional halftone screens, the spacing is constant from the center of one dot to the next, but the dots vary in size to produce the various shades or tints. Many desktop printers and some commercial press jobs use different types of screening, variously known as *error diffusion* or *stochastic screens*, where each dot is the same size, and the color is varied by printing a greater or smaller number of dots in a given area (see Figure 2-2, and the sidebar, "Pixels, Dots, and Dithers," later in this chapter).

Figure 2-2
CMYK halftone



In a conventional CMYK halftone, dot placement is regular, and the illusion of different tonal values is produced by varying the size of the dots.

In a CMYK diffusion dither, the size of the dots is regular, and the illusion of different tonal values is produced by varying the dot placement.

Pixels, Dots, and Dithers

Wherever possible we try to avoid nitpicking about terminology, but the practice of using the terms "dpi" (dots per inch) and "ppi" (pixels per inch) interchangeably inevitably leads to confusion, because dots and pixels are distinct entities with different properties. (Some digital mavens insist on using the term "spi" (samples per inch) rather than dpi when discussing scanner resolution—*that's* nitpicky.

Pixels represent varying levels of density. A single pixel can not only be red, green, and blue at the same time, but have different intensities of red, green, and blue, rather than just on or off. This is where we get the term *continuous tone*. A monitor is an example of a continuous-tone device.

But most digital hard-copy output devices aren't continuous-tone. Instead of pixels, they print dots of ink or toner, which are either on or off, present or absent. Digital printers' resolution is expressed in dots per inch, which describes the number of locations per inch in which the printer can either print a dot, or not print a dot. We can't vary the density of ink, nor can we vary the size of the dot. Rather, we can only tell the output device whether or not to print a dot at each location. A 600-dpi laser printer, can print, or not print, 600 dots per linear inch, while a 2400-dpi imagesetter has to make that same decision 2400 times per linear inch. Each dot has the same density—the only thing we can control is the location of the dots.

We make digital printers produce the *illusion* of continuous tone by arranging these equally sized, constant-density dots using some kind of *dithering*—a way of arranging the dots in a pattern that isn't obvious to the eye. Back in the analog days, prepress folks converted continuous-tone originals to *halftones*—a kind of dither where the dots are constant-density and equally spaced, varying in size to produce the illusion of darker or lighter shades—by projecting the original onto plate material through color filters and a screen like the ones you find on a screen door. The holes in the screen acted as pinhole lenses, producing large dots in the dark areas and smaller ones in the light areas.

Most presses still use this kind dithering, but to make a digital halftone on an imagesetter or platesetter, we arrange its printable dots into larger groups called *halftone cells* or *halftone spots*. We simulate the traditional analog halftone by turning dots on and off in a cel. This type of dither is also known as "AM (Amplitude Modulation) screening," or "conventional halftone."

But the precise color that the printer produces depends on the color of the inks, pigments, or dyes, the color of the paper stock, and the way the colorants interact with the paper, both chemically and physically. Inkjet printers commonly show color shifts over time (most obvious in neutrals) when ink and paper aren't appropriately matched. Color laser printers and color copiers are very susceptible to humidity change. On a commercial press, the color can vary with temperature, humidity, prevailing wind, and the state of the press operator's diet and marriage, but that's another story! So it's very unlikely that two different printing devices will produce the same color from the same set of CMYK values.

Halftoning converts continuous tone images, such as digital files or scans, from pixels into dots so the image can be reproduced on an inkjet printer or a printing press, for example. The best-known type of halftoning is sometimes called *ordered dither* or "conventional halftone dot" in addition to the aforementioned "AM screening". A different kind of halftoning, *error diffusion*, also goes by names such as "FM screening" and "stochastic screening".

AM screening varies the size of the dots, but not their location. Darker areas have larger dots and lighter areas have smaller dots (or no dots). This type of halftoning is common for printing presses as a carry over from making plates through a "screen."

Because of the pattern generated with AM screening, undesirable artifacts can occur with multiple inks when their "screens" conflict with each other. Combating this necessitates rotating each inks' screen to avoid these conflicts, hence the term "screen angles."

The other kind of dithering, eror diffusion or FM screening, is used on most inkjet printers, and occasionally on presses too. FM screening varies the location of the dots, but not their size. Darker areas have more dots closer together while the lighter areas have fewer dots dispersed farther apart. The more random nature of FM screening gives high resolution inkjet output the appearance of continuous tone. FM screening is sometimes used on press, but since small variations in press behavior are exaggerated much more by FM screening than by AM screening, it's usually limited to premium jobs in shops that have gained considerable experience using FM screens.

Dye sublimation, and photographic output methods are considered continuous tone because the colorants' density can be controlled, so dots, and therefore halftoning. aren't needed.

The importance of this to color management is that ICC-based color management only works on pixels, not dots, but the final output is more often than not created with dots. The effect of screening algorithms can affect what we see compared to what was measured and predicted by the color management system (CMS).

So we need to take the screening algorithm into account when we color-manage a device, because different screening algorithms will produce different tonal renderings. (See "Tone Reproduction Characteristics," later in this chapter.)

Digital Evolutions

The point of the previous section is that RGB and CMYK are fundamentally analog concepts—they represent some amount of *colorants*: the dyes, inks, phosphors, or filters we use to control the wavelengths of light. RGB devices such as televisions, monitors, scanners, and digital cameras to this day, and for the forseeable future, all have analog components—things that work in terms of continuous voltages: magnets, lenses, mirrors, and phosphors and filters baked in chemical labs. CMYK printers still deal with the idiosyncracies of chemical inks, dyes, and pigments on sheets of mashed wood pulp that we call "paper."

However, RGB and CMYK numbers are nevertheless … numbers. This has made them ripe for adoption into the digital age where numbers themselves take the form of bits and bytes (see "How the Numbers Work," later in this chapter).

Over the years we've seen more and more analog components replaced by digital. The Darwinian force that drives this evolution is, very simply, money. Digital components are faster, cheaper, and (most importantly from the point of view of color management) repeatable and predictable. All of these benefits translate directly into monetary savings.

But keep in mind two things about this evolution. First, it's incremental. Companies usually produce products that are small improvements over previous technologies—despite what their marketing brochures say. New products and subcomponents of products must coexist with old, and new technologies must be usable by people who have worked with old components for years. Second, because of this incremental evolution, digital RGB and CMYK are often designed to mimic their analog predecessors. The result has been that digital color-reproduction equipment often has odd little idiosyncracies that might not be there if the things had been designed from the ground up to be digital, or mostly digital, devices.

An example is the evolution of the imagesetter, the output device that generates the film used to image plates for offset lithography. Imagesetters evolved from the analog methods of imaging film using photographic techniques—for example, projecting a photographic negative through a fine screen to produce a halftone image (hence the terms "screening" and "screen frequency"). This analog photographic process was replaced by computers controlling lasers that precisely exposed the film microdot by microdot, but these imagesetters still needed traditional analog darkroom equipment, chemistry, and photographically skilled technicians to develop the film. However, bit by bit, even the darkroom processing was replaced by digitally controlled processor units that control all aspects of film development.

But why did we need film at all? Because extremely expensive (and therefore not expendable) printing presses had almost as expensive platemakers that required film for platemaking, and because creating an analog proof from the film was the only affordable method in place for creating a contract between the print client and printer (premium jobs sometimes use actual press proofs—in effect, separate press runs for

proofing only—but they're brutally expensive). Nowadays, however, as platemaking and digital proofing technologies become more reliable, film itself is being skipped (with obvious cost benefits), and the digital process is being extended from the computer right up to the platesetter itself, even including digital platemakers that image the plates right on the press rollers.

What does all this mean for digital RGB and CMYK numbers and color management? It means that all the digital computation and control of the numbers exercised by color management are only as good as their ability to model the behavior of analog components. Digital color management *alters the numbers* to compensate for the behavior of the various analog components. As such, the strengths and weaknesses of color management lie entirely in how well our digital manipulations model the behavior of the analog parts, including that most important analog 'device' of all, the viewer's eye.

In a moment we'll examine the key parameters that describe the analog behavior of color-reproduction devices. But first, we should examine the *digital* part of digital color—the numbers—a bit more closely.

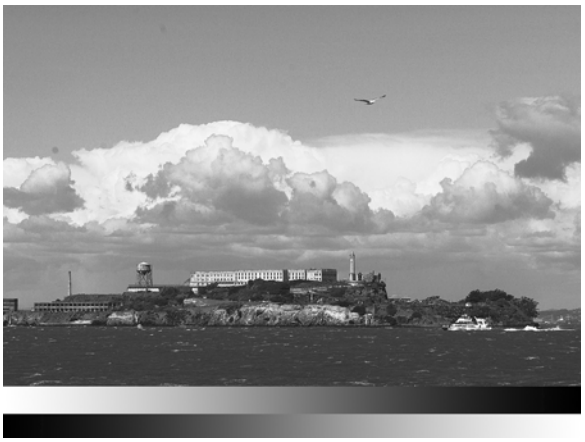
How the Numbers Work

Let's pause for a moment and examine the systems we use for representing—or, more accurately, *encoding*—colors as numbers in a computer. We'll take this the opportunity to clarify a few points that often confuse people about the basics of digital color, which propagates into confusion about color management. Even if you're extremely familiar with the basics of bits, bytes, tones, and colors, this section is worth reviewing as we make a few key points about the difference between colors-as-numbers and colors as "Real World" experiences.

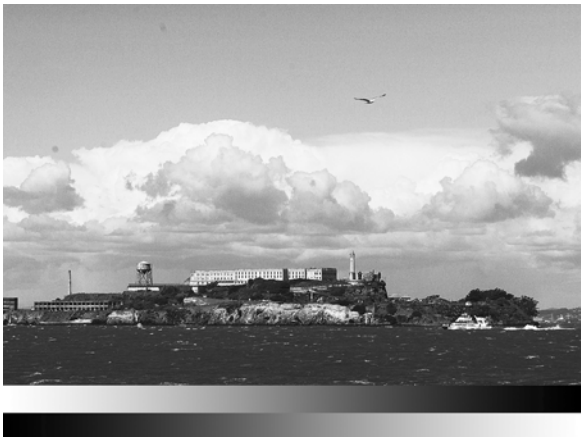
The system computers use for encoding colors as numbers is actually quite simple: colors are comprised of *channels*, and each channel is subdivided into *tone levels*. That's it! We start with a simple model of color perception—the fact that colors are mixtures of red, green, and blue in various intensities—and then we adapt this model for efficient storage, computation, and transportation on computers. The number of channels in our encoding system is usually three, to correspond to our basic three-primary way of seeing colors. The number of tone levels in our

encoding system is usually 256, to correspond to the minimum number of tone levels we need to create the illusion of *continuous tone*—to avoid the artifacts known as *banding* or *posterization*, where a viewer can see noticeable jumps between one tone level and the next (see Figure 2-3).

Figure 2-3
Levels and posterization



256 shades of gray provides the illusion of continuous tone.



With only 128 shades of gray, some tonal detail is lost, and we start to see hints of posterization.

With only 64 shades of gray, we see banding in the gradients and in the sky.

Why 256 Levels?

This number, 256, seems arbitrary and mysterious to some people, but it crops up so many times in computers and color that it's worth making your peace with it. It's not that mysterious. We want to be able to represent enough tone levels so that the step from one tone level to the next is not visible to the viewer. It turns out the that number of tone levels needed to produce the effect of a smooth gradient is about 200 for most people. So why not encode only 200 levels? Why 256? For two reasons.

Headroom. It's useful—in fact, essential for color management—to have some extra tone levels in our data so that the inevitable losses of tone levels at each stage of production (scanning, display, editing, conversion, computation, printing) don't reintroduce banding.

Bits. The second reason is just that we use bits to represent these tone level numbers. Seven bits would let us encode only 128 tone levels (2^7), which would be a surefire way to get banding in our skies and blotches on the cheeks of our fashion models. Eight bits lets us encode 256 tone levels (2^8), which gives us just enough, plus a little headroom. The third reason we go with eight bits is that computer storage is already organized in terms of bytes, where a byte is a unit of exactly eight bits. This quantity of eight bits is already so useful—for example, it's perfect for storing a character of type, which can be any of 256 letters and punctuation marks in a western alphabet—that it seems a cosmic coincidence that a byte is also the perfect amount of memory to encode tone levels for the human visual system. Engineers love those cosmic coincidences!

Millions of Colors

So *8-bit encoding*, with its 256 tone levels per channel is the minimum number of bits we want to store per channel. With RGB images, storing eight bits for each of the three channels gives us 24 bits total (which is why many people use the terms "8-bit color" and "24-bit color" interchangeably to mean the same thing). The number of colors encodable with 256 tone levels in each of three channels is $256 \times 256 \times 256$, or (if you pull out your calculator) about 16.8 million colors! Quite a lot of encodable colors for our 24 bits (or three little bytes) of storage!

Although this basic 3-channel, 8-bit encoding is the most common because it's based on human capabilities, we can easily expand it as

needed to encode more colors for devices other than the human eye, either by adding channels or by increasing the number of bits we store for each channel. For example, when we're preparing an image for a CMYK printer, we increase the number of channels from 3-channel to *4-channel encoding*, not because we need more encodable colors (in fact, we need fewer) but because it's natural to dedicate a channel to each of the four inks.

Similarly, we often go from 8-bit to *16-bit encoding* when saving images captured with a scanner capable of discerning more than 256 levels of RGB (the so-called "10-bit," "12-bit," and "14-bit" scanners—although, because we store files in whole bytes, there are no 10-bit, 12-bit, or 14-bit files, only 8-bit or 16-bit files).

A key point to remember is that this is all talking about the number of *encodings*, the set of numeric color definitions we have available. But just as in the San Francisco Bay Area there are far more telephone numbers than there are actual telephones, with computer color the number of *encodable* colors far exceeds the number of *reproducible* colors. In fact, it far exceeds the number of *perceivable* colors. And even if we make devices such as high-end scanners that can "perceive" more tone levels than the human eye, we can always expand our encoding model to handle it. All that matters is that each perceivable color has a unique encoding, so there are always more encodable colors than we need—just as the phone company must ensure that every telephone has a unique telephone number, so it had better have more telephone numbers than it really needs.

We make this point because it's a key step to understanding the difference between colors as abstract numbers and how those numbers are actually rendered as colors by "Real World" devices—printers, monitors, scanners, etc. When you look at how those numbers are actually interpreted by a device, the number of actual "Real World" colors, drops dramatically! (See the sidebar "Color Definitions and Colors.")

So while it's useful to understand how the numbers work—why we see numbers like 256 or 16.8 million crop up everywhere—don't forget that they're just numbers … until they're interpreted by a color device as colors.

In the next section we'll look at what gives the numbers a precise interpretation as colors. These are the analog parts of our color devices, the things that color management systems need to measure to know how to turn number management into color management.

Color Definitions and Colors

Lots of people confuse the number of color definitions with the number of colors. For example, we mentioned that colors intended for a CMYK printer are naturally encoded in four channels. Does 8-bit CMYK really represent 256 × 256 × 256 × 256 or 4.3 billion encodable colors?! Theoretically, yes. Four arbitrary channels produce 4.3 billion encodings, but when we assign the interpretations C, M, Y, and K to those four channels, we realize that the fourth channel (K) doesn't seem to add many colors. In fact, many of the CMYK encodings represent the same color. For example, 50C, 50M, 50Y, 0K theoretically encodes the same shade of gray as 0C, 0M, 0Y, 50K, so there's a lot of redundancy. Now one could argue that the extra K values contribute additional tonal levels to the CMY channels, but this starts to get way more complicated than we need to get here. Let's just say that the total number of encodable colors in 8-bit CMYK is far less than 4.3 billion.

As another example, we mentioned that there are scanners that claim to be able to see far more than the 256 tone levels encodable in eight bits. These tout 10-bit, 12-bit, or even 14-bit capability. Many people confuse this with the scanner's *dynamic range*, the range from brightest white to darkest dark in which the scanner can distinguish tonal variation reliably. Some scanner vendors may actually claim that these "high-bit" scanners give a larger dynamic range than 8-bit ones. This is nonsense—the dynamic range is an analog limitation of the capture device and has nothing whatsoever to do with bit depth. Higher bit depths simply allow us more editing flexibility by slicing the device's dynamic range into more discrete steps. You can think of dynamic range as the height of a staircase, and bit depth as the number of steps that staircase contains. Obviously, if we want to keep the steps as small as possible (which we do, to avoid posterization or banding), a higher dynamic range needs more steps than a smaller one, but there's no direct relationship between the two.

Why the Numbers Vary

In the coming chapters we'll look at the specifics of measuring the behavior of display devices (monitors), input devices (scanners and digital cameras), and output devices (printers and proofing systems), but here we'll look briefly at the basic parameters that vary from device to device.

All devices vary in certain basic parameters. These are the things you'll measure if you are making your own profiles, and which must remain stable for your color management system to work effectively.

The three main variables are:

- The color and brightness of the colorants (**primaries**).
- The color and brightness of the **white point** and **black point**.
- The **tone reproduction characteristics** of the colorants.

These concepts aren't new to color management or unique to digital devices. They're all variables introduced by analog components like inks on paper, phosphors and analog voltages in monitors, and filtered sensors in scanners. While digital components rarely vary much, analog components vary a great deal in design, manufacturing, and condition.

Colorants (Primaries)

The first, most obvious factor that affects the color a device can reproduce are the colorants it uses to do so. On a monitor, the primaries are the phosphors. In a scanner or digital camera, the primaries are the filters through which the sensors see the image. In a printer, the primaries are the process inks, toners, or dyes laid down on the paper, but, because the subtractive color in CMYK printers is a bit more complicated than the additive color in RGB monitors, we usually supplement measurements of the primaries with measurements of the secondaries (the overprints—Magenta+Yellow, Cyan+Yellow, and Cyan+Magenta) as well (see Figure 2-4).

The exact color of the colorants determines the range of colors the device can reproduce. This is called the color *gamut* of the device. We care not only about the precise color of the primaries, but also how bright they are. In technical terms we often refer to the *density* of the primaries, which is simply their ability to absorb light.

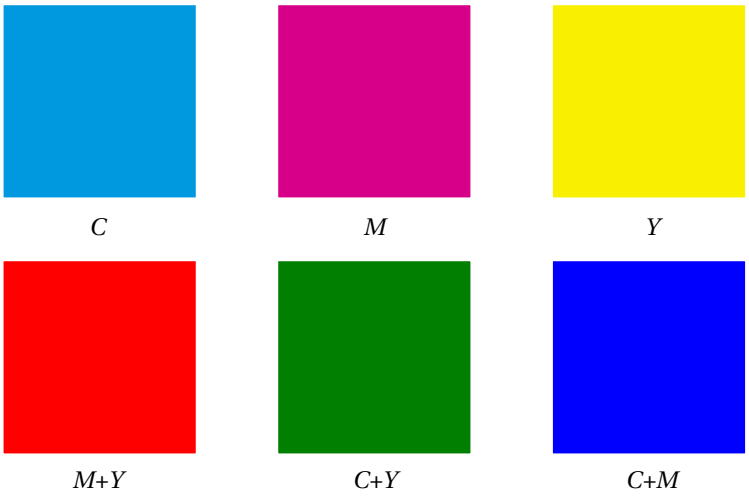


Figure 2-4
Subtractive primary and secondary colors

White Point and Black Point

Besides the primaries, the other two points that define the gamut and hence need to be measured and monitored in a device are the *white point* and *black point*. Books (and people) often talk about the white point and black point in very different terms: with the white point, they're usually concerned with the *color* of white, while with the black point they're more concerned with the *density* (the darkness) of black. In fact, we can talk about both color and density of either the white point or the black point—the difference is only a matter of emphasis. With the white point, the color is more important than its density, and with the black point, the density is more important than its color.

The color of the white point is more important than its density because the eye uses the color of this white as a reference for all other colors. When you view images, the color of white on the monitor or the color of the white paper on a printed page affects your perception for all the other colors in the scene. This *white point adaptation* is an instantaneous and involuntary task performed by your eye, so the color of white is vital. This is why, as we'll see in Chapter 6, we often sacrifice brightness during monitor calibration to get the color of the white point correct. Similarly, when looking at a printed page, it's important to remember that the color of the white point is determined as much by the light that illuminates the page as by the paper color itself.

With black point the emphasis shifts towards density as a more important variable than color. This is because the density of black determines the limit of the *dynamic range*, the range of brightness levels that the device can reproduce. Getting as much dynamic range as possible is always important, as this determines the capacity of the device to render *detail*, the subtle changes in brightness levels that make the difference between a rich, satisfying image and a flat, uninteresting 'mistake.'

On a monitor, we try to calibrate so that we get just enough brightness differences near the bottom end to squeeze some extra detail out of our displayed shadows.

On a printer, we can improve both the color and the density of the black point by adding our friend K to the CMY colorants. Adding K lets us produce a more neutral black point than we could with our somewhat impure C, M, and Y inks, and using four inks rather than three gets us much denser (darker) blacks than we could using only C, M, and Y.

On a scanner, the software lets you change the density of the white and black points from scan to scan either manually or automatically, but for color management we need to use a fixed dynamic range, so we generally set the black and white points to the widest dynamic range the scanner can capture.

Measuring the color and density of the white and black points is usually one of the steps in preparing to use any device in a color management system. You also need to watch out for changes in these values over time, so that you know when to adjust either the device itself or your color management system accordingly.

Tone Reproduction Characteristics

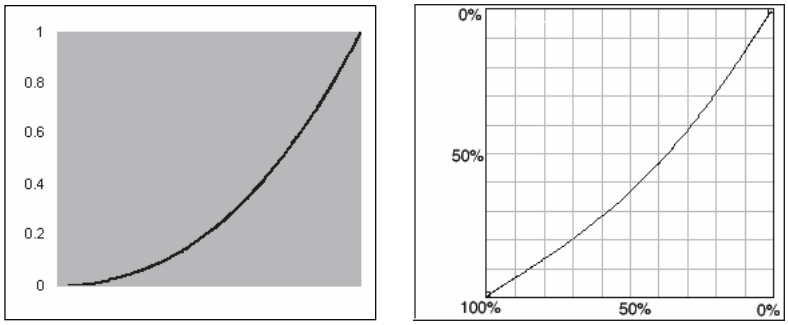
Measuring the precise color and density of the primaries, white point, and black point is essential, but these points only represent the extremes of the device: the most saturated colors, the brightest whites, and the darkest darks. To complete the description of a device, the color management system also needs to know what happens to the "colors between the colors" (to paraphrase a well-known desktop printer commercial).

There are several ways to measure and model devices' tone-reproduction characteristics. The simplest, called a *tone reproduction curve* (TRC), defines the relationship between input values and resulting brightness values in a device. Most analog devices have similar curves that show gain (increase) in the darkness levels that affect the midtones most and taper off in the hightlights and shadows. In monitors, scanners, and digital cameras this is called a *gamma curve*. Printers exhibit a slightly different *dot gain* curve, but the two are similar (see Figure 2-5).

Some printers have much more complicated tonal responses that can't be represented adequately by a simple curve. In these cases, we use a lookup table (LUT), which records representative tonal values from light to dark.

When you take measurements in the process of calibrating or profiling a device for color management, you're measuring the tone-reproduction characteristics of the device as well as the primaries and black/white points. You should try to get a feel for the things that can alter these tonal characteristics—such as changing to a different paper stock, or adjusting the contrast knob on your monitor—because when a device's tone-reproduction characteristics change, you'll have to adjust your color management system to reflect the changes.

Figure 2-5
Tone reproduction curves



gamma curve *dot gain curve*

Device-Specific Color Models

We call RGB and CMYK *device-specific* or *device-dependent* color models, because the actual color we get from a given set of RGB or CMYK numbers is specific to (it depends on) the device that's producing the color. Put simply, this means two things:

- The same set of RGB or CMYK numbers will produce different colors on different devices (or on the same device with different paper, if it's a printer or a press—see Figure 2-6).
- To produce the same color on different devices, we need to change the RGB or CMYK numbers we send to each device—see Figure 2-7.

The problems we face as a consequence are:

- How do we know what color the numbers in an RGB or CMYK file are supposed to represent? In other words, what does "255, 0, 0" mean? Yes, it means 'red,' but precisely what red? The red of your monitor, or Chris's or Fred's? The red sensor in your scanner or Bruce's digital camera?
- How do we know what RGB or CMYK numbers to send to a device to make it produce a desired color? In other words, even if we know precisely what 'red' we're talking about, what RGB numbers do we send to Chris's monitor, or what CMYK percentages do we send to Fred's Epson inkjet printer to reproduce that precise red … if it's possible to reproduce it at all?

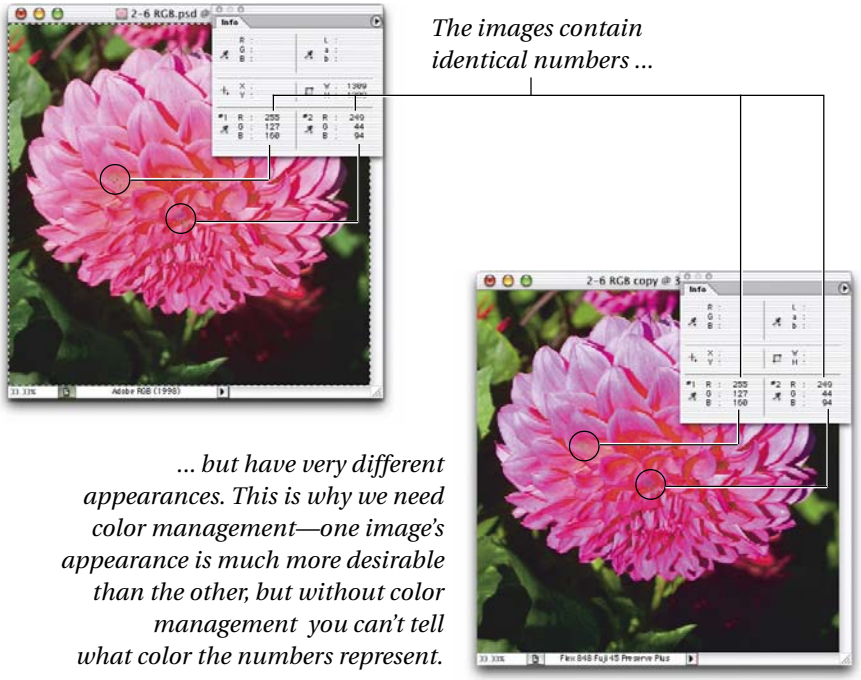


Figure 2-6
Same numbers, different color

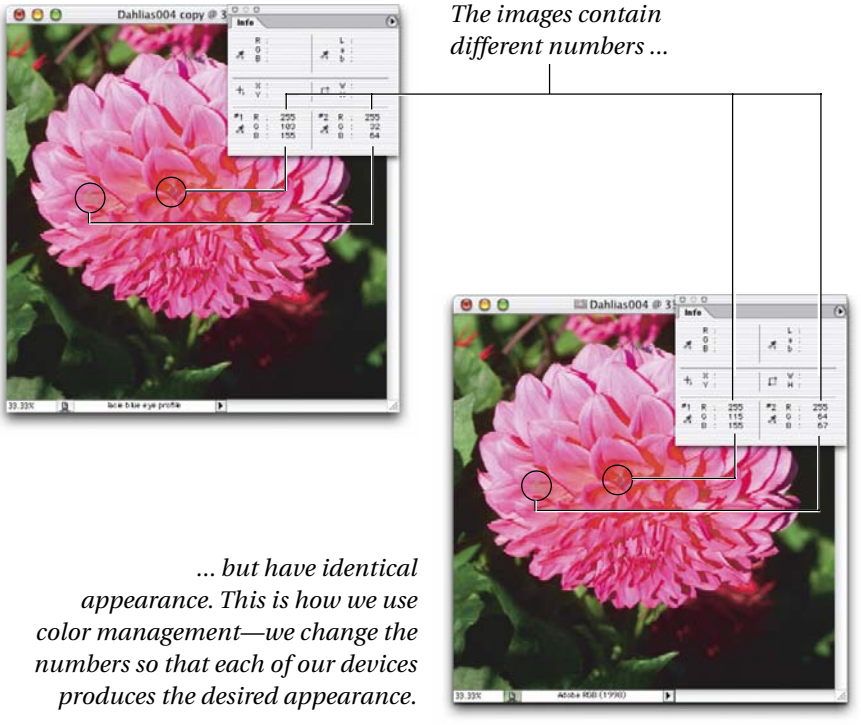


Figure 2-7
Same color, different numbers

Color management systems allow us to solve both problems by attaching absolute color meanings to our RGB and CMYK numbers. By doing so, the numbers cease to be ambiguous. Color management allows us to determine the actual color meaning of a set of RGB or CMYK numbers, and also lets us reproduce that actual color on another device by changing the numbers we send to it. But to do so, color management has to rely on a different kind of numerical model of color, one that's based on human perception rather than device colorants.

Device-Independent Color Models

Fortunately, we have several numerical models of color that are *device-independent*. Instead of using the numbers required to drive a particular device to produce color, device-independent color models attempt to use numbers to model human color perception directly.

The device-independent color models in current use are based on the groundbreaking work we mentioned in Chapter 1 by a body of color scientists and technicians known as the *Commission Internationale de l'Eclairage*, or CIE—in English, the name means "International Commission on Illumination"—and the CIE is the international standards body engaged in producing standards for all aspects of light, including color.

In 1931, the CIE produced a mathematical model of color with the formidable-sounding name CIE XYZ (1931). This model was unique in that it tried to represent mathematically the sensation of color that people with normal color vision would experience when they were fed a precisely defined stimulus under precisely defined viewing conditions. Since that original work was done, the CIE has produced a wild alphabet soup of color models with equally opaque names—CIE LCh, CIELUV, CIE xyY, CIELAB, and so on, all of which are mathematical variants of CIE XYZ.

You don't need to know the differences between the various models in order to use color management effectively. In fact, outside of measuring the colors a device produces as part of the profiling process, you needn't deal with any of the CIE models directly. But it *is* important to understand the distinction between device-dependent models like RGB and CMYK, and device-independent models like CIE XYZ and CIELAB.