International Color Consortium®

White Paper #35

Level: Advanced

Date: February 2012

Use of the parametricCurveType

In Version 4 of the ICC Profile Specification, parametricCurveType was introduced as an alternative to curveType for the representation of one-dimensional transfer functions. Either type can be used for the TRC tags or for the A-curves, B-curves, and M-curves embedded in lutAtoBType or lutBtoAType tags. In contrast to the older curveType, the new V. 4 type defines curves by closed-form expressions, rather than by 1D Look-Up Tables. Each curve is a scalar function of a scalar variable, but the expressions also involve constants, or *parameters*, which are encoded in the corresponding profile tags.

The V. 4 Profile Specification supports five different *Function Types*, requiring between 1 and 7 parameters. The Specification places no restrictions on the values of these parameters, aside from those imposed by the format. According to Clause 10.15, the parameters are encoded in the s15Fixed16Number format. Thus, the values can range from –32768 to almost +32768 (actually 32768 – 1/65536) in steps of 1/65536, or 0.0000152587890625. These restrictions are quite mild and, in practice, are hardly noticeable.

The parametricCurveType can be used to encode a wide variety of different functions. Profiles using the parametricCurveType can contain a wide range of possible parameter sets, and if care is not taken with their selection, some possible parameter sets can create computational problems for a CMM. The purpose of this chapter is to call attention to these problems, which can include divide-by-zero faults, complex roots, discontinuities in value and slope, and inversion ambiguities.

A CMM developer is faced with difficult decisions on how best to handle these problems. Different choices may be made by different developers, which can lead to inconsistent results among CMMs. Some of these choices may even interfere with legitimate choices made by profile creators.

The ICC Specification provides guidance on avoiding complex or undefined values in one particular case, through the selection of the parameter *d*. Other issues can arise in the implementation of the parametricCurveType, and this chapter aims to provide some further guidance in this context for both the profile creator and the CMM developer.

Fundamentals

Curves defined by the parametricCurveType are scalar functions of a scalar variable, which can be written:

$$y = f_n(x)$$

with $n = 0, 1, 2, 3,$ or 4 to designate the different Function Types. The domain of these functions is the unit interval [0, 1] — i.e., $0 \le x \le 1$. The range is also [0, 1], so that y will be clipped if the closed-form expressions produce a value that is outside that interval.

In some cases, the CMM will need to evaluate the inverse of a parametric curve. We will write the inverse of $f_n(x)$ as follows:

$$x = F_n(y)$$

Here, too, the domain and range are [0, 1], and clipping will occur if necessary. The inverse function will be needed when a TRC tag is evaluated in the PCS-to-device direction. (In the case of embedded curves, the profile creator provides the inverse in a separate tag.)

All five Function Types are variations on the basic power law:

$$y = x^{\gamma},$$

where γ ("gamma") is a constant parameter. They differ from one another in the use of various factors or offsets applied to x or y or in the partitioning of the domain into segments where other expressions are applied. The expressions are based on those typically used to model the transfer characteristics of CRT monitors or to define standard color spaces.

In mathematical terms, the parametric curves are normally used to define real-valued, continuous, smooth, monotonically non-decreasing functions mapping [0, 1] onto [0, 1]. Figure 1 shows a typical example: the electro-optical transfer function used in HDTV, as defined by Recommendation ITU-R BT.709. This is a Type 3 curve with a short linear segment at the origin, which becomes tangent to the power-law curve at $x = 0.081$.

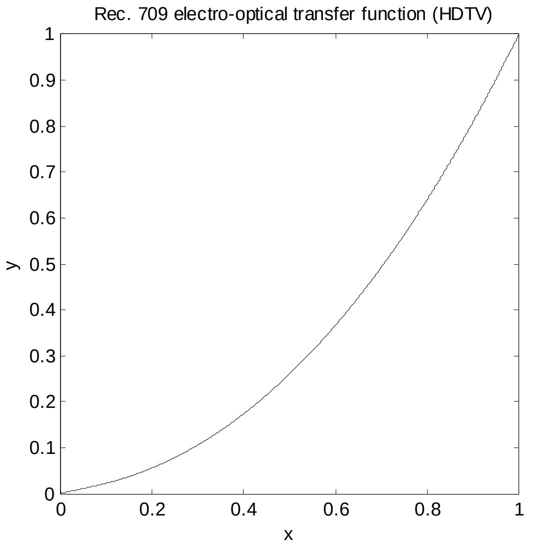

Figure 1: Well-behaved curve

However, any given parameter set may have a different effect and may, in some cases, result in a function exhibiting unwanted or even pathological behavior. See Figure 2 for an extreme example. This is a legitimate Type 4 curve in which the parameters have been assigned values such that the linear segment and a power-law segment are both clipped and are discontinuous with each other, with a large reversal. This curve has no inverse.

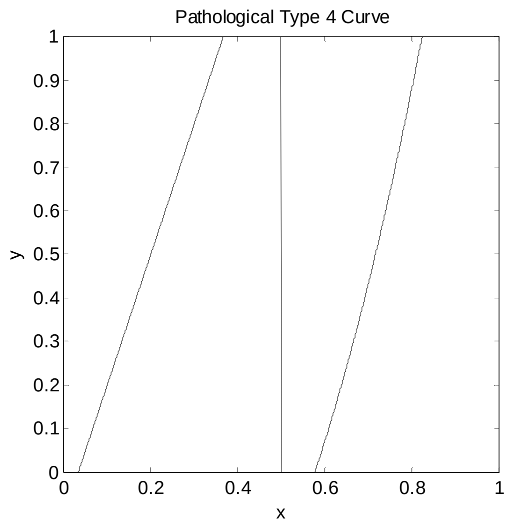

Figure 2: Badly-behaved curve

While this curve has been specially constructed to illustrate bad behavior and is unlikely to occur in practice, a CMM has to be prepared for every eventuality.

The power-law exponent

Typically, the exponent γ will be a small positive number. If so, x^γ will increase monotonically and will map 0 to 0 and 1 to 1. The inverse, $y^{1/\gamma}$, will have the same properties. See Figure 3 for examples.

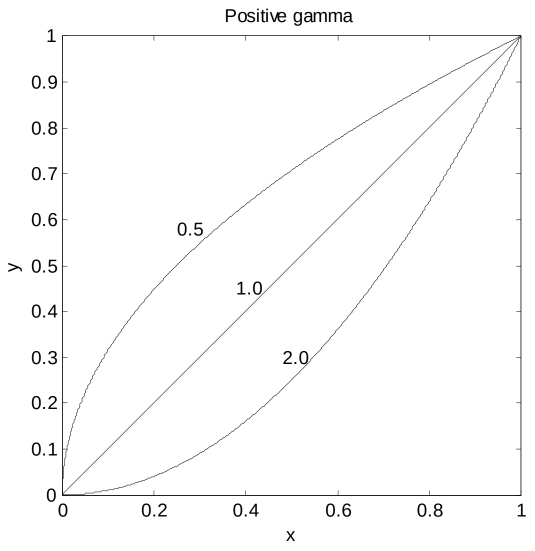

Figure 3: Power law with exponent $\gamma > 0$

If $\gamma < 0$, however, the function values for $0 < x < 1$ will decrease monotonically and will all be greater than one and, therefore, will all be clipped to 1. The function diverges as x approaches 0, so the CMM must be careful of overflow conditions (or even divide-by-zero faults) occurring before the clipping stage. The inverse is ambiguous. (The section on inverse evaluation below has more discussion on this point.) See Figure 4, where $\gamma = -0.5$.

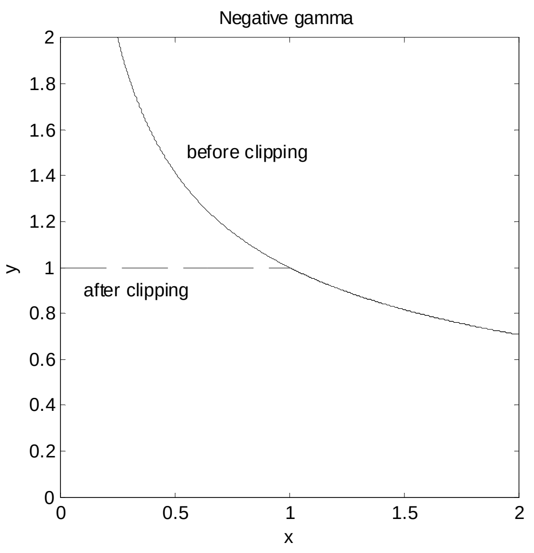

Figure 4: Power law with exponent $\gamma = -1/2$

If $\gamma = 0$, the function is identically equal to 1. The inverse is undefined, and even its exponent, $1/\gamma$, is undefined.

It is not obvious how (or even whether) a CMM should process data when γ is zero or negative. The profile is almost certainly corrupt or in error in such a case. There is clearly no point in defining a constant or decreasing function which will be clipped over the entire domain. The CMM developer may choose, in such a case, to reject the profile or to replace the parametric curve with the identity function, $y = x$ (which amounts to setting $\gamma = 1$), or some other default. Ideally profile creators should abide by the condition $\gamma > 0$, and CMM developers should treat any occurrences of negative and zero values as errors and take appropriate action.

The power-law argument

Function Type 0 is just the basic power law described above:

$$y = f_0(x) = x^\gamma$$

$$x = F_0(y) = y^{1/\gamma}$$

In the other types, the argument to the power law is not simply x, but a linear expression in x. In Types 1, 2, 3, and 4, the argument, which we will call s, takes the form:

$$s = ax + b,$$

where a and b are additional parameters. The power law is then s^γ. Since there is no practical restriction on the parameter values, s can take on any value as x varies between

0 and 1. If s is negative, s^γ can be imaginary or complex. (It will be real for integer γ, but γ cannot be restricted to integer values.) In such a situation, a CMM might choose to take the real part of the expression. Alternatively, it could take the absolute magnitude. It could arbitrarily set the expression to zero or one. Another option is simply to require s to be non-negative.

In Types 1, 2, 3, and 4, the domain is divided into two segments, and the power law is employed only in the higher segment. For instance, the definition of Type 1 is:

$$
\begin{aligned}
y = f_1(x) &= 0, && 0 \le x < -b/a \\
&= s^\gamma, && -b/a \le x \le 1
\end{aligned}
$$

In normal usage, a will be positive and b will be negative, so that the segment boundary, $-b/a$, occurs at a positive value of x. The function is identically zero in the lower segment (Figure 5).

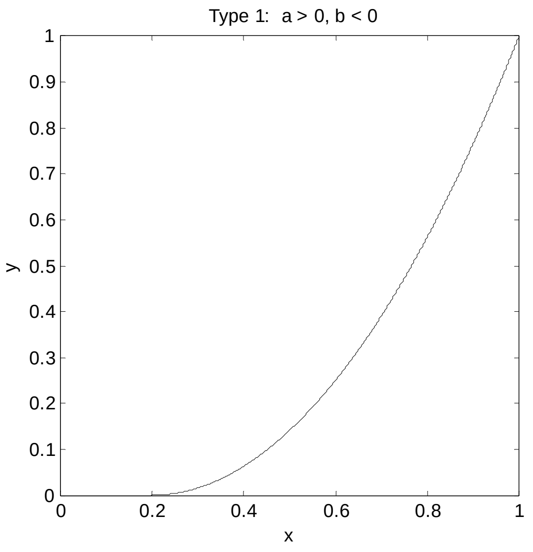

Figure 5: $f_1(x)$, with boundary at 0.2

The argument s is non-negative throughout the higher segment, where the power law is in effect:

$$(-b/a \le x) \;\Rightarrow\; (0 \le ax + b = s)$$

This conclusion is verified by multiplying both sides of the first inequality by a and then adding b; it holds only if $a > 0$, however. Indeed, the inequality is reversed for negative a. (And if $a = 0$, the segment boundary itself is indeterminate.) It seems reasonable to impose the condition a > 0 as a requirement, and negative and zero values can then be treated as errors, and the CMM can take appropriate action — for instance, by substituting $a = 1$ or some other default. But, as in the case of γ, the CMM developer needs to be reassured that profile creators do not have a legitimate use for negative or zero values.

Another reason to require that a be positive is that it compels the power-law function to be monotonically non-decreasing, which is the normal case.

Similarly, one might consider imposing the condition $b < 0$. However, this is probably not necessary. Positive values of b simply mean that the segment boundary will occur at negative x. The power law will be in effect over the entire domain, and there will be no lower segment with $y = 0$. In such a case, s will always be positive, and there will be no risk of complex values. (See Figure 6.)

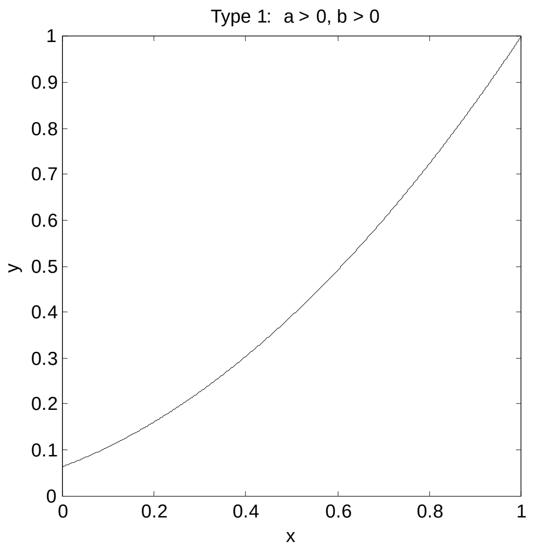

Figure 6: $f_1(x)$, with boundary at negative x

Note that, if the profile creator's intention is to have $y = 1$ just where $x = 1$, then the parameters should meet the condition $a + b = 1$.

Type 2 curves have a similar structure, with a segment boundary at $x = –b/a$, so the same analysis applies. However, in Types 3 and 4, the segment boundary is defined by an independent parameter, d. In the absence of restrictions, it is quite possible for d to be less than $–b/a$. There can then be values of x in the higher segment ($x > d$) at which $s = ax + b$ will be negative. The power law may then produce complex numbers. Figure 7 shows such an example. Here the absolute magnitude has been taken of the complex y-values in the interval between d and $–b/a$, but that is an arbitrary choice. A CMM may, just as arbitrarily, take the real part of the y-values or set them to zero. Since the occurrence of complex values is unlikely to be intentional, there is no universally correct way to handle them.

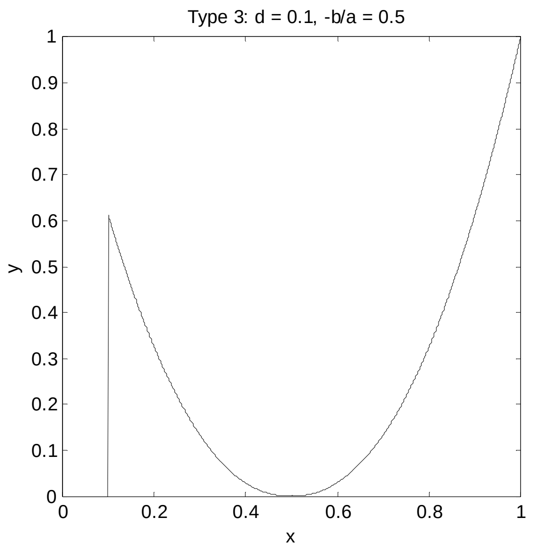

Figure 7: $f_3(x)$, with absolute magnitude of complex y-values

To prevent complex values occurring, the CMM could reject such a value of d and replace it with $d = -b/a$. (See Figure 8.) ICC recommends that $d \geq -b/a$ and profile creators should be aware that CMMs may impose this condition.

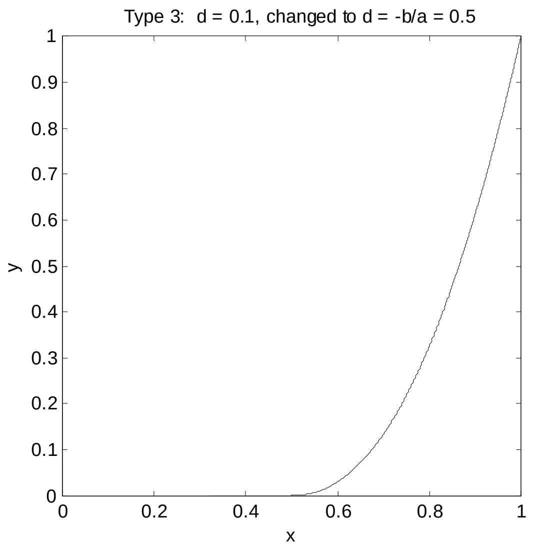

Figure 8: $f_3(x)$, with complex y-values eliminated

Continuity

Continuity across the segment boundary is guaranteed for Type 1 and Type 2. For Type 1 (see definition above), the value of s is exactly zero at the boundary, so the power law will yield a value of zero there. Below the boundary, in the lower segment, the function is identically zero, so $f_1(x)$ is continuous by definition.

Type 2 is similar to Type 1, with the addition of an offset:

$$y = f_2(x) = c, \qquad 0 \le x < -b/a$$
$$= s^\gamma + c, \qquad -b/a \le x \le 1,$$

where c is a constant parameter. The function is identically equal to c in the lower segment, and the power-law curve (in the upper segment) starts out at c, so $f_2(x)$ is also guaranteed to be continuous at the segment boundary. Figure 9 shows a typical example.

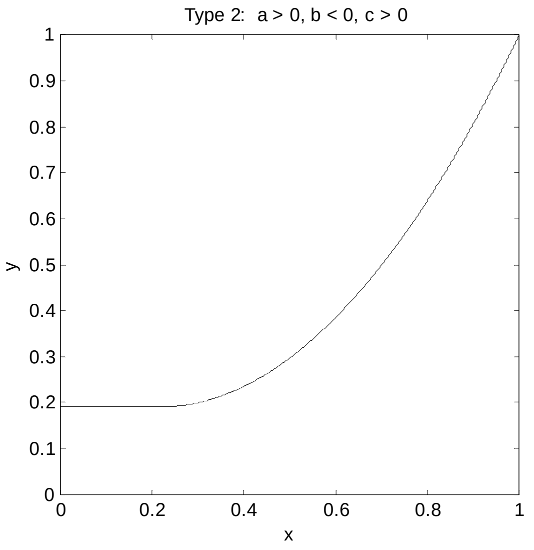

Figure 9: $f_2(x)$ with positive vertical offset

Types 3 and 4 do not enjoy a similar guarantee. Here is the definition of Type 3:

$$y = f_3(x) = cx, \qquad 0 \le x < d$$
$$= s^\gamma, \qquad d \le x \le 1$$

The function will be continuous at $x = d$ only if:

$$cd = (ad + b)^\gamma$$

Figure 10 shows a curve that violates this condition.

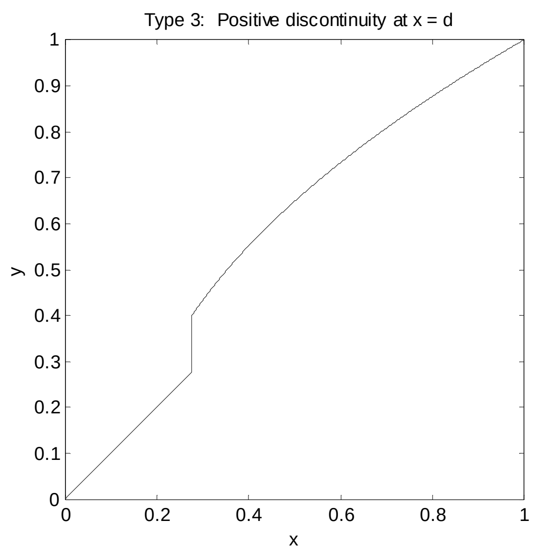

Figure 10: $f_3(x)$, discontinuous

Type 4 is defined as follows:

$$y = f_4(x) = cx + f, \qquad 0 \le x < d$$
$$= s^\gamma + e, \qquad d \le x \le 1$$

where e and f are additional parameters. The corresponding continuity condition is:

$$cd + f = (ad + b)^\gamma + e,$$

a relation involving 7 parameters. Figure 11 shows a curve violating this condition:

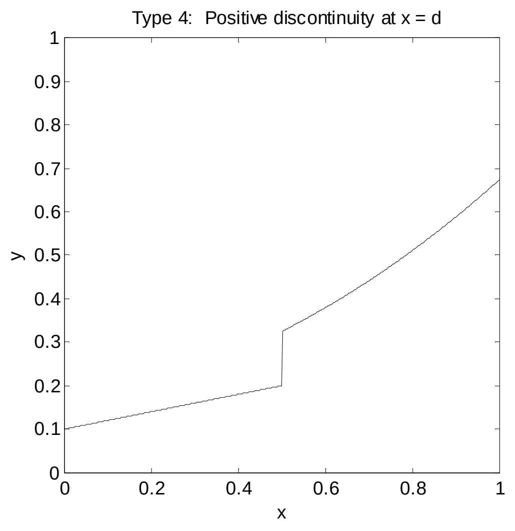

Figure 11: $f_4(x)$, discontinuous

Clearly, the profile creator should be aware of these conditions. In most, if not all, cases, the intention will be to encode a continuous function. Minor discontinuities may well occur through rounding of the parameter values, however, and they can be of either sign, so that reversals (non-monotonic behavior) will occur if the discontinuity is negative. Care is needed in the computation of the parameters if continuity problems are to be avoided.

Discontinuities in themselves do not cause computational problems for the CMM, at least for forward evaluation, so it may be best to leave this issue to the profile creator. The problems related to inverse evaluation of discontinuous curves will be discussed below in Section 6.0.

It is worth pointing out that arbitrary parameter values can lead to strange curve shapes. Discontinuities can be large enough that the function values go out of bounds and get clipped. If the discontinuities are negative, monotonicity can be dramatically violated in such cases as well. (Figure 2 is an example of this behavior.)

Smoothness

In most cases, it is not enough for the curve to be continuous at the segment boundary: It should also be smooth. That means that the first derivative must be continuous across the boundary.

In the case of Type 1 and Type 2, the first derivative is zero in the lower (flat) segment. The derivative of the power law at the boundary will also be zero if $\gamma > 1$, and the function will then be smooth. On the other hand, if $\gamma = 1$, the derivative will be equal to one, and if $\gamma < 1$, it will diverge; in these cases, the curve will take an abrupt bend at the segment boundary. These effects are evident in Figure 12.

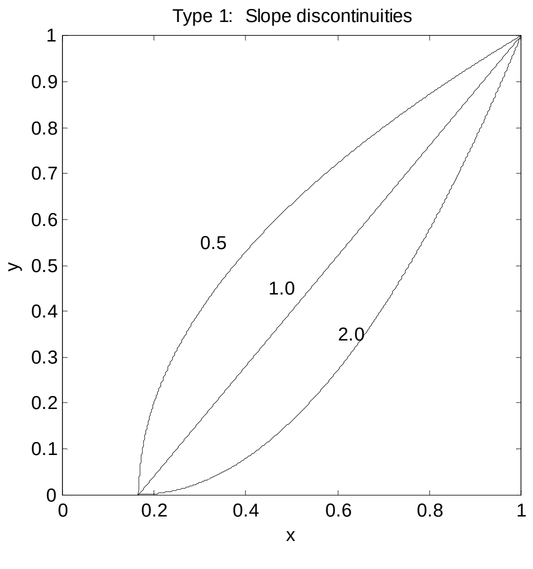

Figure 12: $f_1(x)$, showing effect of γ on smoothness

For Types 3 and 4, smoothness can be achieved only by satisfying the condition:

$$c = a\gamma(ad + b)^{\gamma-1},$$

as well as the continuity condition of Section 4.0 above.

In general, smoothness is a concern for the profile creator, not for the CMM.

Inverse evaluation

Curves in output profile lutAToBType and lutBToAType tags do not normally need to be inverted since the inverse of the lutAToBType is provided by the lutBToAType and vice versa; and similarly for MPE tags, where both forward and inverse functions are provided. Input profiles, where the lutBToAType may not be present, are not inverted in normal practice. Matrix/TRC profiles contain xTRCTag curves (where x is red, green or blue) which in v4 profiles can be defined as parametricCurveTypes, and such profiles commonly require to be inverted by the CMM in order to map PCS values back to the data encoding.

Several kinds of problems can arise when a parametric curve needs to be evaluated in the inverse direction. The most common problem occurs when there is a finite segment of the domain in which the function is constant, or *flat*. If $y = k$ (a constant) for all x in a subinterval $[x_1, x_2]$, there is no unique inverse at k, since any value in that subinterval is a legitimate candidate. While mathematically the inverse simply does not exist, computationally, we may say that the inverse is ambiguous (non-unique), and attempt to remove the ambiguity.

For instance, in Type 1 the function is identically zero in the lower segment $[0, –b/a]$. For $y > 0$, the inverse is simply:

$$x = F_1(y) = (1/a)[y^{1/\gamma} - b]$$

but at $y = 0$ the inverse is ambiguous: it can be any value in the range $[0, –b/a]$. Figure 13 shows the inverse of the curve of Figure 5. Some CMM developers may choose to return 0 for the inverse at $y = 0$, on the grounds that the function passes through zero at zero, and that that feature should be retained in the inverse. Others may choose to return $–b/a$, on the grounds of continuity. Still others may decide to split the difference and return $–b/(2a)$.

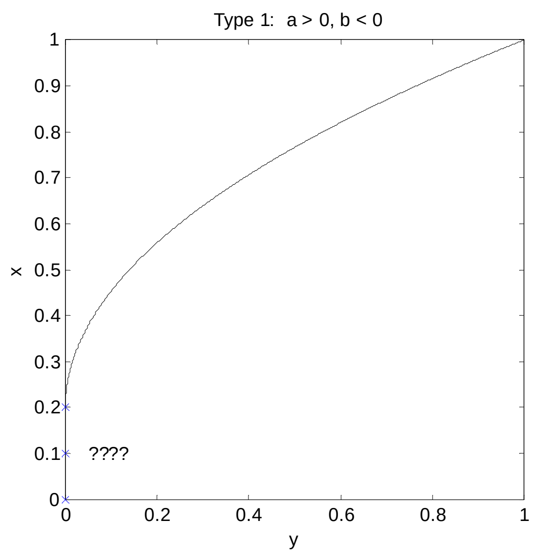

Figure 13: $F_1(y)$, ambiguous inverse at $y = 0$

Flat segments are explicit in the definition of Type 1 and Type 2. They can also occur in Type 3 and Type 4 if the slope parameter, c, in the lower segment has the value zero. Furthermore, flat segments can be produced by the clipping of out-of-bounds values to zero or one.

In a Type 1 or 2 curve, if $a + b > 1$, the argument s will reach 1 before $x = 1$. The curve will then go out of bounds at that point, and clipping will create a flat segment with $y = 1$ at the end of the unit interval.

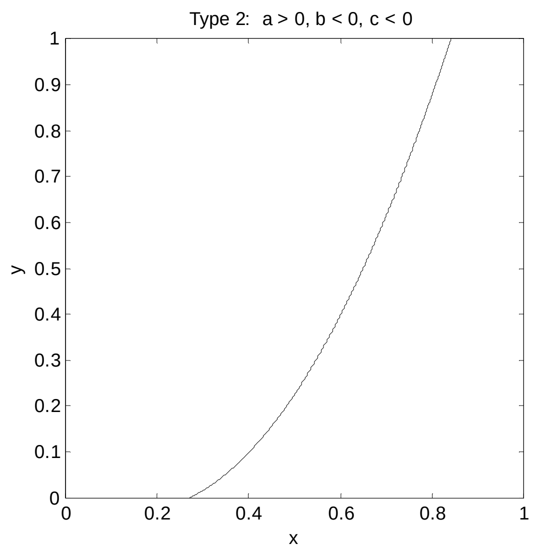

Figure 14: $f_2(x)$ with clipping

Further, suppose that the offset parameter c is negative in a Type 2 curve. Then the function values will be clipped to zero throughout the lower segment and for some portion of the upper segment. In effect, this will create a flat segment with $y = 0$. See

Figure 14 for an example of a curve with two flat segments, due to clipping at zero and at one.

Clipping can also affect Type 3 and Type 4 curves. In fact, since these curves can also be discontinuous (and non-monotonic) at the segment boundary, the functions can be clipped to zero or one in either the lower or the higher segment, or both. The "pathological" curve of Figure 2 is a Type 4 curve with these properties.

No matter how a flat segment arises, it presents an inversion ambiguity, which must be resolved by the CMM.

Other inversion problems can occur when the curve misses some values of *y* in the unit interval. For instance, if *c* is positive in a Type 2 curve, values of *x* in the lower segment will produce $y = c$, and values of *x* in the upper segment will produce values of $y > c$. No value of *x* will produce a value of *y* below *c*. (See Figure 9 for an example.) Thus, for $y < c$, the inverse is completely undefined. CMMs may well vary in their handling of this situation: some may return 0, others $-b/a$ or some other value. Similarly, in a Type 1 curve, if $a + b < 1$, the argument *s* will never reach 1 (for *x* in the unit interval). Values of $y > (a + b)^\gamma$ will never occur, and their inverse will be ambiguous. In this case, most CMMs would return 1 for these *y*-values. (See Figure 15 for a Type 1 curve with missing *y*-values at both ends of the range.)

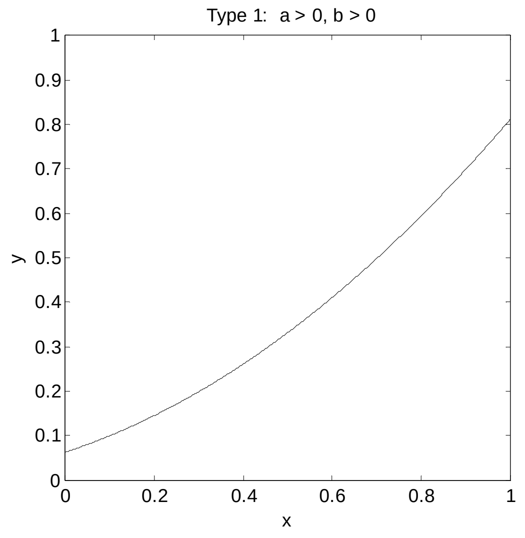

Figure 15: $f_1(x)$ with missing *y*-values, both high and low

Inversion problems can also result from non-monotonicity. For instance, in a Type 4 curve, a negative value of the slope parameter *c* will produce a decreasing function in the lower segment, followed by an increasing function in the upper segment. There can then be a range of *y*-values that are visited twice — once by each segment of the curve. See Figure 16 for an example. One possible resolution is to treat negative *c* as an error: the CMM can then adjust *c* (and also the offset parameter *f*) so as to remove the non-monotonicity.

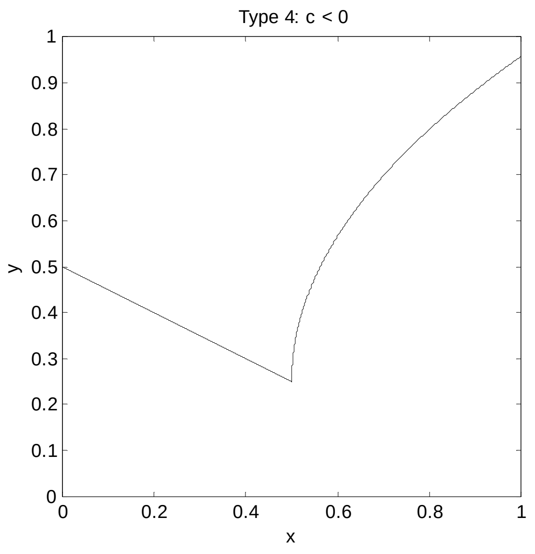

Figure 16: $f_4(x)$ with non-monotonicity

Non-monotonicity problems can also result from negative discontinuities, or reversals. If these are not treated as errors, there will have to be some other way to resolve the ambiguity in inversion. Figure 17 shows such an example.

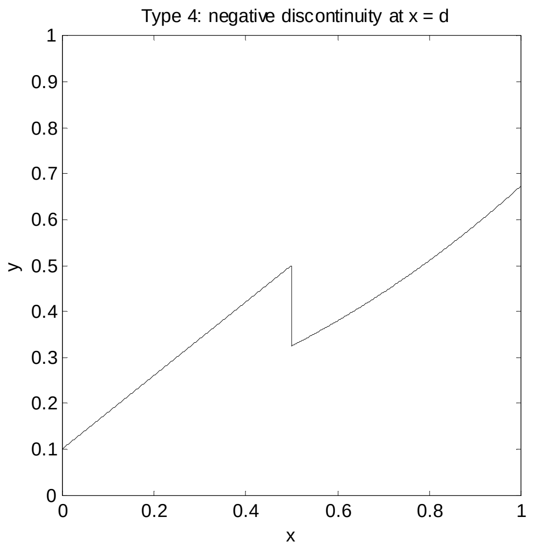

Figure 17: $f_4(x)$ with reversal

MPE curves

The multiProcessElementsType introduces new curve types to the ICC specification, including formula curves (defined similarly to the parametricCurveType) and sampled curve segments. The issues of imaginary numbers, continuity, and smoothness discussed above also apply to MPE segmented curves. However, continuity and smoothness may have different considerations since MPE curves are unbounded and segmented curves are most likely be used to perform the clipping that is required to set up inputs to following CLUT elements.,

Recommendations

The chief difficulty in making recommendations is the balance between the interests of the profile creators and those of the CMM developers. The functional forms designed for parametricCurveType may have originally been based on particular pre-existing use cases. The parameter sets that support those use cases satisfy various implicit constraints and conditions.

It would be possible for the ICC could make those constraints explicit, so that CMMs could be configured so as to impose these conditions as requirements for validity and to treat violations as errors. However, it is often the case that a standard specification can be used in creative ways that were not anticipated by the original designers. Wherever possible, profile creators should be free to explore and exploit these approaches without having their efforts rejected as errors by a CMM. The constraints obeyed by the original use cases may, therefore, be too strict as a guide for future profiles. From that point of view, it is better to err on the side of leniency and to impose only those constraints that are necessary to avoid the most serious problems.

It would probably be a mistake to over-analyze the situation — to enumerate all the possible combinations of parameters, to identify the interactions among them, and to formulate a complicated set of rules and exceptions to the rules. (For instance, negative s could be permitted when γ is an integer, but this is a rather uncommon — and probably pointless — exception to the general rule.) Complicated rules will be difficult to interpret and will tend to create confusion. It is better to adopt a relatively simple set of constraints, based on accepted general principles.

An important simplifying principle, in this regard, is that of *monotonicity*. If all curves were required to increase monotonically, one of the main causes of inversion ambiguity would disappear. However, it is clear that parametricCurveType has been defined so as to enable flat segments, either by design (as in Type 1 and Type 2) or through the mechanism of clipping, so strict monotonicity is too strong a constraint.

Hence, a weaker form of monotonicity is recommended — that the curves be *monotonically non-decreasing*. Flat segments would then be permitted (but only through the aforementioned mechanisms), and the resulting ambiguities in inversion must be dealt with. But a number of computational problems can be eliminated by adhering to this principle.

First, the condition $\gamma > 0$ can be imposed as a requirement. The power law will then be an explicitly increasing function of its argument. Furthermore, for Types 1 through 4, the condition $a > 0$ can be imposed, so that the argument itself is an increasing function of x. Accordingly, all the parametric curves will be monotonically increasing wherever their behavior is determined by a power law. Flat segments will occur only through clipping or in a segment explicitly created by partitioning the domain.

If the domain is partitioned, the lower segment is defined to be flat in Types 1 and 2, but it can have a non-zero slope in Types 3 and 4, according to the value of c. For the sake of consistency, $c = 0$ should be allowed, but the principle of weak monotonicity rules out negative values of c. This implies the condition $c \geq 0$ for Types 3 and 4.

In Types 3 and 4 there is also the possibility of violating monotonicity because of a negative discontinuity (or reversal) at the segment boundary. The corresponding condition that would be imposed to prevent such reversals is

$$cd \leq (ad + b)^{\gamma}$$

for Type 3 and

$$cd + f \leq (ad + b)^{\gamma} + e$$

for Type 4.

In addition to monotonicity, the CMM must adhere to the *reality principle*: no imaginary or complex values. This simply means that the CMM can require that the argument of the power law be non-negative. This is an issue for Types 3 and 4 and can be handled by imposing the condition:

$$ad + b \geq 0$$

Curves that violate these conditions will to some extent be “undefined” — i.e., implementation-dependent. A CMM could then treat such violations as errors. The response of the CMM may vary, depending on whether it is configured as an interactive tool, an OS service, a critical process embedded in a real-time system, or something else. It might reject the profile or the entire job in which it appears, exiting with an error message. Or it might silently substitute a generic profile known to satisfy the conditions. Or it might substitute default values for the problematic parameters.

Parameter substitutions

As an illustrative example, here is one way that the last alternative might be configured:

First, for all Function Types, if $\gamma \leq 0$, substitute $\gamma = 1$.

Next, for Types 1, 2, 3, and 4, if $a \leq 0$, substitute $a = 1$.

For Types 3 and 4, if $ad + b < 0$, substitute $d = -b/a$. Then determine whether the segment boundary is in the unit interval — i.e., $0 < d < 1$. If so, check for a negative discontinuity: For Type 3, if

$$cd > (ad + b)^{\gamma}$$

substitute $c = (1/d)(ad + b)^{\gamma}$. (This c will be non-negative.) For Type 4, if

$$cd + f > (ad + b)^{\gamma} + e$$

we may need to adjust 2 parameters: First, if

$$f > (ad + b)^{\gamma} + e$$

substitute

$$f = (ad + b)^{\gamma} + e$$

Then, whether f has been modified or not, substitute

$$c = (1/d)[(ad + b)^{\gamma} + e - f]$$

which will also be non-negative. If the segment boundary is not in the unit interval, the function will be defined by either the lower segment (if $d \geq 1$) or the higher segment (if $d \leq 0$), but not both. If it is the lower segment (that is, the linear expression), then the slope parameter c must still be checked for negativity; if $c < 0$, it can be reset to zero.

These procedures still allow positive discontinuities to occur at the segment boundary, as well as discontinuities of slope.

Inversion ambiguities

Even after applying the conditions above, the CMM will still have to deal with inversion ambiguities. While CMMs may well differ in the action taken when processing an "undefined" curve, they should be consistent in their processing of valid curves, in both the forward and inverse direction. Based on (weak) monotonicity and continuity as the simplifying principles, the following procedure is recommended:

If a flat segment results from clipping to $y = 0$, the inverse at that point should be defined as the upper endpoint of that segment in x. If a flat segment results from clipping to $y = 1$, the inverse at that point should be defined as the lower endpoint of the segment. If a flat segment is defined explicitly as the lower segment ($x < d$) of Types 2, 3, or 4, the inverse at that value of y should be defined as $x = d$.

An ambiguity arising from missed y-values can be handled as follows: If the curve begins with a non-zero value y_0 at $x = 0$, thus skipping over $y < y_0$, the inverse for those values is defined as $x = 0$. If the curve ends at a value $y_1 < 1$ at $x = 1$, thus skipping any values of $y > y_1$, the inverse of those values is defined as $x = 1$. If there is a (positive) discontinuity at $x = d$ (for Types 3 and 4), thus skipping over an interval in y, the inverse of any values in that gap is defined as $x = d$. (Negative discontinuities, as discussed above, result in "undefined" behavior and need not be considered further.)

Non-parametric curves

Note that parametricCurveType is not the only place in the ICC Specification where such issues can occur. For the sake of consistency, it is advisable to provide similar guidance for the interpretation of curveType. There are two issues here:

First, the 1D Look-Up Table (LUT) in a curveType tag may have a length of 1. This is an exceptional case, in which the single value is interpreted as γ in a simple power law. It is encoded as an unsigned u8Fixed8Number, so negative values cannot occur. However, γ could be zero, and that case should be considered as "undefined" for consistency with the treatment of the parametricCurveType outlined above.

Secondly (and more seriously), there can be inversion ambiguities in the curveType. The 1D LUT can have flat segments, and some rule is needed for defining the inverse at these points. The rule should be consistent with the one given under 'Inversion ambiguities' above, but it needs to handle a greater variety of cases: Flat segments in the LUT can occur anywhere, not just at the beginning or at the end of the unit interval. In addition to flat segments, reversals (violations of weak monotonicity) can also occur in the LUT; in such cases, the inverse should be designated as "undefined". Missing y-values can be handled in a manner similar to the rules described under 'Inversion ambiguities' above.