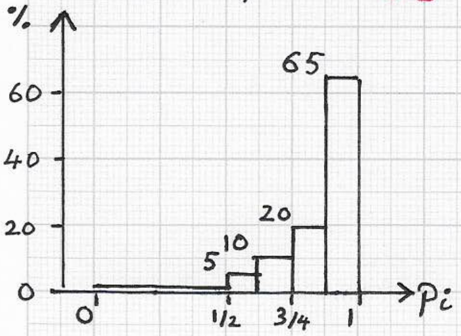
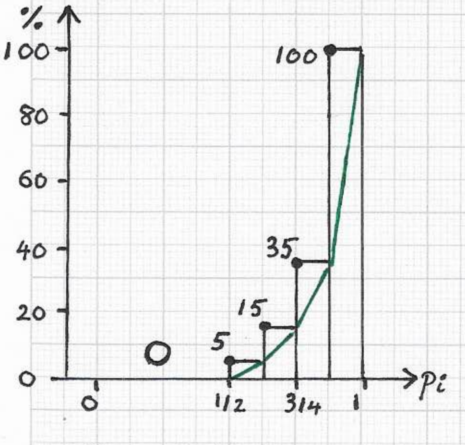
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Binned distribution example: p_i-value histogram resulting when calculating similarity value for each element ○ and stored class sample set {●}:



The interval [0, 1] is subdivided into 8 uniform bins, i.e., bin_1, ... bin_8; the sketched percentage tuple for these bins is (0,0,0,0,5,10,20,65).



Cumulative distribution resulting from the binned histogram shown above. The added polyline (green) is the "right envelope" we can use to define the "RAMP" of the decider function $F_i(p_i)$.

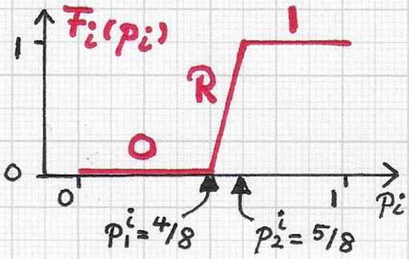
The second dataset, {○}, is effectively used to compute for each element ○ its maximal similarity value p_i, i.e., the maximum of all similarities between the one element ○ and all the elements of {●}. (The similarity measure SIM_{max} can be employed for this purpose.) The result of the computation of similarity values for each element in {○} is a distribution of p_i-values in [0, 1]. The figures (left) provide an example. As a consequence of the computation of similarity values (via SIM_{max}), the vast majority of p_i-values are very close to 1. The binned p_i-value histogram has an associated cumulative distribution that can be used to construct the desired univariate decider function $F_i(p_i)$ automatically. The "right envelope" (green polyline) inserted into the cumulative distribution sketch (bottom) indicates that [4/8, 5/8] is a viable interval for the "RAMP" of $F_i(p_i)$.

...

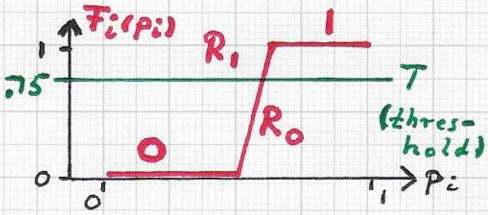
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks: ...

Viable decider function $F_i(p_i)$ constructed automatically from cumulative distribution shown in figure on previous page:

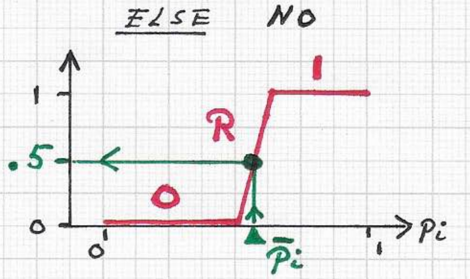


Resulting decider function for a specific class and scale i.



Binary classification result:

IF $F_i(p_i) \geq T$
THEN YES
ELSE NO



Probabilistic classification:

PROB (YES) = $F_i(\bar{p}_i)$
PROB (NO) = $1 - F_i(\bar{p}_i)$

Two options to use $F_i(p_i)$ for classification

WE CAN NOW CREATE UNIVARIATE DECIDER FUNCTIONS FOR ALL MATERIAL CLASSES AND ALL SCALES. THESE FUNCTIONS CAN BE WRITTEN AS $F_i^{cl} = F_{sc}^{cl} = F_{sc}^{cl}(p_{sc}^{cl})$ — WHERE $cl = 1 \ldots C$ IS THE CLASS INDEX AND $sc = 1 \ldots H$ IS THE SCALE INDEX.

Thus, given a new unclassified image segment, we first compute its **H** eigenfunction-based coefficient value histograms; in a second step, we can compute $C \cdot H$ univariate decider function values for this unclassified segment. These values result from evaluating $F_{sc}^{cl}(p_{sc}^{cl})$, $cl = 1 \ldots C$, $sc = 1 \ldots H$. The computation of the value of the argument variable p_{sc}^{cl} of the unclassified segment is also done by calculating **SIM**$_{max}$ for the specific segment's histogram to be classified and the stored sample set in the "database" for the class and scale being considered.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Example of table of $F_{sc}^{cl}(p_{sc}^{cl})$-values for 4 classes and 5 scales:

cl \ sc	1	2	3	4	5	P
1	0	0	0	0	0	0
2	0	.1	.3	.6	1	.4
3	1	.6	.3	.1	0	.4
4	1	1	1	1	1	1
M	2	2	1	2	2	☒

cl = class
sc = scale
P = PROB (YES)
M = no. of matches

Resulting F_{sc}^{cl} table when comparing an unclassified image segment with 4 classes at 5 scales. For example, using a threshold of $T=.5$ leads to a total number of matches of 9.

The P column provides the average value of the values of a row. The M row provides the number of identified matches for a specific scale.

⇒ How does one use F_{sc}^{cl}- and P-values different from 0 and 1 'optimally'?

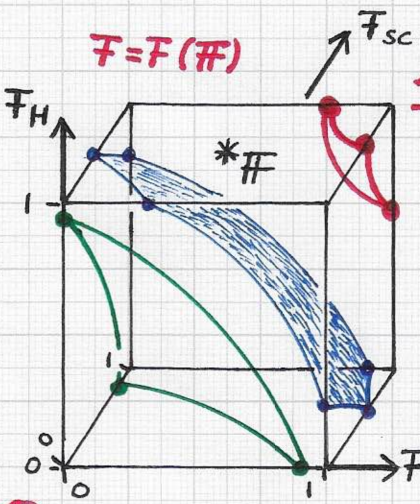
For a given unclassified image segment we can now calculate a table of all $F_{sc}^{cl}(p_{sc}^{cl})$-values that result when evaluating all univariate decider function F_{sc}^{cl}, $cl = 1 \ldots C$, $sc = 1 \ldots H$. The table (left) provides an example of possible values. The unclassified segment **S** does not match any scale of class 1; it does match all scales of class 4. Thus, **S** is a perfect mis-match for class 1 and a perfect match for class 4. Segment **S** matches classes 2 and 3 only at certain scales – and only at a certain level of probability (as defined by the respective "RAMP" parts of the decider functions). Values above the examplary threshold $T=.5$ are shown in green in the table. Therefore, when employing the concept of binary classification via the threshold **T**, a value above (or equal to) **T**'s value is a match.

Best-possible use of F_{sc}^{cl}- and P-values different from 0 and 1 is an OPTIMIZATION problem.

...

OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...



① Sketch of "decision boundaries. An $\mathbb{F}$-tuple defines a point in the H-dimensional hyper-cube. The final over-all decider function $\mathcal{F} = \mathcal{F}(\mathbb{F})$ has a specific contour surface behavior. This sketch shows 3 potential contours of $\mathcal{F}$, 3 hyper-surfaces in the unit hyper-cube. When using a threshold method to determine whethe $\mathbb{F}$ represents a class match (or not), these contours define the boundaries between match and mis-match.

For decision-making one only must be able to effectively compute $\mathcal{F}$'s value for the tuple $\mathbb{F}$.

All univariate decider functions $\mathcal{F}_{sc}^{cl}$ generate values between 0 and 1. For a fixed class cl, we obtain H $\mathcal{F}_{sc}$-values, sc=1...H, for an un-classified segment S. The H values define the class-specific tuple $\mathbb{F} = (\mathcal{F}_1, ..., \mathcal{F}_H)$. The figure (left) represents this tuple $\mathbb{F}$ as point '$*$' in an H-dimensional unit hyper-cube $[0,1]^H$. The sketched point lies in the interior of the hyper--cube, and one must do one of two things:

i) One defines and computes an overall decider function $\mathcal{F}$, where $\mathcal{F} = \mathcal{F}(\mathcal{F}_1, ..., \mathcal{F}_H)$. The value of $\mathcal{F}$ can then be viewed as a probability for S belonging to this class or not.

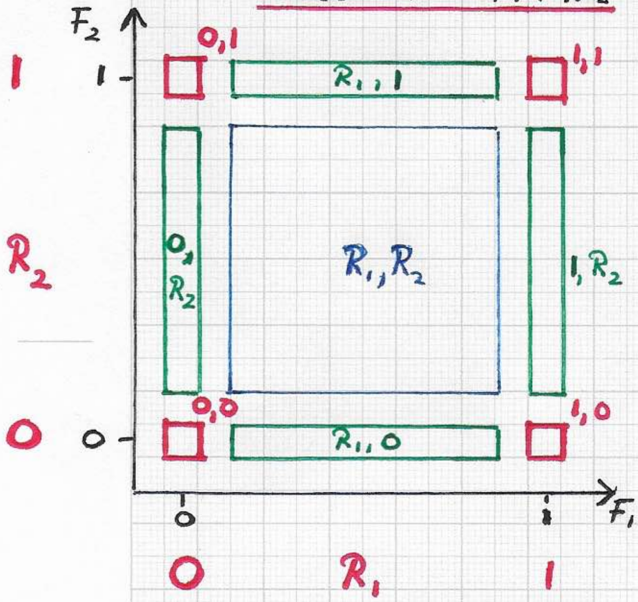
ii) Using a threshold approach, with T as threshold parameter, one can use $\mathcal{F}$'s value to define: $\mathcal{F} \geq T \Rightarrow$ S belongs to this class; $\mathcal{F} < T \Rightarrow$ S does not belong to this class.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks: ...

Example: Combinations of 0-, R- and 1-regions of two univariate decider functions F_1 and F_2 one must consider for the construction of $F(F_1, F_2)$:



The functions F_1 and F_2 yield "three types of values": 0 or 1, or a value between 0 and 1. Thus, one must carefully consider the necessary combinations of these types for the construction of the overall function $F(\mathbb{F})$.

For the construction of the optimal, least-squares-based function $F(\mathbb{F})$ one needs to know/create "exact values" F_j, where

$$F_j = F(\mathbb{F}_j) = F(F_1^j, F_2^j, \ldots),$$
$$F_j \in \{0,1\} \text{ or } F_j \in [0,1].$$

The figure (left) shows an "exploded view" of the nine combinatorially possible combinations of the three distinct regions of two univariate decider functions $F_1(p_1)$ and $F_2(p_2)$: **0-, R- and 1-regions.** The goal is the definition and computation of the defining parameter values of an **overall decider function $F = F(F_1, F_2, \ldots)$.** The function F can either compute a binary decision value (1-match, 0-mis-match) or a probability value for a match or mis-match. Of course, the individual "RAMPs" of the functions $F_1(p_1), F_2(p_2), \ldots$ generate values in $(0,1)$ that can effectively be viewed as match probabilities at the levels of the individual functions $F_i(p_i)$.

Assuming that one knows the "exact" or "desired values" F_j for tuples $(F_1^j, F_2^j, \ldots)$, one can use the conditions $F_j = F(\mathbb{F}_j) = F(F_1^j, \ldots, F_H^j) =$

$$= \sum_{sc=1}^{H} w_{sc} \cdot F_{sc}^j \quad , j = 1 \ldots N,$$

to compute optimal weights w_{sc} from N values F_j. ...