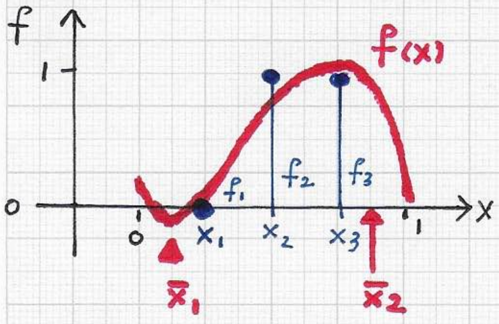
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Simple example of the use of the TPS method (without polynomial precision) for best approximation (see example on p. 13, 8/16/2022):



• Given data: $x_1=\frac{1}{4}, x_2=\frac{1}{2}, x_3=\frac{3}{4}$; $f_1=0, f_2=f_3=1$

• Knots: $\bar{x}_1 = 1/8$, $\bar{x}_2 = 7/8$

• TPS $f(x) = \sum_{i=1}^{2} c_i r_i(x)$, where $r_i(x) = (x-x_i)^2 \cdot \log|x-x_i|$

• NOTE: log IS $\log_e$.

• Over-determined linear system (see right):

$R\mathbb{c} = \mathbb{f} \Rightarrow \mathbb{c} = (R^T R)^{-1} R^T \mathbb{f}$

$\ldots \Rightarrow \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} -6.211576 \\ 0.406387 \end{bmatrix}$

$\Rightarrow f(x) = -6.211576\, r_1(x) + 0.406387\, r_2(x)$

! $\Rightarrow$ The TPS and MC methods produce significantly different results.

Hardy's MC and the TPS methods both use radial basis functions (RBFs). Nevertheless, since the MC and TPS basis functions are different, their best approximation results are generally also (rather) different – when comparing best approximation results for the same "input data." For a comparison, we use the "input data" from the example on page 13 (8/16/2022) to generate a best TPS approximation. It is important to point out that the log function in the TPS is the NATURAL log function $\log_e$. ! The two basis functions in the example are

$r_1(x) = (x-1/8)^2 \log|x-1/8|$ and

$r_2(x) = (x-7/8)^2 \log|x-7/8|$.

The three equations defining the best approximation problem are

$$f(x_j) = \sum_{i=1}^{2} c_i r_i(x_j) = f_j, \quad j=1,\ldots,3.$$

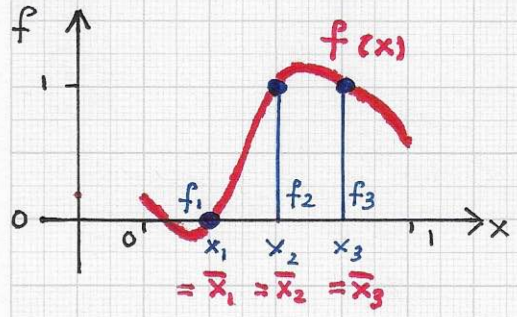
$$\Longleftrightarrow \begin{bmatrix} r_1(x_1) & r_2(x_1) \\ r_1(x_2) & r_2(x_2) \\ r_1(x_3) & r_2(x_3) \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}.$$

$$\Longleftrightarrow \quad R \cdot \mathbb{c} = \mathbb{f}.$$

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

Simple example of interpolation done via TPS method:



• Given data: $x_1=\frac{1}{4}$, $x_2=\frac{1}{2}$, $x_3=\frac{3}{4}$
$f_1=0$, $f_2=f_3=1$

• Knots: $\bar{x}_i = x_i$, $i=1\ldots3$

• Interpolation conditions:

$$f(x_j)=\sum_{i=1}^{3} c_i r_i(x_j) = f_j, \; j=1\ldots3$$

System in matrix form:

$$\begin{bmatrix} 0 & \log 2 & 2\log 2 \\ \log 2 & 0 & \log 2 \\ 2\log 2 & \log 2 & 0 \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 0 \\ -8 \\ -8 \end{bmatrix}.$$

$$\Rightarrow \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \frac{1}{\log 2} \begin{bmatrix} -6 \\ 4 \\ -2 \end{bmatrix}.$$

$$\Rightarrow f(x) = {}^1\!/\!\log 2 \cdot \left(-6(x-{}^1\!/\!_4)^2 \log|x-{}^1\!/\!_4| + 4(x-{}^1\!/\!_2)^2 \log|x-{}^1\!/\!_2| - 2(x-{}^3\!/\!_4)^2 \log|x-{}^3\!/\!_4|\right).$$

As expected, the geometric shapes of the graphs of f(x) generated by 2-knot approximation and by 3-knot interpolation are very similar.

After computing all needed matrix component values, one obtains

$$\underbrace{\begin{bmatrix} \log{}^1\!/\!_8 & 25\log{}^5\!/\!_8 \\ 9\log{}^3\!/\!_8 & 9\log{}^3\!/\!_8 \\ 25\log{}^5\!/\!_8 & \log{}^1\!/\!_8 \end{bmatrix}}_{R} \cdot \underbrace{\begin{bmatrix} c_1 \\ c_2 \end{bmatrix}}_{\mathbb{C}} = \underbrace{\begin{bmatrix} 0 \\ 64 \\ 64 \end{bmatrix}}_{\mathbb{f}}.$$

Using the least-squares method, the resulting $(R^T R)\mathbb{C} = R^T \mathbb{f}$ system to solve is

$$\begin{bmatrix} 220.3128 & 126.7914 \\ 126.7914 & 220.3128 \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} -1316.9635 \\ -698.0419 \end{bmatrix}.$$

$$\Rightarrow \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \approx \begin{bmatrix} -6.211576 \\ 0.406387 \end{bmatrix}.$$

Thus, the functions f(x) generated by the MC and TPS methods are rather different. (Compare the sketches of their graphs on the previous pages.)

The left figure sketches the scenario where the **TPS** method is used for interpolation. The example in this figure is merely provided to show the two f(x) results obtained via best **TPS** approximation using two knots (see previous page) and via interpolation using three knots $\bar{x}_i = x_i$, $i=1\ldots3$.

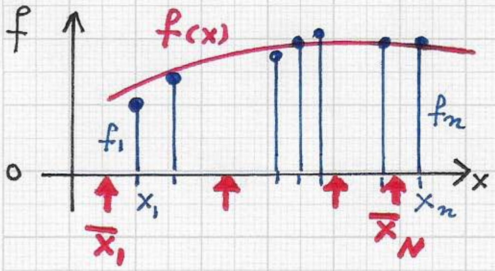
...

Stratovan

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

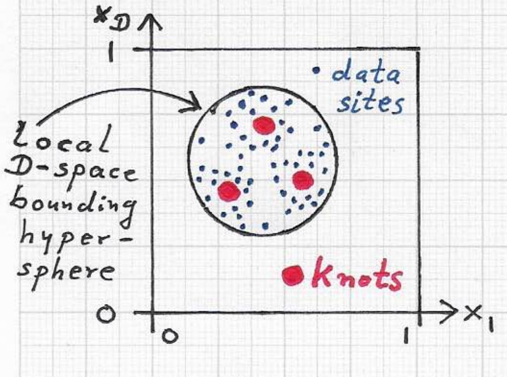
• Data sites vs. knots:



• Best approximation:

$$f(x) = \sum_{i=1}^{N} c_i\, r_i(x)$$

• Bivariate/multi-variate setting, data sites vs. knots of basis functions:



The number of given data sites (n), the (local) densities of data sites and the locations of the data sites should drive the definition of the N knots to be placed. For example, the knot locations should qualitatively reflect the (local) densities of the given sites. Thus, knot placement can be treated as yet another optimization problem.

Best approximation of given function values f_j at sites x_j (or $\mathbb{x}_j$ in the multivariate case), $j=1,\ldots,n$, using a radial basis function (RBF) approach, makes it necessary to define the number N ($N \ll n$) and locations/positions of the centers, the **KNOTS**, of the basis functions $r_i(x)$ (or $r_i(\mathbb{x})$), $i=1,\ldots,N$. (See, for example, "Knot selection for least squares thin plate splines," by J. R. McMahon and R. Franke, SIAM J. Sci. Stat. Comput. 13(2), pp. 484–494, 1992.) Knot selection is rather important in the context of computing probability values for data characterization and classification: A lower value of N leads to more efficiency, and a "good placement" of the knots $\overline{x}_i$ (or $\overline{\mathbb{x}}_i$) leads to better approximation behavior of the eventual best approximation, the **MODEL** function $f(x) = \sum_{i=1}^{N} c_i\, r_i(x) + a_0 + a_1 x + a_2 x^2 + \ldots$ (or $f(\mathbb{x})$).

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS - Cont'd.

• Laplacian eigenfunctions and neural networks:...

Knot placement viewed as an optimization problem:

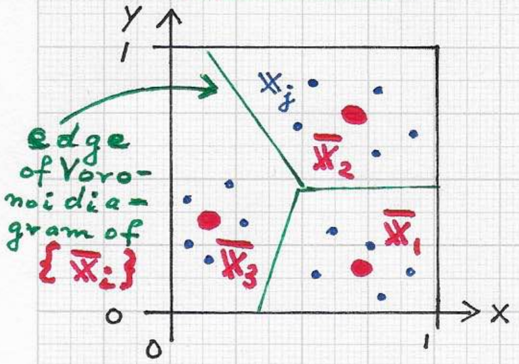


Figure shows a generic example for a unit hyper-cube domain (square). Given are the data sites

$x_j,\ j=1,\dots,n,$

and the goal is to place $N=3$ knots

$\overline{x}_i\ ,\ i=1,\dots,N,$

optimally. The optimization approach makes it necessary to define an appropriate cost/error function. A proper error function is

$$E=\sum_{j=1}^{n}\ \min_{i}\ \left((x_j-\overline{x}_i)^2+(y_j-\overline{y}_i)^2\right).$$

Here, the 2D case is used, i.e., a point x is the point $x=(x,y)^T$. The function E is the sum of squared distances between data sites and their nearest knots, respectively. The goal is to minimize the value of E.

The rationale motivating the construction and use of local best (least-squares) approximation functions for discrete, scattered probability data in the context of classification is the following: The given probability data are "uncertain" and are "not without errors." Errors can result from measurement steps, numerical calculations or human interventions. Further, for highly efficient processing, i.e., classification, a MODEL (model function) of a potentially very large number of discrete probability data can be readily used, evaluated, to perform a local classification computation - especially when local MODELS (model functions) are pre-computed and available in a database. Thus, the main goal for KNOT placement is preservation of data density: The local density/distribution of knots should reflect the local density/distribution of the original data sites.

...

OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

- Laplacian eigenfunctions and neural networks:...

  Simple iterative algorithm for knot placement converging to a local minimum of the error function $E(\bar{x}_1, \ldots, \bar{x}_N)$:

- Input: $\{x_j\}_{j=1}^{n}$, $N<n$
- Output: $\{\bar{x}_i\}_{i=1}^{N}$
- Algorithm:

  i) Compute initial values for all knots $\bar{x}_i$.

  ```
  /* Compute values ran- */
  /* domly, OR place knots */
  /* to obtain a distribu- */
  /* tion similar to that of */
  /* the sites, OR ...     */
  ```

  ii) For each site x_j determine its closest knot, i.e., determine the tile i of knot x_i that contains x_j.

  iii) Update the location of knot x_i by computing the centroid coordinates of all sites x_j residing in the (current) tile of x_i and assign this centroid as new location of knot x_i.

  iv) Termination: Perform steps ii) and iii) repeatedly until a state is reached when knot locations no longer change.

- Note. It is possible that knot locations are equal to site locations. The knot closest to a site might be ambiguous.

The figure on the previous page shows the edges of the **Voronoi diagram/tessellation defined by the knots $\bar{x}_i$.** This Voronoi diagram is intimately related to the chosen error function to be minimized, $E = \sum_{j=1}^{n} \min_i \left(\|x_j - \bar{x}_i\|^2 \right)$. This error function E is continuous and consists of quadratic pieces. A necessary condition for $E(\bar{x}_1, \ldots, \bar{x}_N)$ to be minimal is the requirement that the partial derivatives of E with respect to the D coordinates of the knots $\bar{x}_i$ ($\bar{x}_i$ is a point in D-dimensional space) are zero. As a consequence, the knot location of $\bar{x}_i$ is the centroid of the original data sites residing in the tile of knot $\bar{x}_i$. (**Note.** Consider the 1D case where locations $x_1, \ldots, x_n$ are given and only one knot is placed. Here, the error function E is defined as $E = \sum_{j=1}^{n} (x_j - \bar{x})^2$, with $\bar{x}$ being the only knot. The goal $E \rightarrow \min$ leads to $\frac{d}{d\bar{x}} E = -2 \sum_{j=1}^{n} (x_j - \bar{x}) = 0$. Thus, $\bar{x} = \frac{1}{n} \sum_{j=1}^{n} x_j$.)

...