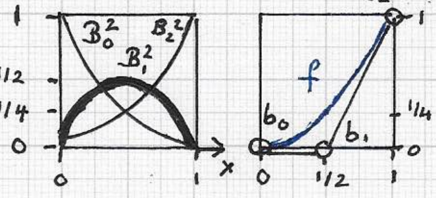
Stratovan

FRACTIONAL CALCULUS AND FEATURES – Cont'd.

• **Fractional derivative:**

Polynomial of degree 2 in Bernstein-Bézier form

$$f(x) = \sum_{i=0}^{2} b_i \, B_i^2(x)$$



$$B_i^2(x) = \binom{2}{i}(1-x)^{2-i}x^i,$$
$i = 0 \ldots 2$

b_i = Bernstein-Bézier coeffs./control points

$$f'(x) = 2 \cdot \sum_{i=0}^{1} \Delta b_i \, B_i^1(x)$$
$$= 2 \sum_{i=0}^{1} \bar{b}_i \, B_i^1(x)$$

raise degree

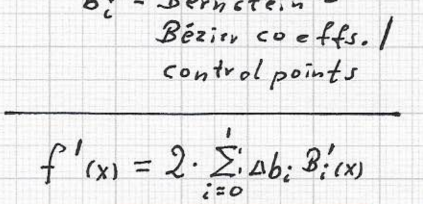
$$= 2 \sum_{i=0}^{2} \hat{b}_i \, B_i^2(x),$$

where $\hat{b}_0 = \frac{0}{2}\bar{b}_{-1} + \frac{2}{2}\bar{b}_0$,

$\hat{b}_1 = \frac{1}{2}\bar{b}_0 + \frac{1}{2}\bar{b}_1$,

$\hat{b}_2 = \frac{2}{2}\bar{b}_1 + \frac{0}{2}\bar{b}_2$,

and $\bar{b}_i = \Delta b_i = b_{i+1} - b_i$



When considering only POLYNOMIALS (in monomial or Bernstein-Bézier form), one can view the definition of "simplified fractional derivatives" as blending of the $(n+1)$ integer-order derivatives $f^{(0)}(x), \ldots, f^{(n)}(x)$ of a degree-n polynomial. Further, since each of these $(n+1)$ derivatives can be written — via **degree-raising** — as a degree-n polynomial, non-integer-order derivatives are obtained by interpolating the $(n+1)$ polynomials $f^{(0)}(x), \ldots, f^{(n)}(x)$ in a second direction — α-direction — to define a fractional derivative $f^{(\alpha)}(x)$, $\alpha \in [0, n]$, $\alpha \in \mathbb{R}$.

The representation of a polynomial in Bernstein-Bézier form has the "advantage" that the "control polygon" defined by the coefficients b_i closely resembles the shape of the graph of a polynomial $f(x)$. We therefore use the Bernstein-Bézier basis polynomials $B_i^n(x)$ here. Raising the degrees of all $(n+1)$ derivatives of a polynomial $f(x)$

FRACTIONAL CALCULUS AND FEATURES – Cont'd.

• Fractional derivatives:

Polynomial of degree 2 in Bernstein-Bézier form

...

$$f''(x) = 2\cdot 1\cdot \sum_{i=0}^{0} \Delta^2 b_i\, B_i^0(x)$$

$$= 2\cdot 1\cdot \sum_{i=0}^{0} \bar{\bar{b}}_i\, B_i^0(x)$$

raise degree twice

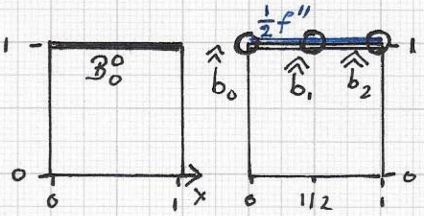
$$= 2\cdot 1\cdot \sum_{i=0}^{1} \tilde{b}_i\, B_i^1(x)$$

$$= 2\cdot 1\cdot \sum_{i=0}^{2} \hat{\hat{b}}_i\, B_i^2(x),$$

where $\tilde{b}_0 = \frac{0}{1}\bar{\bar{b}}_{-1} + \frac{1}{1}\bar{\bar{b}}_0$,
$\tilde{b}_1 = \frac{1}{1}\bar{\bar{b}}_0 + \frac{0}{1}\bar{\bar{b}}_1$,

$\hat{\hat{b}}_0 = \frac{0}{2}\tilde{b}_{-1} + \frac{2}{2}\tilde{b}_0$,
$\hat{\hat{b}}_1 = \frac{1}{2}\tilde{b}_0 + \frac{1}{2}\tilde{b}_1$,
$\hat{\hat{b}}_2 = \frac{2}{2}\tilde{b}_1 + \frac{0}{2}\tilde{b}_2$,

and $\bar{\bar{b}}_i = \Delta^2 b_i = b_{i+2} - 2b_{i+1} + b_i$



$$\Rightarrow f(x) = x^2 = \sum_{i=0}^{2} b_i\, B_i^2(x),$$

$$f'(x) = 2x = 2\cdot \sum_{i=0}^{2} \hat{b}_i\, B_i^2(x),$$

$$f''(x) = 2 = 2\cdot 1\cdot \sum_{i=0}^{2} \hat{\hat{b}}_i\, B_i^2(x)$$

is done elegantly for this polynomial when given in Bernstein-Bézier form. In the example (left), the polynomial $f(x) = x^2$ and its first and second derivatives are written as degree-2 Bernstein-Bézier-based polynomials. Using the algorithm for degree-raising and the forward-difference operator 'Δ' for the coefficients, the three derivatives of $f(x) = x^2$ – i.e., $f^{(0)}(x)$, $f^{(1)}(x)$, $f^{(2)}(x)$ – can all be written with three coefficients, always involving all **three** quadratic Bernstein-Bézier polynomials $B_0^2(x)$, $B_1^2(x)$ and $B_2^2(x)$.

Generally, the j^{th} derivative of the polynomial $f(x) = \sum_{i=0}^{n} b_i\, B_i^n(x)$ is given as

$$\frac{d^j}{dx^j} f(x) = \frac{n!}{(n-j)!} \sum_{i=0}^{n-j} \Delta^j b_i\, B_i^{n-j}(x),$$

where j is an integer and

$$\Delta^j b_i = \Delta^{j-1} b_{j+1} - \Delta^{j-1} b_j,$$

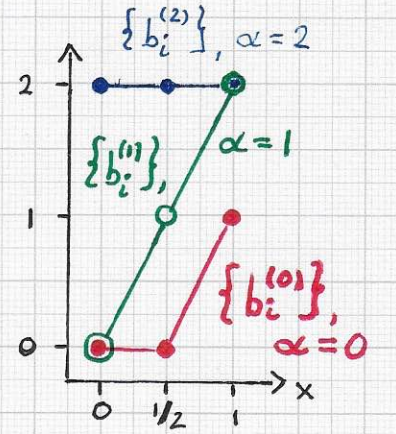
with $\Delta^0 b_i = b_i$. The formula used for elevating/raising **the** degree of $f(x)$ from n to $(n+1)$ is:

$$\hat{b}_i = \frac{i}{n+1} b_{i-1} + \frac{n+1-i}{n+1} b_i, \quad i = 0 \ldots (n+1).$$

$$\Rightarrow f(x) = \sum_{i=0}^{n+1} \hat{b}_i\, B_i^{n+1}(x).$$

FRACTIONAL CALCULUS AND FEATURES – Cont'd.

• Fractional derivatives:



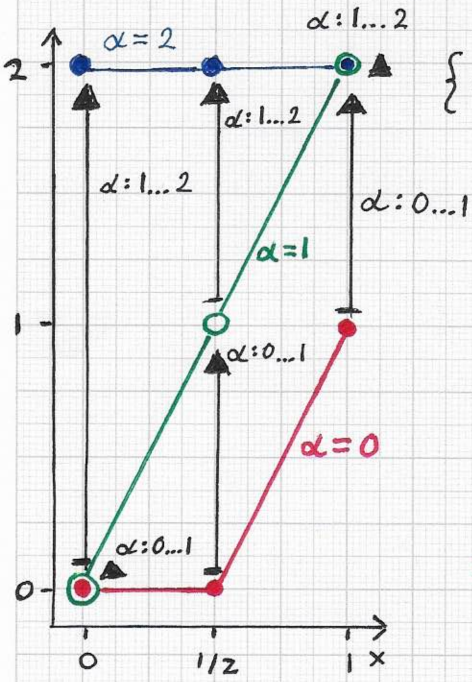
Control polygons of function x^2 for its derivatives $f^{(0)}(x)$, $f^{(1)}(x)$ and $f^{(2)}(x)$

We use the notation $b_i^{(j)}$ to denote the i^{th} "control point"/coefficient for the j^{th} derivative of a polynomial $f(x)$, i.e., $f(x) = \sum_{i=0}^{n} b_i^{(0)} B_i^n(x)$. Considering the example $f(x) = x^2$, its three derivatives in Bernstein-Bézier form are

$$f^{(j)}(x) = \sum_{i=0}^{2} b_i^{(j)} B_i^2(x) \quad , \; j = 0 \ldots 2,$$

where $b_i^{(0)} = b_i$, $b_i^{(1)} = 2\,\hat{b}_i$ and $b_i^{(2)} = 2 \cdot 1 \cdot \hat{\hat{b}}_i$. Considering the example (left image), it is possible to think about $f^{(j)}(x)$ in terms of its associated control polygon, i.e., the geometry of $f^{(j)}(x)$'s graph.

{ • Note: The x-coordinates of control points $(x_i, b_i^{(j)})^T$ are given by $x_i = i/n$, $i = 0 \ldots n$, for a degree-n polynomial defined over the interval $[0, 1]$. }



Piecewise linear interpolation of control polygons via blending control points $b_i^{(j)}$ and $b_i^{(j+1)}$

The simplest, most straightforward and numerically most "robust" method for blending all derivatives $f^{(j)}(x)$ to define non-integer-order derivatives $f^{(\alpha)}(x)$ interpolates two derivatives linearly:

$$f^{(\alpha)}(x) = (\lceil \alpha \rceil - \alpha)\, f^{(\lfloor \alpha \rfloor)} + (\alpha - \lfloor \alpha \rfloor)\, f^{(\lceil \alpha \rceil)} ,$$

$\alpha \in (0,1)$, $\alpha \in (1,2)$, …, $\alpha \in (n-1, n)$, $\lfloor \alpha \rfloor$, $\lceil \alpha \rceil$ = floor, ceiling of α.

FRACTIONAL CALCULUS AND FEATURES – Cont'd.

• Fractional derivatives: The following questions and aspects arise when defining a simple and "meaningful" (="information-enriching") method for locally estimating fractional derivatives for pixels or voxels of a 2D/3D image:

● Why use polynomials $f(x)$?

$$f(x) = f^{(0)}(x) = \sum_{i=0}^{n} c_i x^i = \sum_{i=0}^{n} b_i B_i^n(x)$$

(i) Why should one consider polynomials (tensor products of polynomials in the 2D and 3D cases) as class of functions for which to define fractional derivatives? The use of low-degree polynomials to define local approximations for images is commonly done and accepted practice! Many discrete filter masks/templates and finite difference schemes exist for the class of polynomial functions!

● How to interpolate $f^{(0)}(x), \dots, f^{(n)}(x)$ to obtain "good" non-integer-order derivatives $f^{(\alpha)}(x)$, $0<\alpha<n$?

(ii) Given a polynomial $f^{(0)}(x)$ of degree n and its $(n+1)$ integer-order derivatives, $f^{(0)}(x), \dots, f^{(n)}(x)$, is there a "best way" to interpolate these $(n+1)$ polynomials in α-direction $(0<\alpha<n)$ such that, for a specific value $\bar{x}$, the value of $f^{(\alpha)}(\bar{x})$ = "Interpolate $(\alpha, f^{(0)}(\bar{x}), \dots, f^{(n)}(\bar{x}))$" represents an interpolated value that is "ideal" (i.e., most beneficial for feature-based image data classification)?

FRACTIONAL CALCULUS AND FEATURES – Cont'd.

• Fractional derivatives: (ii) ... Three methods are obvious candidates to consider for computing an interpolated value for $f^{(\alpha)}(\bar{x})$:

• **Piecewise Linear interpolation**

a) linearly interpolate $f^{(\lfloor\alpha\rfloor)}(\bar{x})$ and $f^{(\lceil\alpha\rceil)}$ for $\lfloor\alpha\rfloor < \alpha < \lceil\alpha\rceil$, $0<\alpha<n$;

• **Natural cubic spline interpolation**

b) compute the natural cubic spline interpolating the values $f^{(0)}(\bar{x})$, $f^{(1)}(\bar{x}), \ldots, f^{(n)}(\bar{x})$ and evaluate this spline (depending on α) for α;

• **Polynomial interpolation**

c) compute the degree-n polynomial (depending on α) interpolating the values $f^{(0)}(\bar{x}), \ldots, f^{(n)}(\bar{x})$ and evaluate it for α.

(iii) Given a specific method used for computing fractional derivatives $f^{(\alpha)}(x)$, does this method lead to a corresponding discrete filter mask that supports efficient and numerically stable fractional derivative computations?

Bernstein-Bézier control polygons for $f^{(0)} = x^2$ (•), $f^{(1)} = 2x$ (•), $f^{(2)} = 2$ (•); piecewise linear interpolation in α-direction

Concerning the type of interpolation performed in α-direction, piecewise linear interpolation is simple, efficient, numerically stable, and it computes only CONVEX COMBINATIONS of $f^{(i)}(x)$ values; thus, UNDER- and OVERSHOOTS are IMPOSSIBLE.

∴ BH