■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... **•Note.** The three-dimensional example discussed generated three normalized eigenvectors that are NOT mutually orthogonal to each other: $\mathbf{e}_1 = (-\sqrt{2}/2,\ 0,\ \sqrt{2}/2)^T$, $\mathbf{e}_2 = (-\sqrt{2}/2,\ \sqrt{2}/2,\ 0)^T$ and $\mathbf{e}_3 = (\sqrt{3}/3,\ \sqrt{3}/3,\ \sqrt{3}/3)^T$. The vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ are not orthogonal to each other. As a consequene, these three eigenvectors do not define three mutually perpendicular "edge directions" of a (minimal) bounding box, a cuboid. For a variety of reasons, orthogonality of basis vectors should be assured. In the case of this simple example, one can keep $\mathbf{e}_1$ and $\mathbf{e}_3$ as they are a pair of orthonormal vectors — and compute the needed third vector $\mathbf{e}_2$ by applying Gram-Schmidt orthonormalization to this vector's original coordinate values. By performing the needed operations, one obtains $\mathbf{e}_2 := \mathbf{e}_2 - \langle \mathbf{e}_2, \mathbf{e}_1 \rangle \mathbf{e}_1 - \langle \mathbf{e}_2, \mathbf{e}_3 \rangle \mathbf{e}_3 = (-\sqrt{2}/4,\ \sqrt{2}/2,\ -\sqrt{2}/4)^T$. Subsequently normalizing this vector yields the result $\mathbf{e}_2 := \mathbf{e}_2 / \| \mathbf{e}_2 \|$, which is $\mathbf{e}_2 = (-\sqrt{6}/6,\ \sqrt{6}/3,\ -\sqrt{6}/6)^T$. Using this vector as designate gradient of the polynomial F_2, one obtains $F_2 = -\sqrt{6}/6\, X_1 + \sqrt{6}/3\, X_2 - \sqrt{6}/6\, X_3$. Evaluating this polynomial at the four given points $\mathbf{x}_i$, $i = 1 \ldots 4$, produces the extrema $F_2^{min} = -2/3 \sqrt{6}$ and $F_2^{max} = 4/3 \sqrt{6}$. The associated radius is $2 r_2 = F_2^{max} - F_2^{min} = 2\sqrt{6} \Rightarrow r_2 = \sqrt{6}$. The origin becomes $\bar{\mathbf{o}} = \frac{1}{2} \sum_{d=1}^{3} (F_d^{min} + F_d^{max})\, \mathbf{e}_d = (1/3,\ 4/3,\ 1/3)^T$.

•••

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... Based on the local origin $\overline{\mathbb{O}}$ and the three orthonormal vectors $\mathbb{e}_1$, $\mathbb{e}_2$ and $\mathbb{e}_3$, we can now map the original points $\mathbb{x}_i$.

• The first mapping $\mathbb{x}_i \mapsto \mathbb{x}_i - \overline{\mathbb{O}}$ yields

$$\mathbb{x}_1 \mapsto (0,0,0)^T - (1/3, 4/3, 1/3)^T = (-1/3, -4/3, -1/3)^T,$$
$$\mathbb{x}_2 \mapsto (4,0,0)^T - \overline{\mathbb{O}} = (11/3, -4/3, -1/3)^T,$$
$$\mathbb{x}_3 \mapsto (0,4,0)^T - \overline{\mathbb{O}} = (-1/3, 8/3, -1/3)^T,$$
$$\mathbb{x}_4 \mapsto (0,0,4)^T - \overline{\mathbb{O}} = (-1/3, -4/3, 11/3)^T.$$

• The second mapping requires us to compute scalar products between the vectors $\mathbb{e}_1$, $\mathbb{e}_2$ and $\mathbb{e}_3$ and the vectors obtained after the first mapping. We can compute these scalar products via matrix multiplication: The first matrix has $\mathbb{e}_1$, $\mathbb{e}_2$ and $\mathbb{e}_3$ as its rows, and the second matrix has the four vectors resulting from the first mapping as its columns, i.e.,

$$\begin{bmatrix} -\sqrt{2}/2 & 0 & \sqrt{2}/2 \\ -\sqrt{6}/6 & \sqrt{6}/3 & -\sqrt{6}/6 \\ \sqrt{3}/3 & \sqrt{3}/3 & \sqrt{3}/3 \end{bmatrix} \begin{bmatrix} -1/3 & 11/3 & -1/3 & -1/3 \\ -4/3 & -4/3 & 8/3 & -4/3 \\ -1/3 & -1/3 & -1/3 & 11/3 \end{bmatrix} = \begin{bmatrix} 0 & -2\sqrt{2} & 0 & 2\sqrt{2} \\ -1/3\sqrt{6} & -\sqrt{6} & \sqrt{6} & -\sqrt{6} \\ -2/3\sqrt{3} & 2/3\sqrt{3} & 2/3\sqrt{3} & 2/3\sqrt{3} \end{bmatrix}.$$

• The third mapping divides the three rows of the matrix obtained via the second mapping by $r_1 = 2\sqrt{2}$, $r_2 = \sqrt{6}$ and $r_3 = 2/3\sqrt{3}$, respectively. The result matrix is

$$\begin{bmatrix} 0 & -1 & 0 & 1 \\ -1/3 & -1 & 1 & -1 \\ -1 & 1 & 1 & 1 \end{bmatrix}.$$

The points have local coordinates $\mathbb{u}_1 = (0, -1/3, -1)^T$, $\mathbb{u}_2 = (-1,-1,1)^T$, $\mathbb{u}_3 = (0, 1, 1)^T$, $\mathbb{u}_4 = (1,-1,1)^T$.

•••

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks: ...

• **Note.** The covariance matrix C defined by the four points x_i used in this example leads to a degenerate case: One eigenvalue has multiplicity 2 ($\lambda_1 = \lambda_2 = 16$). This fact implies that the eigenvectors associated with λ_1 and λ_2 are not – do not necessarily have to be – orthogonal to each other. In such a degenerate situation it is necessary to employ an additional orthonormalization step (Gram-Schmidt orthonormalization, for example), to ensure that the set of unit basis vectors in the set $\{\mathbf{e}_d\}_{d=1}^{D}$ is indeed an orthonormal basis. Further, one must "robustly" handle singular, degenerate cases of pointsets that lead to radius values $r_d = 0$. In such situations, one should define $r_d := \varepsilon$, and also establish a non-zero basis vector(s) for the respective radius(ii) r_d.

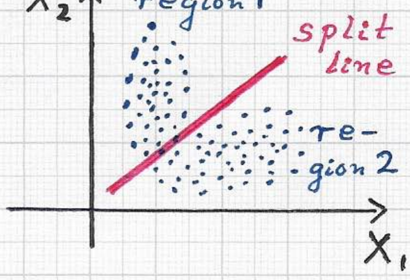
In summary, the described linear coordinate transformation performed in D-dimensional space must

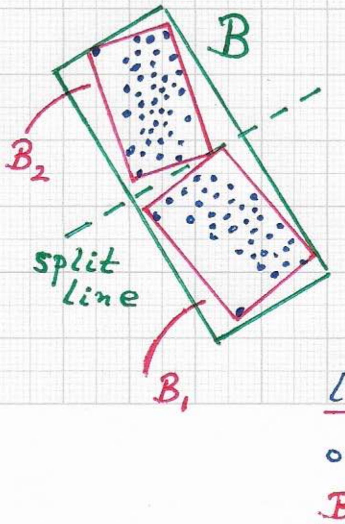
- have numerical stability that handles all cases of singular, degenerate point configurations,
- always generating a D-dimensional bounding box with an orthogonal coordinate system "at its center" – and radii > 0 for all D directions/dimensions.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... ■ **SUBDIVISION/SPLITTING.** In a complex real-world setting, the point data set $\{x_i\}$ is likely a data set not "nicely bounded" by an ellipse/ellipsoid/hyper-ellipsoid. Therefore, one must employ a scheme — an algorithm and data structure — supporting a representation of the "space occupied by $\{x_i\}$" via the use of several (sub-) regions, see left figure. It is important to consider the "meaning" of a point set $\{x_i\}$: The points in this set belong to a specific class — representing a specific class at some derived, high level of abstraction. Given a new, not-classified point, one must decide, with a very high level of probability, whether this new point belongs to / does not belong to the class represented by the discrete set $\{x_i\}$. The described me- for making the necessary decision is based on a linear coordinate transformation that maps all points, by considering covariance behavior, in their (minimal) bounding box to "local box coordinates." The example shown in the left figure sketches a scenario where the original box B is split into boxes B_1 and B_2 — implying two coordinate transformations.

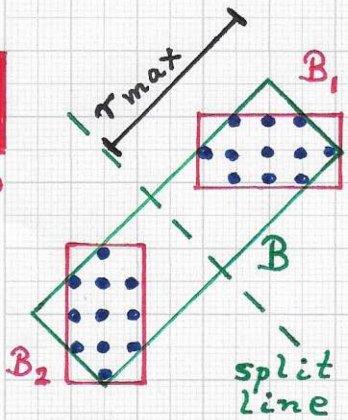




...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... Thus, one must design an efficient SUBDIVISION/SPLITTING approach that separates a point set into two subsets. First, one must determine whether a subdivision should be performed; second, in order to split the data set, one must execute the replacement of the respective bounding box into two bounding boxes, thereby replacing one local box coordinate system by two local box coordinate systems – with two different local linear coordinate transformations. One should subdivide boxes (separate point sets) until each box (and its associated point set) satisfies a specific condition – defining "how well a box represents the region implied by the box-associated points." Intuitively, it seems generally appropriate to split a box in the/a direction of maximal box radius. Considering the fact that such a process is based on recursive binary splitting, a BINARY TREE is an appropriate data structure to be used. The boxes (and associated point subsets and linear coordinate transformations) at the leaf level in the tree define the desired representation. The left figure illustrates an r_{max}-based split.



...