Dual Cubics

Jason Sanmiya
Fun Propulsion Labs at Google
February 27, 2015

Summary

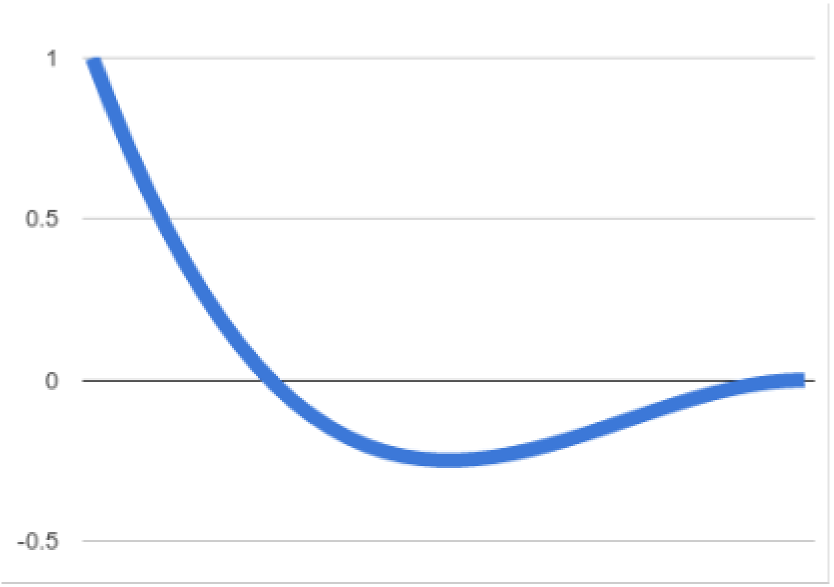
Splines are cruel. Intuitively, we imagine splines as nice, smooth paths between our control points. But between our control points, splines can do whatever they want.

For example, on the right, both splines have the same start and end values and derivatives. In a happy world, our spline would look like the spline on the bottom. But with a simple cubic, we overshoot the target, like the spline on the top. Both curves are valid, since they both respect the control points, but the bottom spline is more well-behaved.

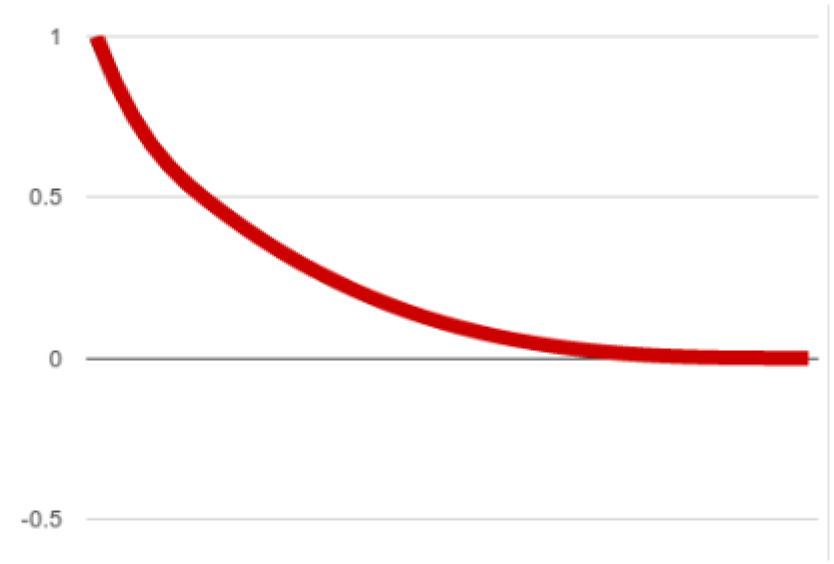
This paper proposes adding restrictions to cubic splines, to make them well-behaved. Then we create one curve composed of two well-behaved cubics, joined in the middle.

We present the math to join these cubics, to ensure that they are well-behaved, and to allow us to also specify second derivatives at the control points.

We call these curves "dual cubics".



above: *a simple cubic curve overshooting the target*
below: *a dual cubic curve acting more predictably*



Contents

Application

There is a smorgasbord of potential uses: data compression, audio libraries, physics engines, animation data, curve fitting, etc., etc.. Any time that both runtime speed and curve predictability is required, dual cubics could be a good choice.

Problems with Existing Solutions

The Problem with Simple Polynomial Splines

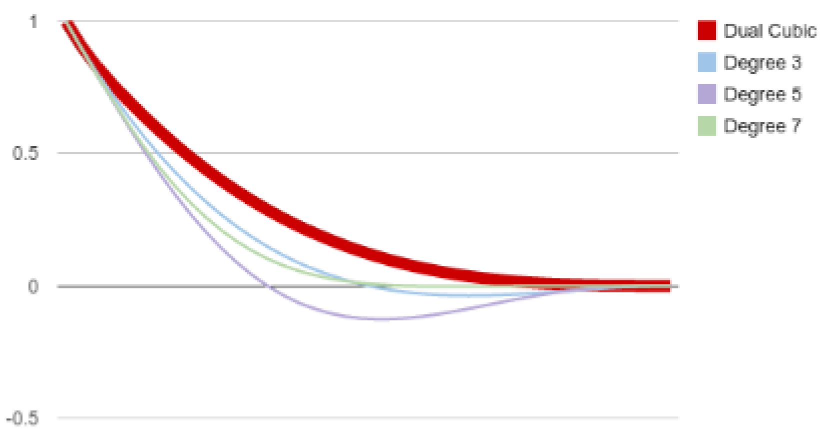
Simple splines tend to overshoot the target. You can see this in the graphs to the right. The problem is more pronounced as the slope difference increases.

Increasing the degree of the polynomial lessens the problem, but doesn't eliminate it fully. See Runge's phenomenon for interesting details.

Monotone cubic interpolation can decrease the problem, but at the cost of changing the slopes at your control points. Also, the curve often still has bumps in it.

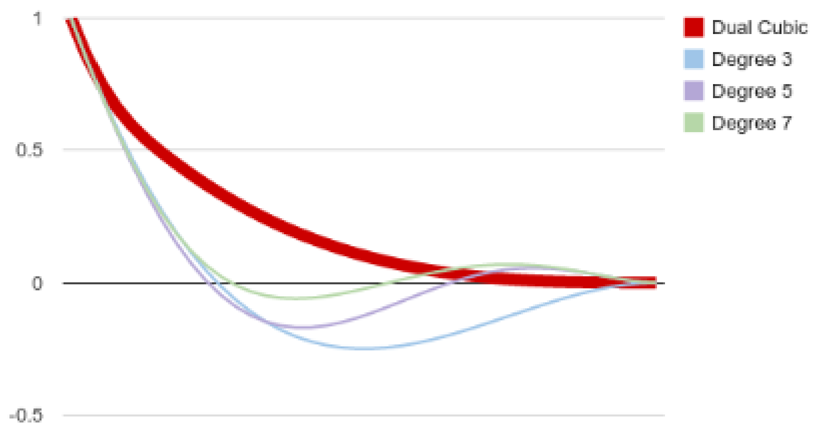
The problem is apparent when both upward and downward curvatures appear in the spline. We end up with an S-shaped, oscillating curve. We never want this. We want the curvature to change at most once, and only if required.

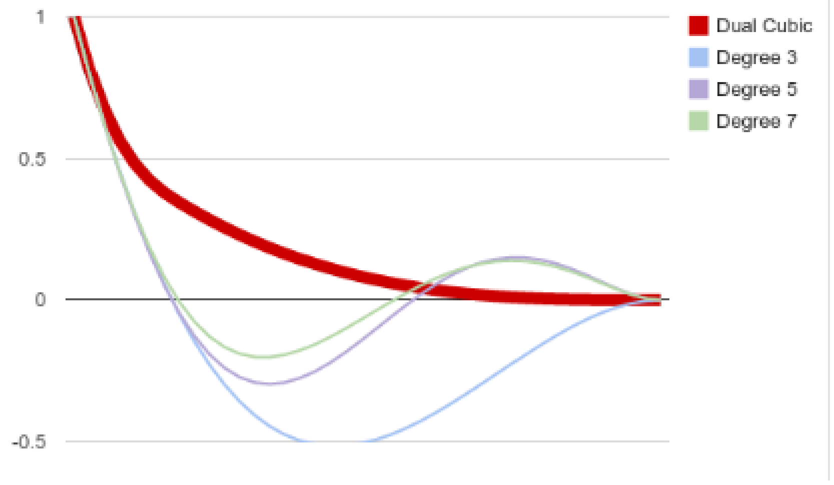
Dual cubic splines are composed of two cubics. They force each cubic to curve either upwards or downwards over their entire domain. This restriction makes the cubics behave much more predictably.



Simple polynomials of degree 3 (cubics), 5, and 7. All overshoot the target, or oscillate, when the initial slope is steep and the end slope is flat.

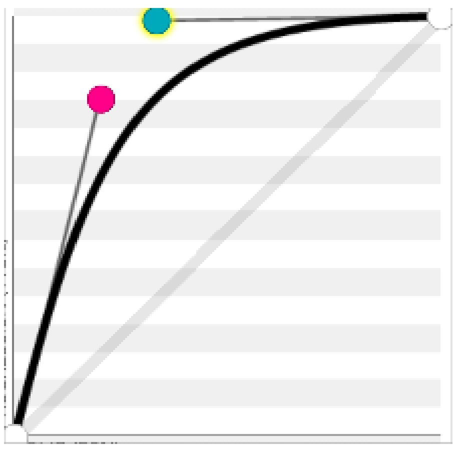
above: *initial slope of -4, width and height are 1*
immediate below: *initial slope of -6*
far below: *initial slope of -8*





The Problem with 2D Bezier Splines

1D Bezier splines are simple polynomials, and have the same overshoot problem described above. However, if we think of a graph as a 2D space, we can use 2D Bezier splines to construct 1D curves.



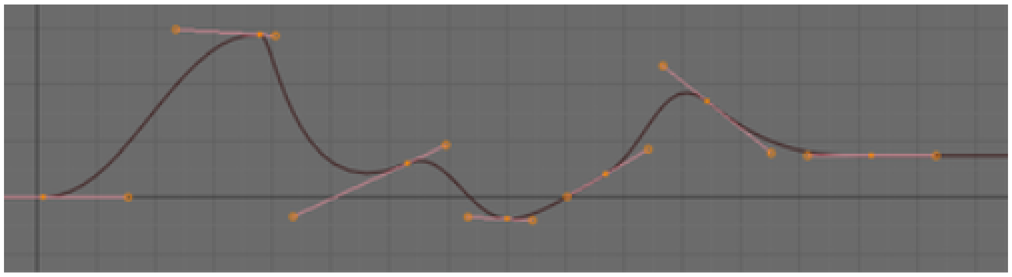
above: *A 1D graph treated as a 2D Bezier curve. The shape is always well-behaved because Beziers interpolate between their control points. However, there is no closed form way to calculate y from x.*

2D Bezier splines are perfect for creating curves that look just like we want them to look.

Beziers are interpolated splines. Cubic Beziers are the scalar sum of four control points. Bezier curves are guaranteed to be within the convex hull of their control points, so they are well behaved.

However, there is one big problem with using 2D Beziers as a 1D curve. For a given *x*, there is no closed-form way to calculate *y*. If the Bezier $B(t)$ runs over $t \in [0,1]$ then we have to use Newton's method to find the *t* that corresponds to *x*. Moreover, it's possible to construct Beziers for which Newton's method does not converge, so in the worst case we'll need to do a binary search. See this reference for the math.

Evaluation speed is critical for us, so 2D Bezier splines are, unfortunately, not an option.



above: *a series of 2D Bezier curves in the Blender spline editor. Programs like Blender choose ease of authorship over runtime speed. For us, runtime speed is paramount.*

Dual Cubic Splines

We define a spline that is composed of two, well-behaved cubic splines that are joined smoothly in the middle. For normal cubics, we are limited to specifying values and derivatives at the control points. We'll show how dual cubics allow us to specify second derivatives as well.

Uniform Curvature

A spline is well-behaved if it always curves upwards or it always curves downwards. That is, it avoids having an S-shape. We'll call this property *uniform curvature*.

More formally, the spline $f(x)$ has *uniform curvature* if $f''(x) \geq 0 \ \forall x \in [0,1]$ or $f''(x) \leq 0 \ \forall x \in [0,1]$.

Cubics

Cubics can be specified by values and derivatives at two points.

We use cubic splines in their simplest form over the domain $x \in [0,1]$.

$$f(x) = dx^3 + cx^2 + bx + a$$
$$f'(x) = 3dx^2 + 2cx + b$$
$$f''(x) = 6dx + 2c$$

Note: Bernstein polynomials are often used over the domain $x \in [0,1]$, but the math here is simplest with polynomials defined as above.

Given start and end values $f(0) = y_0, f(1) = y_1$ and derivatives $f'(0) = s_0, f'(1) = s_1$, the cubic coefficients $a, b, c,$ and d are uniquely determined.

$$a = f(0) = y_0$$
$$b = f'(0) = s_0$$

c and d are found via the control point at $x = 1$.

$$d + c + s_0 + y_0 = f(1) = y_1$$
$$3d + 2c + s_0 = f'(1) = s_1$$

solving the two equations gives,

$$c = 3(y_1 - y_0) - 2s_0 - s_1$$
$$d = s_0 + s_1 - 2(y_1 - y_0)$$

Decomposition into Two Cubics

Our control points are at $x = 0$ and $x = 1$. At the control points we specify the value, the first derivative, and the second derivative.

$$f(0) = y_0,\ f'(0) = s_0,\ f''(0) = w_0$$
$$f(0) = y_0,\ f'(0) = s_0,\ f''(1) = w_1$$

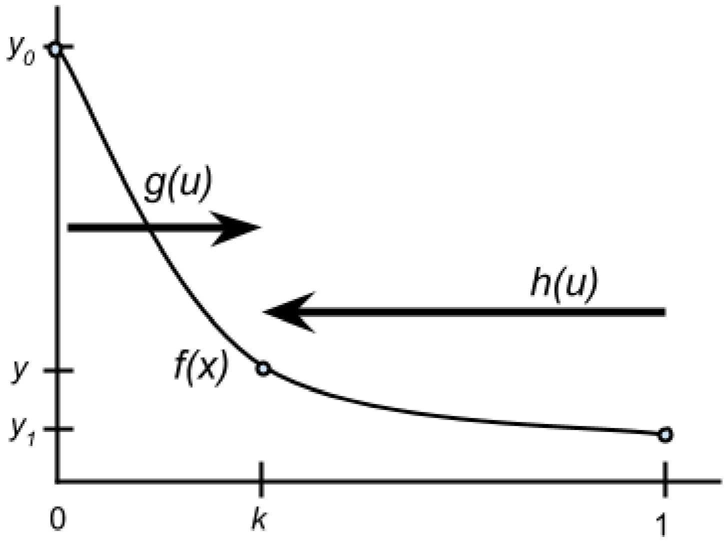
Our goal is to create a smooth function $f(x)$ for $x = 0..1$.

We decompose $f(x)$ into two splines $g(u)$ and $h(u)$, defined for $u = 0..1$,

$$f(x)\ = g(\tfrac{x}{k})\ for\ 0 \le x \le k$$
$$f(x) = h(\tfrac{1-x}{1-k})\ for\ k \le x \le 1$$

where $k \in (0, 1)$ is the joining point.

Picture this as g starting at $x = 0$ and growing to the right, and h starting at $x = 1$ and growing to the left.



Equivalently, we can formulate g and h, and their derivatives, in terms of *f*.

$$g(u)\ = f(ku)\ for\ 0 \le u \le 1$$
$$g'(u)\ = kf'(ku)\ for\ 0 \le u \le 1$$
$$g''(u)\ = k^2 f''(ku)\ for\ 0 \le u \le 1$$

$$h(u) = f(1 - ju)\ for\ 0 \le u \le 1,\ where\ j = 1 - k$$
$$h'(u) = \ -jf'(1 - ju)\ for\ 0 \le u \le 1$$
$$h''(u) = \ j^2 f''(1 - ju)\ for\ 0 \le u \le 1$$

The control points define g and h at 0.

$$g(0) = f(0) = y_0$$
$$g'(0) = kf(0) = ks_0$$

$$h(0) = f(1) = y_1$$
$$h'(0) = -jf'(1) = -js_1$$

The midpoint $x = k$, where g and h meet, must have the same value $f(k) = y$ and derivative $f'(k) = s$,

$$g(1) = f(k) = y$$
$$g'(1) = kf'(k) = ks$$
$$h(1) = f(k) = y$$
$$h'(1) = -jf'(k) = -js$$

Now that we have the start and end values and derivatives for g and h, we can define them as described in “Cubics” above,

$$g(u) = [k(s_0 + s) - 2(y - y_0)]u^3 + [3(y - y_0) - k(2s_0 + s)]u^2 + ks_0 u + y_0$$
$$h(u) = [-j(s_1 + s) - 2(y - y_1)]u^3 + [3(y - y_1) + j(2s_1 + s)]u^2 + js_1 u + y_1$$

From this description of g and h, we also know that,

$$g''(u) = 6[k(s_0 + s) - 2(y - y_0)]u + 2[3(y - y_0) - k(2s_0 + s)]$$
$$h''(u) = 6[-j(s_1 + s) - 2(y - y_1)]u + 2[3(y - y_1) + j(2s_1 + s)]$$

So,

$$g''(0) = 6(y - y_0) - 2k(2s_0 + s)$$
$$h''(0) = 6(y - y_1) + 2j(2s_1 + s)$$

And from the original definition of g and h we know,

$$g''(0) = k^2 f''(0) = k^2 w_0$$
$$h''(0) = j^2 f''(1) = j^2 w_1$$

Relating these two definitions of $g''(0)$ and $h''(0)$, and substituting $j = 1 - k$ creates two linear equations in s and y,

$$k^2 w_0 = 6(y - y_0) - 2k(2s_0 + s)$$
$$(1 - k)^2 w_1 = 6(y - y_1) + 2(1 - k)(2s_1 + s)$$

Solving these two equations for y and s gives (see Appendix 1 for details),

$$s = 3(y_1 - y_0) - 2(ks_0 + (1 - k)s_1) - \tfrac{1}{2}(k^2 w_0 - (1 - k)^2 w_1)$$
$$y = y_0 + k(y_1 - y_0) + k(1 - k)(\tfrac{2}{3}(s_0 - s_1) + \tfrac{1}{6}(kw_0 + (1 - k)w_1))$$

This gives us a complete parameterization of g and h (and hence f) from our control point constants $y_0,\ y_1,\ s_0,\ s_1,\ w_0,\ and\ w_1$.

The variable $0 < k < 1$ is still free for us to manipulate at our discretion.

Ensuring Well-Behaved

By varying k we'll try to ensure that g and h have uniform curvature. Note that this cannot always be done: In extreme cases we'll need to adjust w_0 and w_1, too.

By substituting s and y into $g''(1)$ and $h''(1)$ we get the following quadratics in k (see Appendix 2 for details).

$$r_g(k) = g''(1) / k = \; w_d k^2 + (4s_d - w_0 - 2w_1)k + 6y_d - 2s_0 - 4s_1 + w_1$$
$$r_h(k) = h''(1) / (1-k) = \; -w_d k^2 \; + (-4s_d + 3w_1)k - 6y_d + 6s_1 - 2w_1$$

where,

$$y_d = y_1 - y_0$$
$$w_d = w_1 - w_0$$
$$s_d = s_1 - s_0$$

Since $g''(u)$ is linear, g has uniform curvature iff $Sign(g''(1)) = Sign(g''(0))$. Which implies,

$$Sign(r_g(k)) = Sign(g''(1) / k) \; = \; Sign(g''(1)) = Sign(g''(0)) = Sign(k^2 w_0) = Sign(w_0)$$

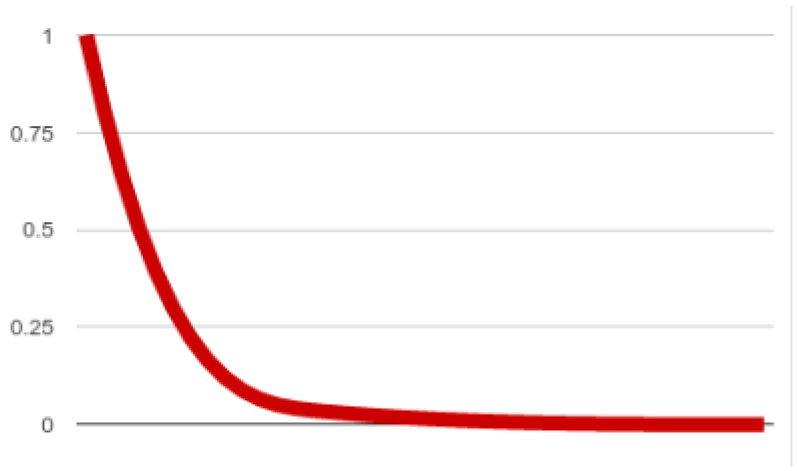
And similarly, h has uniform curvature iff $Sign(r_h(k)) = Sign(w_1)$.

$r_g(k)$ and $r_h(k)$ are quadratic, so they have at most two intervals that satisfy the requirements $Sign(r_g(k)) = Sign(w_0)$ and $Sign(r_h(k)) = Sign(w_1)$. If there is any value $k \in (0, 1)$ in the intersection of those intervals, then f will be well behaved when we use that k.

Estimating Good Second Derivatives

Most of the time, you'd rather not specify second derivatives explicitly. They're a little bit too abstract of a concept. You'd just want some reasonable default value to be used.

In these cases, we'd like to automatically calculate some second derivatives for the control points. The second derivatives that we choose should make the curve as smooth as possible. Poor second derivatives will result in an elbow-like shape where g and h join.



above: *Poor second derivatives can create an elbow-like shape where* g *and* h *join*

Extreme Second Derivatives

To get a sense of the valid range of second derivatives, we can consider the extreme case where $g''(1) = 0$ and $h''(1) = 0$. When the second derivatives are zero at $u = 1$, the second derivatives must be their largest at $u = 0$.

Assuming we know k, we have two linear equations in two unknowns, w_0 and w_1.

$$0 = g''(1) / k = w_d k^2 + (4s_d - w_0 - 2w_1)k + 6y_d - 2s_0 - 4s_1 + w_1$$
$$0 = h''(1) / (1 - k) = -w_d k^2 + (-4s_d + 3w_1)k - 6y_d + 6s_1 - 2w_1$$

Solving gives the extreme values for w_0 and w_1 (see appendix for details).

$$w_0^{max} = s_d + \tfrac{1}{k}(3y_d - 2s_0 - s_1)$$
$$w_1^{max} = \tfrac{1}{k-1}(s_d k + 3y_d - 3s_1)$$

Good second derivatives are somewhere between 0 and these extremes. We require further investigation to figure out just where the best values are. Also, it's not obvious which value of k we should pick.

Approximating *k*

Above we show how to find extreme values for w_0 and w_1, but these formulas assume that we know k. Unfortunately, we can only calculate the valid k range once we have w_0 and w_1. So to break the circle, we use a heuristic to calculate an approximate k first.

Generally, we want k to be closer to the steeper node. One approximation that appears to work well is this described below. First, define the *steepness* of a slope *s* as,

$$Steepness(s) = max(0, log|s|)$$

Then

$$percent_{steep} = |Steepness(s_0) - Steepness(s_1)| / steepness_{max}$$
$$k_{approx} = Lerp(0.5, k_{steep}, percent_{steep})$$

Where

$steepness_{max} = 4$, and

$k_{steep} = 0.1$ when $Steepness(s_0) > Steepness(s_1)$ and $k_{steep} = 0.9$ otherwise

are constants that work fairly well.

Approximating Second Derivatives from Extremes and *k*

Given k_{approx} above, we can calculate w_0^{max} and w_1^{max}. From there, we can set

$percent_0 = percent_{steep}$ if $Steepness(s_0) > Steepness(s_1)$, and $1 - percent_{steep}$ otherwise

Then,

$$w_0^{estimate} = percent_0\, w_0^{max}$$
$$w_1^{estimate} = (1 - percent_0)\, w_1^{max}$$

give reasonably well-behaved curves.

References

A Primer on Bézier Curves
Cubic-Bezier Visualizer
A good summary of the math of Beziers

Appendix

1. Solving for *s* and *y*

$$k^2w_0 - (1-k)^2w_1 = \; -6y_0 - 2k(2s_0 + s) + 6y_1 - 2(1-k)(2s_1 + s)$$
$$k^2w_0 - (1-k)^2w_1 + 6y_0 - 6y_1 + 4ks_0 + 4(1-k)s_1 = \; -2ks - 2(1-k)s$$
$$k^2w_0 - (1-k)^2w_1 + 6(y_0 - y_1) + 4(ks_0 + (1-k)s_1) = \; -2s$$
$$s = 3(y_1 - y_0) - 2(ks_0 + (1-k)s_1) - \tfrac{1}{2}(k^2w_0 - (1-k)^2w_1)$$
$$s = 3(y_1 - y_0) - 2Lerp(s_1, s_0, k) - \tfrac{1}{2}(k^2w_0 - (1-k)^2w_1)$$

$$k^2w_0 = 6(y - y_0) - 4ks_0 - 2k(3(y_1 - y_0) - 2(ks_0 + (1-k)s_1) - \tfrac{1}{2}(k^2w_0 - (1-k)^2w_1))$$
$$6(y - y_0) \; = k^2w_0 + 4ks_0 + 2k(3(y_1 - y_0) - 2(ks_0 + (1-k)s_1) - \tfrac{1}{2}(k^2w_0 - (1-k)^2w_1))$$
$$6(y - y_0) \; = k(kw_0 \; + 4s_0 + 6(y_1 - y_0) - 4(ks_0 + (1-k)s_1) - (k^2w_0 - (1-k)^2w_1))$$
$$6(y - y_0) \; = k(6(y_1 - y_0) \; + 4s_0 - 4ks_0 - 4s_1 + 4ks_1 + kw_0 - k^2w_0 + (1-k)^2w_1)$$
$$6(y - y_0) \; = k(6(y_1 - y_0) \; + 4s_0(1-k) - 4s_1(1-k) + kw_0(1-k) + (1-k)^2w_1)$$
$$6y \; = 6y_0 + 6k(y_1 - y_0) \; + k(1-k)(4s_0 - 4s_1 + kw_0 + (1-k)w_1)$$
$$y = y_0 + k(y_1 - y_0) \; + k(1-k)(\tfrac{2}{3}(s_0 - s_1) + \tfrac{1}{6}(kw_0 + (1-k)w_1))$$
$$y = Lerp(y_0, y_1, k) + k(1-k)(\tfrac{-2}{3}(s_1 - s_0) + \tfrac{1}{6}Lerp(w_1, w_0, k))$$

2. Substituting *s* and *y* into *g''(1)* and *h''(1)*

$$g''(1) \; = 6[k(s_0 + s) - 2(y - y_0)] + 2[3(y - y_0) - k(2s_0 + s)]$$
$$g''(1) \; = 6ks_0 + 6ks - 12(y - y_0) + 6(y - y_0) - 4ks_0 - 2ks$$
$$g''(1) \; = 2ks_0 + 4ks - 6(y - y_0)$$
$$g''(1) \,/\, k = 4s + 2s_0 - 6(y - y_0)/k$$

Expand the terms on the right with the definitions of s and y.

$$s = 3(y_1 - y_0) - 2(ks_0 + (1-k)s_1) - \tfrac{1}{2}(k^2w_0 - (1-k)^2w_1)$$
$$2s = 6(y_1 - y_0) - 4(ks_0 + (1-k)s_1) - (k^2w_0 - (1-k)^2w_1)$$
$$2s = 6y_d - 4(-ks_d + s_1) - (k^2w_0 - (1 - 2k + k^2)\, w_1)$$
$$2s = 6y_d + 4ks_d - 4s_1 - (k^2w_0 - k^2w_1 + 2kw_1 - w_1)$$
$$2s = 6y_d + 4ks_d - 4s_1 - (-\,k^2w_d + 2kw_1 - w_1)$$

$$
\begin{aligned}
&2s = 6y_d + 4ks_d - 4s_1 + k^2w_d - 2kw_1 + w_1\\
&2s = k^2w_d - 2kw_1 + 4ks_d + 6y_d - 4s_1 + w_1\\
&4s + 2s_0 = 2k^2w_d - 4kw_1 + 8ks_d + 12y_d - 8s_1 + 2w_1 + 2s_0\\
&4s + 2s_0 = 2k^2w_d - 4kw_1 + 8ks_d + 12y_d - 2s_d - 6s_1 + 2w_1
\end{aligned}
$$

$$
\begin{aligned}
&y = y_0 + k(y_1 - y_0) \ + k(1-k)(\tfrac{2}{3}(s_0 - s_1) + \tfrac{1}{6}(kw_0 + (1-k)w_1))\\
&6(y - y_0) = 6k(y_1 - y_0) \ + k(1-k)(4(s_0 - s_1) + (kw_0 + (1-k)w_1))\\
&6(y - y_0)/k = 6(y_1 - y_0) \ + (1-k)(4(s_0 - s_1) + (kw_0 + (1-k)w_1))\\
&6(y - y_0)/k = 6y_d + (1-k)(-4s_d - kw_d + w_1)\\
&6(y - y_0)/k = 6y_d - 4s_d - kw_d + w_1 + 4ks_d + k^2w_d - kw_1\\
&6(y - y_0)/k = k^2w_d - kw_d - kw_1 + 4ks_d + 6y_d - 4s_d + w_1\\
&-6(y - y_0)/k = -k^2w_d + kw_d + kw_1 - 4ks_d - 6y_d + 4s_d - w_1
\end{aligned}
$$

$$
\begin{aligned}
g''(1) / k \ &= k^2w_d - 3kw_1 + kw_d + 4ks_d + 6y_d + 2s_d - 6s_1 + w_1\\
&= w_dk^2 + (-3w_1 + w_d + 4s_d)k + 6y_d + 2(s_d - 3s_1) + w_1\\
&= w_dk^2 + (-2w_1 - w_0 + 4s_d)k + 6y_d + 2(-s_0 - 2s_1) + w_1\\
&= w_dk^2 + (4s_d - w_u)k + 6y_d - 2s_u + w_1
\end{aligned}
$$

where,

$$
\begin{aligned}
&w_u = w_0 + 2w_1\\
&s_u = s_0 + 2s_1
\end{aligned}
$$

Solve similarly for $h''(1)$,

$$
\begin{aligned}
&h''(1) = 6[j(s_1 + s) - 2(y - y_1)] + 2[3(y - y_1) \ - j(2s_1 + s)]\\
&h''(1) \ = 2js_1 + 4js - 6(y - y_1)\\
&h''(1) / j = 4s + 2s_1 - 6(y - y_1)/j\\
&-h''(1) / (1-k) = 4s + 2s_1 + 6(y - y_1)/(1-k)\\
&h''(1) / (1-k) =- 4s - 2s_1 - 6(y - y_1)/(1-k)
\end{aligned}
$$

$$
\begin{aligned}
&2s = k^2w_d - 2kw_1 + 4ks_d + 6y_d - 4s_1 + w_1\\
&2s = k^2w_d + 4ks_d - 2kw_1 + 6y_d - 4s_1 + w_1\\
&4s = 2k^2w_d + 8ks_d - 4kw_1 + 12y_d - 8s_1 + 2w_1\\
&4s + 2s_1 = 2k^2w_d + 8ks_d - 4kw_1 + 12y_d - 6s_1 + 2w_1\\
&-4s - 2s_1 = -2k^2w_d - 8ks_d + 4kw_1 - 12y_d + 6s_1 - 2w_1
\end{aligned}
$$

$$
\begin{aligned}
&y = y_0 + k(y_1 - y_0) \ + k(1-k)(\tfrac{2}{3}(s_0 - s_1) + \tfrac{1}{6}(kw_0 + (1-k)w_1))\\
&y = y_1 + (1-k)(y_0 - y_1) \ + k(1-k)(\tfrac{2}{3}(s_0 - s_1) + \tfrac{1}{6}(kw_0 + (1-k)w_1))\\
&(y - y_1)/(1-k) = (y_0 - y_1) \ + k(\tfrac{2}{3}(s_0 - s_1) + \tfrac{1}{6}(kw_0 + (1-k)w_1))
\end{aligned}
$$

$$6(y-y_1)/(1-k) = 6(y_0-y_1) + k(4(s_0-s_1)+(kw_0+(1-k)w_1))$$
$$6(y-y_1)/(1-k) =- 6y_d + k(-4s_d + kw_0 - kw_1 + w_1)$$
$$6(y-y_1)/(1-k) =- 6y_d + k(-4s_d - kw_d + w_1)$$
$$6(y-y_1)/(1-k) =- 6y_d - 4ks_d - k^2w_d + kw_1$$
$$6(y-y_1)/(1-k) =- k^2w_d - 4ks_d + kw_1 - 6y_d$$
$$-6(y-y_1)/(1-k) = k^2w_d + 4ks_d - kw_1 + 6y_d$$

$$\begin{aligned} h''(1)\,/\,(1-k) &= -k^2w_d - 4ks_d + 3kw_1 - 6y_d + 6s_1 - 2w_1 \\ &= -w_dk^2 + (-4s_d + 3w_1)k - 6y_d + 6s_1 - 2w_1 \end{aligned}$$

3. Solve for second derivatives when *g''(1)* and *h''(1)* are zero

We find second derivatives such that g''(1) and h''(1) are zero.

$$0 = r_g(k) = g''(1)\,/\,k = w_dk^2 + (4s_d - w_0 - 2w_1)k + 6y_d - 2s_0 - 4s_1 + w_1$$
$$0 = r_h(k) = h''(1)\,/\,(1-k) = -w_dk^2 + (-4s_d + 3w_1)k - 6y_d + 6s_1 - 2w_1$$

$$\begin{aligned} r_g(k) &= w_dk^2 + (4s_d - w_0 - 2w_1)k + 6y_d - 2s_0 - 4s_1 + w_1 \\ &= w_1k^2 - w_0k^2 + 4s_dk - w_0k - 2w_1k + 6y_d - 2s_0 - 4s_1 + w_1 \\ &= (k^2 - 2k + 1)w_1 + (-k^2 - k)w_0 + 4s_dk + 6y_d - 2s_0 - 4s_1 \\ &= (k-1)^2w_1 - k(k+1)w_0 + 4s_dk + 6y_d - 2s_0 - 4s_1 \end{aligned}$$

$$\begin{aligned} r_h(k) &= -w_dk^2 + (-4s_d + 3w_1)k - 6y_d + 6s_1 - 2w_1 \\ &= -w_1k^2 + w_0k^2 - 4s_dk + 3w_1k - 6y_d + 6s_1 - 2w_1 \\ &= (-k^2 + 3k - 2)w_1 + k^2w_0 - 4s_dk - 6y_d + 6s_1 \\ &= -(k-1)(k-2)w_1 + k^2w_0 - 4s_dk - 6y_d + 6s_1 \end{aligned}$$

First solve for w_0 by elminating w_1.

$$0 = (k-1)^2(k-2)w_1 - k(k+1)(k-2)w_0 + (k-2)(4s_dk + 6y_d - 2s_0 - 4s_1)$$
$$0 = -(k-1)^2(k-2)w_1 + k^2(k-1)w_0 + (k-1)(-4s_dk - 6y_d + 6s_1)$$

$$\begin{aligned} &(k^2(k-1) - k(k+1)(k-2))w_0 \\ &\quad = k(k(k-1) - (k+1)(k-2))w_0 \\ &\quad = k(k^2 - k - (k^2 - 2k + k - 2))w_0 \\ &\quad = k(k^2 - k - k^2 + k + 2)w_0 \\ &\quad = 2kw_0 \end{aligned}$$

$$\begin{aligned} &(k-2)(4s_dk + 6y_d - 2s_0 - 4s_1) + (k-1)(-4s_dk - 6y_d + 6s_1) \\ &\quad = (4s_dk^2 + 6y_dk - 2s_0k - 4s_1k) + (-8s_dk - 12y_d + 4s_0 + 8s_1) \\ &\qquad + (-4s_dk^2 - 6y_dk + 6s_1k) + (4s_dk + 6y_d - 6s_1) \end{aligned}$$

$$\begin{aligned}
&= -2s_0k + 2s_1k - 4s_dk - 6y_d + 4s_0 + 2s_1 \\
&= 2s_dk - 4s_dk - 6y_d + 4s_0 + 2s_1 \\
&= -2s_dk - 6y_d + 4s_0 + 2s_1
\end{aligned}$$

$$0 = kw_0 - s_dk - 3y_d + 2s_0 + s_1$$
$$w_0 = s_d + \tfrac{1}{k}(3y_d - 2s_0 - s_1)$$

Then solve for w_1 by eliminating w_0.

$$0 = k(k-1)^2w_1 - k^2(k+1)w_0 + 4s_dk^2 + 6y_dk - 2s_0k - 4s_1k$$
$$0 = -(k+1)(k-1)(k-2)w_1 + k^2(k+1)w_0 - 4s_dk(k+1) - 6y_d(k+1) + 6s_1(k+1)$$

$$0 = (k(k-1)^2 - (k+1)(k-1)(k-2))w_1 + (4s_dk^2 + 6y_dk - 2s_0k - 4s_1k) - 4s_dk(k+1) - 6y_d(k+1) + 6s_1(k+1)$$

$$\begin{aligned}
&k(k-1)^2 - (k+1)(k-1)(k-2) \\
&\quad = k(k^2 - 2k + 1) - (k^2 - 1)(k-2) \\
&\quad = k^3 - 2k^2 + k - (k^3 - 2k^2 - k + 2) \\
&\quad = k^3 - 2k^2 + k - k^3 + 2k^2 + k - 2 \\
&\quad = 2k - 2 \\
&\quad = 2(k-1)
\end{aligned}$$

$$\begin{aligned}
&(4s_dk^2 + 6y_dk - 2s_0k - 4s_1k) - 4s_dk(k+1) - 6y_d(k+1) + 6s_1(k+1) \\
&\quad = 4s_dk^2 + 6y_dk - 2s_0k - 4s_1k - 4s_dk^2 - 4s_dk - 6y_dk - 6y_d + 6s_1k + 6s_1 \\
&\quad = -2s_0k - 4s_1k - 4s_dk - 6y_d + 6s_1k + 6s_1 \\
&\quad = -2s_0k - 4s_1k - 4(s_1 - s_0)k - 6y_d + 6s_1k + 6s_1 \\
&\quad = -2s_0k - 4s_1k - 4s_1k + 4s_0k - 6y_d + 6s_1k + 6s_1 \\
&\quad = 2s_0k - 2s_1k - 6y_d + 6s_1 \\
&\quad = -2s_dk - 6y_d + 6s_1
\end{aligned}$$

$$0 = (k-1)w_1 - s_dk - 3y_d + 3s_1$$
$$w_1 = \tfrac{1}{k-1}(s_dk + 3y_d - 3s_1)$$