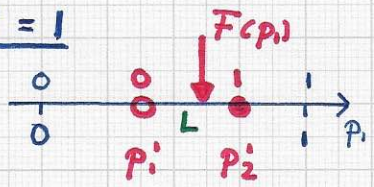
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks: ...

• L-cuboid examples:

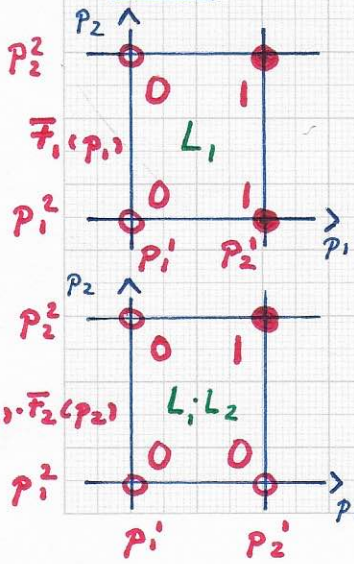
• H = 1



The 1D cuboid is the interval (p_1', p_2'); it is the only L-cuboid type for H=1. The value for p_1, where $p_1' < p_1 < p_2'$, is the decider function value

$$F(p_1) = F_1(p_1) = \frac{p_1 - p_1'}{p_2' - p_1'} = \frac{\delta_1}{\Delta_1}.$$

• H = 2

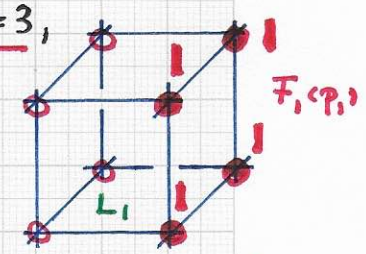


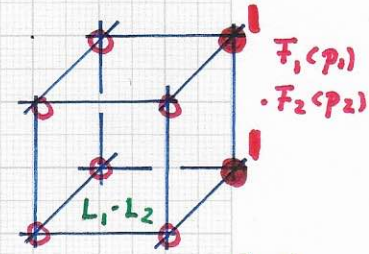
Two cases of L-cuboid types in the 2D setting. An L_1 1-linear cuboid and an $L_1 \cdot L_2$ 2-linear cuboid are shown. For the L_1 case one obtains $F(p_1, p_2) = F_1(p_1) = \delta_1/\Delta_1$.

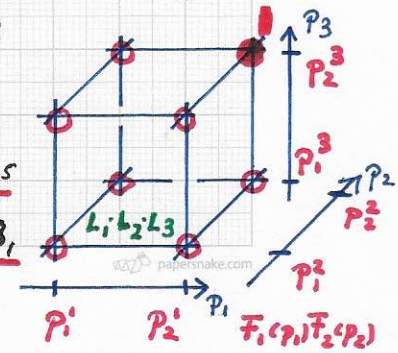
For the $L_1 \cdot L_2$ case

$$F(p_1, p_2) = F_1(p_1) F_2(p_2) = (\delta_1/\Delta_1) \cdot (\delta_2/\Delta_2).$$

The L-cuboids are special: (i) they have vertex values 0 and 1 only, and (ii) they never have edges with decreasing value ("from 1 to 0" in any of the positive p_i-directions), as a consequence of their non-decreasing monotonic function value behavior in all domain directions, i.e., $p_1^-, \ldots, p_H^-$ direction. The figures for H=1 and H=2 (left) illustrate examples of the two types of behavior possible: 1-linear behavior (H=1) and 1- and 2-Linear behavior (H=2). For H=3, an L-cuboid can have 1 vertex, 2 vertices or 4 vertices with value 1, see figure (right). Generally, the number of vertices with value 1 of an L-cuboid for H dimensions is 1, 2, 4, 8, ... and 2^{H-1}.







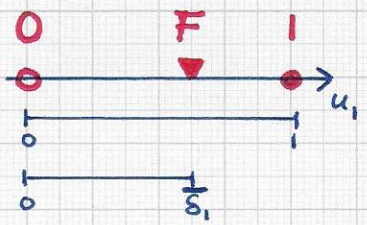
...

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks: ...

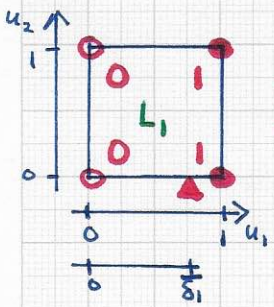
• Evaluation of "L-functions" in normalized unit hyper-cube domains — by mapping an L-cuboid to a hyper-cube:

• H = 1

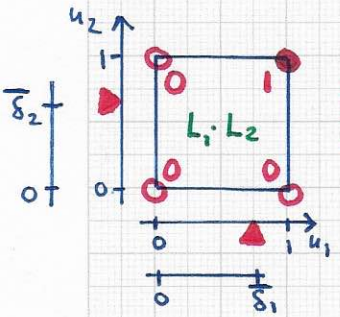


$F = \bar{\delta}_1$

• H = 2



$F = \bar{\delta}_1$



$F = \bar{\delta}_1 \cdot \bar{\delta}_2$

The values $\bar{\delta}_1$ and $\bar{\delta}_2$ can also be viewed as "Local coordinates" in a unit, standard hyper-cube.

The figure on the previous page for H = 3 shows three examples that stand for the 3 prototypical types of multi-Linear L-cuboids possible: 1-, 2- and 3-Linear. The illustrated examples have the associated decider functions

$$F(p_1,p_2,p_3) = F(p_1) = F_1(p_1) = \delta_1/\Delta_1 ,$$

$$F(p_1,p_2,p_3) = F(p_1,p_2) = F_1(p_1)\,F_2(p_2) = (\delta_1/\Delta_1)(\delta_2/\Delta_2) ,$$

$$F(p_1,p_2,p_3) = F_1(p_1)\,F_2(p_2)\,F_3(p_3) = (\delta_1/\Delta_1)(\delta_2/\Delta_2)(\delta_3/\Delta_3) .$$

Here, $\delta_i = p_i - p_1^i$ and $\Delta_i = p_2^i - p_1^i$.

Thus, given a tuple $(p_1,\dots,p_H)$ with K of its components — e.g., $p_{i_1},\dots,p_{i_K}$ — having values satisfying $p_1^{i_1} < p_{i_1} < p_2^{i_1}$, ..., $p_1^{i_K} < p_{i_K} < p_2^{i_K}$, the value of the overall decider function in the associated K-Linear L-cuboid is

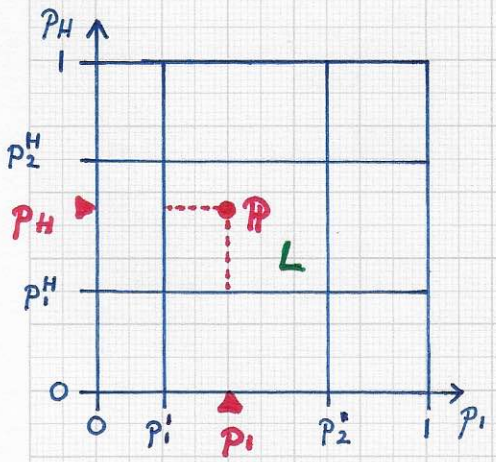
$$F(p_1,\dots,p_H) = F(p_{i_1},\dots,p_{i_K}) = F_{i_1}(p_{i_1})\cdot\dots\cdot F_{i_K}(p_{i_K}) = (\delta_{i_1}/\Delta_{i_1})\cdot\dots\cdot(\delta_{i_K}/\Delta_{i_K}) .$$

The index set $\{i_1,\dots,i_K\}$ is a subset of the index set $\{1,2,\dots,H\}$. Using an efficient data structure for the cuboids, the computation of the products is not expensive.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks:...



$\mathbb{P} = (p_1, \ldots, p_H)$

tuple $\mathbb{P}$ in an L-cuboid

⇒ compute value of F iteratively as follows:

$F = 1$;

$F *= ((p_1 - p_1') / (p_2' - p_1'))$;

⋮

$F *= ((p_H - p_1^H) / (p_2^H - p_1^H))$;

Iterative computation of decider function F when the specific $\mathbb{P}$-tuple lies in an L-cuboid.

The cases $F=0$ and $F=1$ can be detected easily via the associated conditions (page 15, 5/5/2022). The iterative procedure described here is applicable to the $2^H - 1$ L-cuboids.

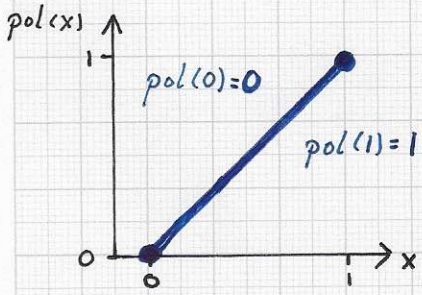
Whenever a tuple $(p_1, \ldots, p_H)$ does not satisfy the conditions that imply that $F(p_1, \ldots, p_H) = 0$ or $F(p_1, \ldots, p_H) = 1$, at least one of the p_i-values must satisfy the condition $p_1^i < p_i < p_2^i$. In this case, one must evaluate the function F in a cuboid of one of the possible multi-linear L-functions. In "C-language-like" style, the multi-linear function F can be evaluated as follows:

- $F = 1$;
- FOR $i = 1$ TO H DO
  - IF $p_1^i < p_i < p_2^i$
    - $F *= ((p_i - p_1^i) / (p_2^i - p_1^i))$;

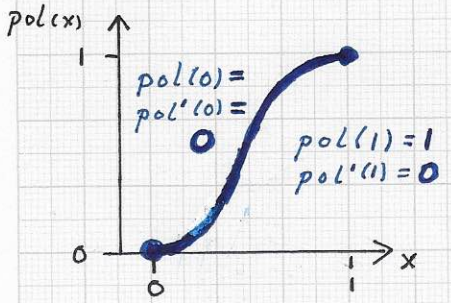
At the end of this multiplication procedure, F has as its value the product of all local, normalized p_i-values relative to the L-type cuboid containing a given tuple $(p_1, \ldots, p_H)$. The figure (left) illustrates this situation and summarizes the multiplication procedure. THE SMALLER THE COMBINED, TOTAL HYPER-VOLUME OF ALL L-CUBOIDS IS, THE SMALLER IS THE NUMBER OF TIMES ONE MUST HANDLE THEM. ...

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks: ...

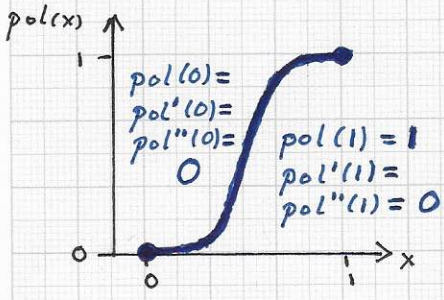


$pol(x) = L(x) = x$



$pol(x) = -2x^3 + 3x^2$

(cubic Hermite pol.)



$pol(x) = 6x^5 - 15x^4 + 10x^3$

(quintic Hermite pol.)

Decider functions $F_i(p_i)$ have a "RAMP" for the transition from value 0 to value 1. HERMITE functions can be used to define the "RAMP."

In summary, only 3 cases can arise when defining $F(p_1,\dots,p_H)$ as the tensor product of H individual decider functions $F_i(p_i)$, $i=1\dots H$, when each F_i has a 0-value, L-value and 1-value segment:

i) IF $p_1 \le p_1^1 \vee \dots \vee p_H \le p_1^H$
THEN $F = 0$;

ii) IF $p_1 \ge p_2^1 \wedge \dots \wedge p_H \ge p_2^H$
THEN $F = 1$;

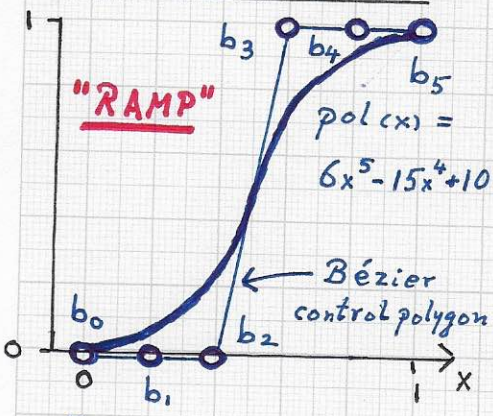
iii) IF neither i) nor ii) is the case
THEN /* F is multi-linear */
$F = 1$;
FOR $i = 1$ TO H DO
IF $p_1^i < p_i < p_2^i$
THEN $F *= (\delta_i / \Delta_i)$;

Thus, the practical implementation of this overall decider function is simple, efficient and elegant. "Most importantly," one can still comprehend and fully understand the deterministic computations involved in calculating the value of $F(p)$ defining the classification outcome.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks: ...



Representation of the quintic Hermite polynomial in Bernstein-Bézier form - involving six Bézier control points $b_0, \ldots, b_5$ shown in the figure. They define the polynomial's control polygon.

The control points are the coefficients (or ordinates) when representing the quintic Hermite polynomial in the Bernstein basis:

$$pol(x) = \sum_{i=0}^{5} b_i B_i^5(x),$$

where $b_0 = b_1 = b_2 = 0$ and $b_3 = b_4 = b_5 = 1$.

The x-values associated with $b_0, \ldots, b_5$ are $0 \cdot 15, \ldots, 5 \cdot 15$, respectively. The Bernstein polynomials are

$$B_i^5(x) = \binom{5}{i} (1-x)^{5-i} \cdot x^i,$$

for $x \in [0, 1]$.

The only information that this classification approach expects to be supplied by an "expert" as input is the set of pairs (p_1^i, p_2^i), $i = 1 \ldots H$. In other words, an expert provides "two facts": (i) a decider function $F_i(p_i)$ is 0 for $p_i \in [0, p_1^i]$ and (ii) a decider function $F_i(p_i)$ is 1 for $p_i \in [p_2^i, 1]$. THE EXPERT DOES NOT KNOW, DOES NOT SPECIFY HOW $F_i(p_i)$ SHOULD/MUST TRANSITION FROM 0 TO 1 IN THE "RAMP" INTERVAL (p_1^i, p_2^i). THIS CAN BE VIEWED AS A DEGREE OF FREEDOM TO BE USED TO OPTIMIZE CLASSIFICATION RESULTS.

For the purpose of classification it is not necessary that a "RAMP" function satisfies the derivative requirements that a polynomial Hermite function must satisfy at the ends (at $x = 0$ and $x = 1$). A "RAMP" function only must monotonically increase in value from 0 to 1. ...