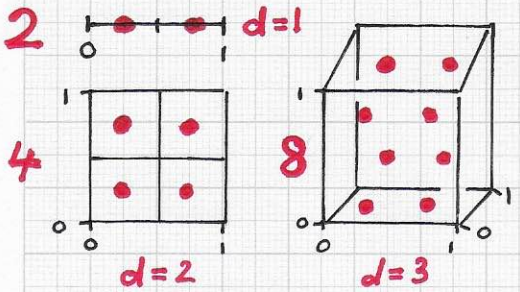
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

"Data becomes increasingly sparse with increasing dimensionality":



Simple example: Unit hypercubes of dimensions 1, 2 and 3. The numbers of associated point data are 2^1, 2^2 and 2^3." This "ideal progression" of the numbers of point data is practically impossible when high-dimensional domains must be discretely sampled.

Thus, one must consider volume – of a hyper-cube or hyper-ball – as a function of dimension and radius. The formulas for volumes of a d-dimensional ball are not commonly needed, and we mention them here:

$$V_{2i} = \frac{1}{i!}\,\pi^i\,r^{2i},$$

$$V_{2i+1} = \frac{2^{2i+1}\,i!}{(2i+1)!}\,\pi^i\,r^{2i+1},$$

$$i = 0, 1, 2, \ldots .$$

The table (right) lists a few specific values.

It is difficult to define a "good value" for k (the number of sample data in a local neighborhood) when applying a localized version of a Shepard probability function to a d-dimensional domain, i.e., evaluating a d-variate Shepard function. In this context, it is important to keep in mind how, for example, the d-dimensional volumes of d-dimensional hyper-cubes and hyper-balls of radius r increase with increasing dimension d:

d	V(□)	V(○)
0	1	1
1	r	$2r$
2	r^2	πr^2
3	r^3	$\frac{4}{3}\pi r^3$
4	r^4	$\frac{1}{2}\pi^2 r^4$
5	r^5	$\frac{8}{15}\pi^2 r^5$
6	r^6	$\frac{1}{6}\pi^3 r^6$

V(□) is the volume of a hyper-cube; V(○) is the volume of a hyper-ball. Generally, volume is a function of the form $V = \text{const} \cdot r^d$. In our application, $1 \le d \le H \cdot (C+1)$.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks...

Rapid growth of k-values with increasing number of dimensions. For example, we consider the following functions:

i) $k = 2 + 2d$,

ii) $k = 2 + 2d^2$,

iii) $k = 2 + 2^d$, $d = 1, 2, 3, \ldots$

These definitions produce these sequences of k-values:

d	i)	ii)	iii)
1	4	4	4
2	6	10	6
3	8	20	10
4	10	34	18
5	12	52	34
6	14	74	66
7	16	100	130
8	18	130	258
9	20	164	514
10	22	202	1026

The progression of the k-values listed in this table emphasizes that the number of available p_{ci}^{sc} variables (=d) must be small, i.e., one must select a "perfect" subset of all p_{ci}^{sc} variables.

Thus, it is imperative for efficiency to keep the value of k as small as possible, while still ensuring that classification performance is acceptable. Ultimately, the maximal value for k is the total number of all available classified sample data – when the Shepard function becomes a global Shepard function. Thus, one optimization objective is the "minimal and thus optimal selection" of the available p_{ci}^{sc} variables.

For example, one can consider the following formulas for setting a k-value:

→ polynomial: $k = k_0 + k_1 d^p$

→ exponential: $k = k_0 + k_1^d$

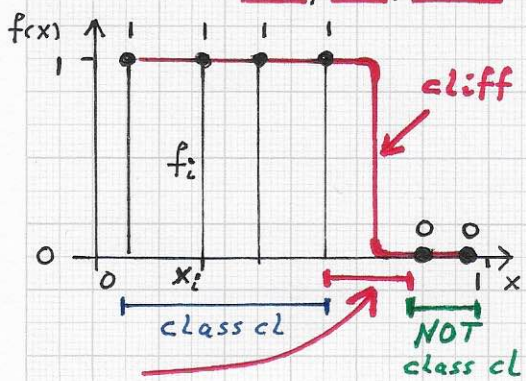
Here, k_0 is a constant, ensuring that minimally k_0 local data are used. Of course, polynomial k-functions with large p-values are not viable for large d-values. Generally, an exponential k-function cannot be used.

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Optimization of exponents of distance functions of Shepard function:



cliff region = most important region

steepness and location of cliff crucial "design parameters"

The cliff region represents the INTERFACE (region), i.e. the region where classification characteristics for class-cl and NOT-class-cl behaviors change, transition, are closely together. Modeling the Shepard function well in this region is most important for good classification performance.

Discrete optimization would determine the values of n_i of the function

$$f(x) = \frac{\sum_{i=1}^{N} f_i / (d_i^2)^{n_i}}{\sum_{i=1}^{N} 1 / (d_i^2)^{n_i}},$$

$n_i \in \{1, 2, 3, \ldots\}$.

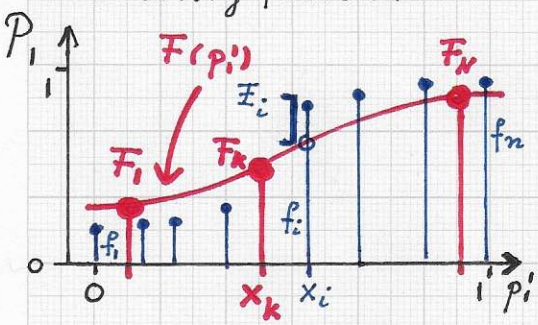
The left figure illustrates the most relevant aspect of the optimization of a Shepard probability function – for a simple synthetic univariate case. Here, sample data belong to a class cl (1) or not (0). The shown scenario has (only) one region / subinterval in the domain where the "cliff" region must be modeled by the Shepard function. The optimization objective is the maximization of correct class-cl and NOT-class-cl decisions for unclassified data in the cliff region, i.e., minimizing the errors made when classifying unclassified data as belonging to class cl or NOT class cl, with a certain probability calculated via the Shepard function (using local data).

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Simplest, univariate scenario of defining a CLASS-specific probability function:



Ideally, best approximations F of CLASS-specific, overall probability functions P_{cl} are computed LOCALLY, using all known sample data in a local neighborhood. This must be based on highly efficient computations. Thus, $N \ll n$.

The best approximation is a Shepard function, written in terms of radial basis functions, RBFs:

$$F(p_l') = \sum_{k=1}^{N} \frac{F_k}{d_k^2} \Big/ \sum_{j=1}^{N} \frac{1}{d_j^2} = \sum_{k=1}^{N} F_k \cdot w_k(p_l').$$

Thus, the equation system for the unknown F_k-values is

$$F(x_i) = \sum_{k=1}^{N} \ldots = f_i, \quad i = 1, \ldots, n.$$

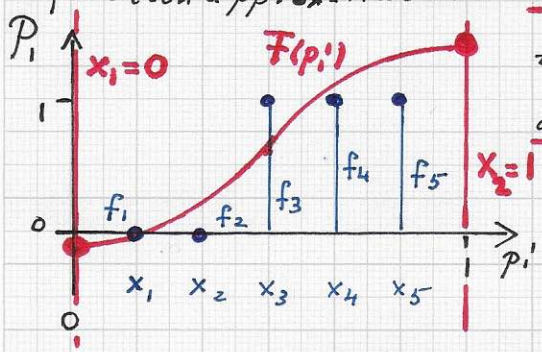
We now describe an approach that derives a Shepard function model using best approximation for the discrete setting. The main idea is discussed for a simple univariate setting sketched in the figure (left). The goal is to compute a best model function for one CLASS-specific probability function P_l (for class l) that merely depends on one probability value for class l, scale l, i.e., p_l'. The given data is the discrete set $\{(x_i, f_i)\}_{i=1}^{n}$, where $x_i \in [0,1]$ is a specific p_l'-value and $f_i \in [0,1]$ is the associated overall CLASS-specific probability P_l. The unknown function $P_l(p_l')$ is modeled via the best approximation function/model $F(p_l')$. The function F is a Shepard function for defined knots x_k, $k = 1, \ldots, N$, and to-be-computed coefficients (= Shepard function values) F_k. The values f_i must be approximated optimally by F, i.e., F must minimize the value $\sum_i E_i^2$. ...

Stratovan

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Computing a best Shepard function approximation:



• Given data: $x_i = i/6$, $i=1...5$;

$f_1=f_2=0$; $f_3=f_4=f_5=1$

• Resulting overdetermined linear system:

$$\begin{bmatrix} 25/26 & 1/26 \\ 4/5 & 1/5 \\ 1/2 & 1/2 \\ 1/5 & 4/5 \\ 1/26 & 25/26 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$W \quad F = f$$

$\Rightarrow$ solve $W^T W F = W^T f$:

$$\begin{bmatrix} 31367 & 10883 \\ 10883 & 31367 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 12480 \\ 38220 \end{bmatrix}$$

$\Rightarrow F_1 = -225771/8665348$

$F_2 = +10636509/8665348$

$\Rightarrow F_1 \approx -0.026$

$F_2 \approx +1.227$

$\Rightarrow F_1$ associated with $x_1=0$;

F_2 associated with $x_2=1$.

$\Rightarrow F(p_1') = -0.026\, w_1(p_1') + 1.227\, w_2(p_1')$.

Referring to the figure from the previous page, the individual equations involved in the computation of the best Shepard function approximation are:

$$\sum_{k=1}^{N} F_k \cdot w_k(x_1) = f_1$$
$$\vdots \qquad\qquad \vdots$$
$$\sum_{k=1}^{N} F_k \cdot w_k(x_n) = f_n$$

$$\Leftrightarrow \begin{bmatrix} w_1(x_1) & \dots & w_N(x_1) \\ \vdots & & \vdots \\ w_1(x_n) & \dots & w_N(x_n) \end{bmatrix} \begin{bmatrix} F_1 \\ \vdots \\ F_N \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}$$

$$\Leftrightarrow \quad W \quad F = f.$$

To avoid singular behavior of the weight function, we must ensure that $x_k \neq x_i$ – for all values of k and i. The unknown coefficient values for F_k are defined by the least squares solution

$$F = (W^T W)^{-1} W^T f.$$

This solution minimizes the value of the squared-error sum $\sum_{i=1}^{n} E_i^2 = \sum_{i=1}^{n} \left(f_i - \sum_{k=1}^{N} F_k\, w_k(x_i) \right)^2$. In our application, it is relevant that this value is smaller than a data-dependent threshold ε. If this is not the case, one will have to insert an additional knot(s) x_k.

...