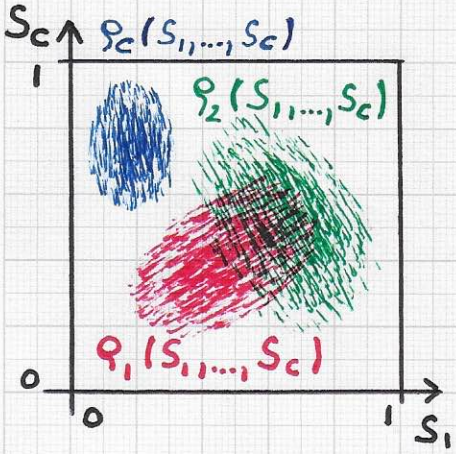
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...



Pages 16–20, dated 9/4/2022 and 9/5/2022, discuss the $-tuple data that is calculated for an unclassified material segment and used to determine class-membership probabilities for the unclassified segment – C probabilities, for classes 1, ..., C. Page 20 introduced a simple, rudimentary idea for calculating these probabilities via density functions with continuous domains in the C-dimensional $-space. At this point, we have a better, more general understanding of the use of such density funetions and can establish a more appropriate definition of the needed probabilities. We use the definition from the previous page, p. 5, as basis and obtain the generalized definition, sketched in the top figure:

$$\mathbb{P}(\$) = \left(P_1(S_1,\dots,S_C), \dots, P_C(S_1,\dots,S_C) \right), \text{ where}$$

$$P_i(S_1,\dots,S_C) = \begin{cases} s(S_1,\dots,S_C)\cdot f_i(S_1,\dots,S_C), \\ \qquad \text{if } \sum_{j=1}^{C} \varrho_j(S_1,\dots,S_C) \neq 0 \\ 0, \text{ otherwise} \end{cases}, \text{ and}$$

$$s(S_1,\dots,S_C) = 1/M \sum_{j=1}^{C} \varrho_j(S_1,\dots,S_C),$$

$$f_i(S_1,\dots,S_C) = \varrho_i(S_1,\dots,S_C) \Big/ \sum_{j=1}^{C} \varrho_j(S_1,\dots,S_C), \quad i=1,\dots,C$$

•••

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks:...

• As described, a multitude of steps is involved in the overall complex pipeline that ultimately produces a "probability tuple" $\mathbb{P}$: imaging; data pre-processing; sample selection for a sample database of materials to be detected; volumetric segmentation of volume image data for the extraction of distinct materials; definition of features of volume (voxel) data; ensuring that the sample database captures the allowable degree of variation of the images representing the same material class; ensuring that the numbers of samples stored in the database reflect the relative numbers of appearances of the respective material classes in the "real world"; computing and representing all feature values, and derived data, for all sample data in the database and an unclassified material; defining, computing and comparing histograms / distributions of feature values; performing all data processing, analysis, comparison and classification steps in a multi-scale / multi-frequency framework; calculating an overall "probability tuple" $((P_0,)\ P_1, \ldots, P_c)$ for an unclassified material, where a P_{cl}-value indicates how probable it is that the material belongs to class cl.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

● The P_{cl}-values should not be understood as "probabilities in a pure and strict statistical sense"; the nature of these P_{cl}-values — i.e., their definition and complex computation — does not comply with all requirements applicable to a standard notion of probability. Regardless, we will continue to view P_{cl}-values as probability values in our material classification context: P_{cl}-values serve as "near-optimal indicators" for class-membership of an unclassified material.

⇒ **GOAL**: Given the $\mathbb{P} = (P_0, P_1, \ldots, P_C)$-tuple for an unclassified material, the goal is the maximization of the probability of the material's classification as a material of the correct class cl, $cl \in \{1, \ldots, C\}$; class 0 is understood as the class of all materials that are not class-1, ..., class-C materials. Alternatively, one can state that the goal is to maximize the number of correct classifications (and minimize the number of incorrect classifications), when classifying a set of unclassified materials.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

• Thus, one can describe the classification step as follows, on a rather abstract level:

> Given the $(C+1)$ P_{cl}-values for an unclassified material, where all P_{cl}-values are non-negative real numbers, map the tuple $(P_0, P_1, \ldots, P_C)$ to a binary decision tuple $(b_0, b_1, \ldots, b_C)$, where $b_{cl} \in \{0,1\}$ and one b_{cl}-value is 1 and all other b_{cl}-values are 0. The bit-value 1 indicates that the unclassified material is classified as a representative of the class associated with the 1-bit.
> Further, for a set (or "stream") of tuples $(P_0, P_1, \ldots, P_C)$, the formula (or algorithm) used to determine the binary decision tuples $(b_0, b_1, \ldots, b_C)$ must maximize the number of correct material classifications.

• Note. It is possible to ensure that all P_{cl}-values satisfy $0 \le P_{cl} \le 1$ via the use of the constant M, see formula on p. 6 dated 10/4/2022.

• Note. The term "STREAM" used above in the description of the classification step is important: "Streamed data" or "data streams" make it possible to continually refine and improve classification.

•••

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks:...

• Note. When the classification system is used in a real-world setting, it effectively must process a never-ending stream of unclassified materials. One must assume that it is highly desirable — and most likely necessary — that the system continually adapts to changing conditions over time. Examples of conditional changes include: (i) a material class(es) is (are) no longer of relevance and can be removed from the set of classes to be recognized; (ii) a previously not considered material class(es) is (are) at some point of relevance and must be added to the set of classes to be recognized; and (iii) the composition, and thus the resulting features, of a relevant material(s) changes with time and one must update the sample database etc. accordingly. Further, the numbers of / ratios of material classes change over time, and such a statistical change in class occurrences has an impact on classification performance and must therefore lead to adaptation. Moreover, whenever the system mis-classifies a material, the system should be refined (or, ideally, should automatically refine itself).

...