PERIODICA POLYTECHNICA SER. CIV. ENG. VOL. 43, NO. 1, PP. 63–86 (1999)

GIS FUNCTIONS – INTERPOLATION[1]

Ferenc SÁRKÖZY

Department of Surveying
Technical University Budapest
H–1521 Budapest, Hungary
Phone: (36 1) 463 3212, Fax: (36 1) 463 3209
E-mail: sarkozy@altgeod.agt.bme.hu

Received : June 1, 1998

Abstract

The main strength of GIS (Geographical Information Systems) is the common analysis of compound spatial and attributive data. These latter are collected by samples. The values of the collected type of data can be expanded to the sites where no samples are available using interpolation methods. The interpolation can be performed either in the phase of data production or when the data are used for spatial analysis. In the second case the interpolation is a GIS function.

A brief review of interpolation methods is given with emphasis on the applicability of the particular method to one of the cases mentioned above.

The possibilities of two new methods – the wavelet transformation and the artificial neural network (ANN) approach are also discussed.

Especial attention is given to the possibilities of artificial neural networks in model building and interpolation.

Connection between the data model and ANN approach is examined. In context of ANN the links between interpolation, classification and GIS functions are explained.

As conclusions, several statements about GIS functions and modeling are given.

Keywords: GIS, GIS functionality, GIS tools, spatial data models, data production, interpolation, classification, artificial neural networks.

Introduction

In the paper (SÁRKÖZY, 1999) analyzing the recent situation and possible further development trends of GIS in context of the systematization of GIS functions I have recommended the more exact separation of data production from data analysis, that is the functions of data preparation from standard GIS functions.

However, the functions for expanding and deepening the target space can be used in both phases of spatial informatics.

At this point I should refer to my recommendation related to a new data model representing the natural phenomena (SÁRKÖZY, 1994). This data model called ‘function field’ assumes the determination of functions describing the phenomena. If the input data are compiled as functions and the GIS software can handle

[1]Research partially supported by the Hungarian Science Foundation Grant No. T 016487

this model, then the basic GIS analysis functions do not have to perform further interpolations.

That is from point of view of system principles the belonging of interpolation functions either to the data production functions or to the GIS analysis functions depends also on the data model of the particular GIS.

However, in both cases the importance of interpolation methods is continuously growing especially by connecting the GIS with some kind of modeling software.

1. The Role of Interpolation

More and more phenomena can be measured and might be involved in the spatial analysis. Among others we can mention the precipitation, temperature, soil parameters, ground water characteristics, pollution sources, vegetation data.

We are not able to measure the values of the particular phenomenon in all points of the sphere, but only in sample points. The interpolation gives us values in such points where we have no measurements.

The goodness of interpolation can be characterized by the discrepancy of the interpolated value from the true value. Because the true value is not known in general, we can select some measured points for testing the interpolation procedure.

In the stage of data production we can calculate the values of a particular phenomenon in predefined spots using interpolation procedures. For example if we want to deliver the data in a regular grid, but the samples are measured in scattered points we have to calculate the values of grid points from the samples using interpolation procedures.

In the phase of analysis we face another problem. We can have the different attributive data in different (regular) spots. For example we have two phenomena: temperature and precipitation (average and agglomerated for a month, respectively), given in two not coinciding grids. To perform the interaction computations we should transform the data to a common grid (to a common co-ordinate system) with the same discrete steps.

Seemingly similar tasks of interpolation are used for the creation of area objects representing a particular interval of the data in question. The construction of iso-lines or iso-surfaces in 3 dimensional case, however, aims to create and visualize aesthetic features and not too accurate ones (because of the generalisation of individual function values into interval membership, that is because of the artificial degradation of resolution the high accuracy in interpolation has no reason).

Discussing the role of interpolation we should pay attention to the *global* and *local* subdivisions of interpolation approaches. Even if we take into consideration that the global approaches turn into local ones by numerical solutions, we have to realize that the GIS requires *more* local (or *less* global) methods, while the data production *more* global (*less* local) procedures.

We have also to note that the GIS interpolation computations should be fast and desirably automatic without setting a large amount of parameters.

2. Interpolation Methods

There are different ways of classifying the interpolation methods. Probably my approach does not satisfy strict mathematical requirements, but attempts to clarify the issue for the wide community of people using GIS.

We will discuss the following groups of methods:

- Method of geometrical nearness;
- Statistical methods based on weighted average;
- Methods using basis functions;
- Method of artificial neural networks.

2.1. Method Based on Geometrical Nearness – the Voronoy Approach

As a consequence of involvement in research on geometric aspects of crystallography the Russian mathematician VORONOY Georgy Fedoszeevich (1868-1908) discovered in the first decade of the century a special continuous subdivision of the space by convex polyeders. In 2 dimensions the polyeders are turning into polygons called *Voronoy cells*.

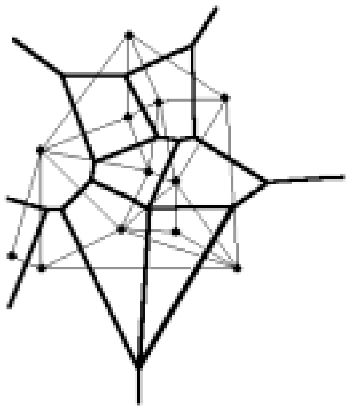

Fig. 1.

The geometric rule defining the cell is very simple: each cell has only one sample point and all other points inside the cell are closer to this sample point than to any other sample outside the cell.

To construct the Voronoy cell for a sample point we have to connect all data points to the selected one and to drop the bisecting perpendiculars to each link. The bisectors will intersect each other. Now we can select on each bisector two intersections the nearest to the sample point two determining the cell. In *Fig. 1* only those parts of the effective bisectors are shown which are forming the cells.

As interpolation method the Voronoy cells are mostly used for precipitation. Using the measured values of rain in the sample points the method extends their validity for the particular cell, that is, it supposes that these values are characteristic of the entire cell. This type of interpolation results in sudden changes of the function values on the borders of the cells. Therefore its use is limited.

However, at least two other applications make it worth speaking about it.

The Voronoy approach is suitable for dynamic spatial object condensation (W. YANG and Ch. GOLD, 1995) that is for realization of nested GIS implementation models that allow storage and processing of the objects at different levels of generalization.

The second application, the *Delone triangulation*, is directly related to our topic.

First we should consider that three points determine a plane. If we want to model the surface of the function with plane triangles fitted to the measured values in sample points, then we have to construct a continuous triangulation network on the points. However, if trying to connect the points, we will find that there are possible different triangulations for the same points. The different triangulations represent different interpolation surfaces and as a consequence different interpolated values in identical sites. That means that without unique triangulation the interpolation by plane triangles is useless.

In the thirties of our century, DELONE Boris Nikolaevich, professor of mathematics at the Moscow University, found out the dual task of the Voronoy subdivision. He proved that the field of scattered points could be uniquely transformed to a triangulated network using the links determined by the Voronoy cells of the points. With other words if we use the links belonging to the border lines of the cells (thin lines in *Fig. 1*), then the unique triangulation is established.

On the basis of this result work the TIN modules of the different GIS software.

For correctness we should mention that the interpolation method itself belongs to the third group (methods using basis functions), the geometrical partitioning, however, has so strong ties to the Voronoy cells, which gives reason to deal with both together.

2.2. Statistical Methods

The statistical methods interpolate the function value in the unknown point using weighted average of the known values in the sample points:

$$F^*(\mathbf{x}_0) = \sum_{i=1}^{n} \lambda_{i,0} \cdot F(\mathbf{x}_i) \ , \qquad (1)$$

where $F^*(\mathbf{x}_0)$ is the interpolated value in the place $\mathbf{x}_0$, $F(\mathbf{x}_i)$ is the value measured in the sample point $\mathbf{x}_i$, $i = 1, \ldots, n$, and numbers $\lambda_{i,0}$ are the weights.

As a rule the statistical methods are local, because they involve into the average only a number (n in *Eq.* (1)) of neighboring samples located usually inside a distance limit (e.g. *range* in the kriging) from the point to be interpolated.

All statistical methods have the property of smoothing the data reducing the sudden juttings and dips. As we pointed out (F. SÁRKÖZY and G. GÁSPÁR, 1992) *statistical methods could interpolate only values that are inside the interval between the largest and smallest sample value involved in the average building.*

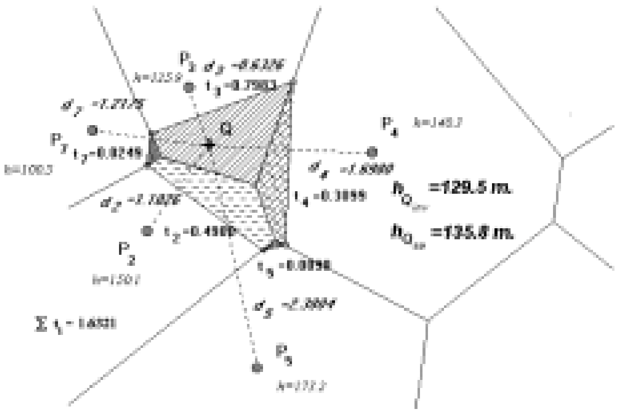

Fig. 2.

2.2.1. *Area Stealing Interpolation (Ch. Gold, 1991)*

This method, often called also as ‘natural neighbor interpolation’, determines the weights using the Voronoy tesselation.

In *Fig. 2* we constructed the Voronoy cells for the sample points $\boldsymbol{P}_1 \ldots \boldsymbol{P}_7$ with measured heights $h_1 \ldots h_7$ ($\boldsymbol{P}_1$ is outside the figure).We want to interpolate the $\boldsymbol{h}_Q$ height of the new point $\boldsymbol{Q}$.

To interpolate to a point we have to redefine the tesselation with the addition of the interpolation point. Now we have two tesselations, the original one and the new cell system constructed after inserting the new point. The cell of the new point (filled with patterns in *Fig. 2*) covers some parts of cells originally owned by particular sample points. These particular points will be involved into the interpolation of the new point. That is this method automatically selects from the sample points those which should take part in the interpolation. These points are called as natural neighbors.

The weights of the natural neighbors are nothing else but the areas which the new cell cuts out from the original cell owned by a particular neighbor. These areas

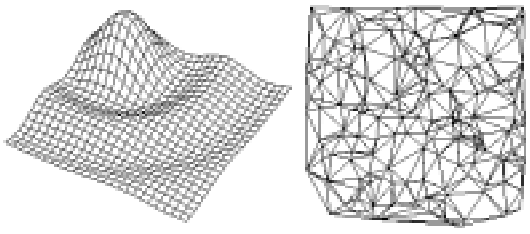

Fig. 3. a,b

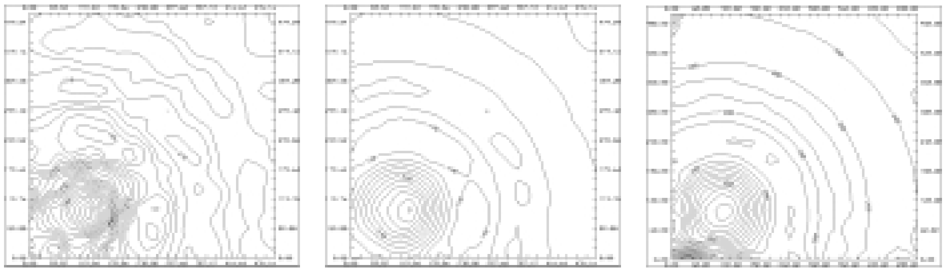

Fig. 4. a,b,c

are the ‘stolen areas’, denoted in *Fig. 2* by t_i. After computing the average we have got that $\mathbf{h}_Q = \mathbf{135.8}$.

2.2.2. *Weighting with Inverse Distances*

This simple method uses for weights some power of the inverse distance:

$$\lambda_{i,0} = \frac{\frac{1}{d_{i,o}^p}}{\sum_{i=1}^{n} \frac{1}{d_{i,0}^p}}, \tag{2}$$

where $p = 1, 2$ or 3.

If we select $p = 2$, and use the same subset of sample points which was determined by the stealing area method, then we get $\mathbf{h}_Q = \mathbf{129.5}$. The results are rather different; and less accurate is undoubtedly the latter one.

Moreover, the main disadvantage of this method lies in the arbitrary definition of the interpolation subset since the method itself does not generate the points to be involved in the interpolation.

2.2.3. *Kriging*

As a more sophisticated method some versions of ‘kriging’ can be used (for more details about kriging see some of the textbooks of geostatistics e.g. (E.H. ISAAKS,

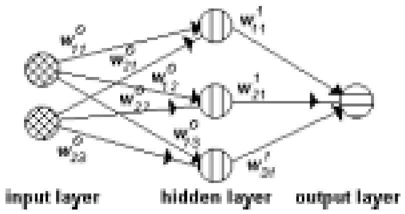

Fig. 5.

R. M. SRIVASTAVA, 1989)). Kriging estimates the unknown values with minimum variances **if** the measured data *fulfill some conditions of stationarity* (first order stationarity, second order stationarity, intrinsic stationarity). In the cases when no stationarity hypotheses can be stated the *u*niversal kriging can be used. This method, however, has not a unique solution and needs a large amount of interactive input and subjective assessment during the processing.

For the topic of our discussion a relatively new branch of kriging the *c*o-kriging has essential importance. This method uses one or more *secondary features* which are usually spatially correlated with the primary feature (e.g. heights secondary, rain primary). If the secondary features have more dense sample sets than the less intensively captured primary feature, with cokriging we can estimate with higher accuracy without any surplus expenditure.

The first step in kriging is the computation of a so called *experimental semi-variogram* using the following formula:

$$\gamma(h) = \frac{1}{2n(h)} \sum_{i=1}^{n(h)} [F(\mathbf{x}_i) - F(\mathbf{x}_i - \mathbf{h})]^2 \,, \tag{3}$$

where $\gamma(h)$ is the estimated semi-variance for the distance h, $n(h)$ is the number of measured point pairs in the distance class h, $F(.)$ is a measured value in $(.)$.

Eq. (3) is relatively easily computable if the measured points are ordered in a regular grid and the field has isotropy, that is $\gamma(h)$ depends only on h, but not on its direction. If the known points are located not regularly, distance classes have to be formed, and in the lack of isotropy different semi-variograms should be constructed in the typical groups of directions.

In the next step the experimental semi-variogram(s) should be approximated by some kind of functions fitted to the experimental data by means of the least squares method.

The real computation of weights used in (1) to interpolate the $Z(\mathbf{x}_0)$ unknown function value is performed by the solution of a system of $(n + 1)$ linear equations in the form as follows:

$$\Lambda_0 = \mathbf{K}^{-1}\mathbf{C}_0 \,, \tag{4}$$

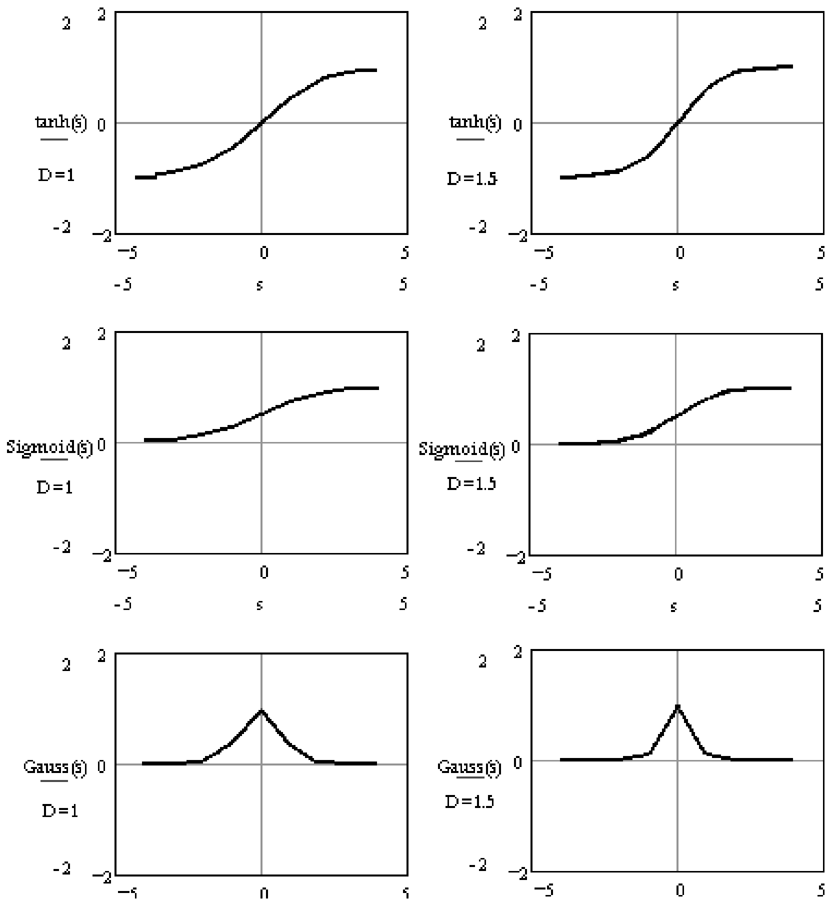

Fig. 6.

where $\Lambda_0^* = |\lambda_{1,0} \ldots \lambda_{n,0}|$, $\mathbf{C}_0^* = |c_{0,1} \ldots c_{0,n} 1|$ and the so called matrix Krige

$$\mathbf{K} = \begin{vmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1n} & 1 \\ c_{21} & c_{22} & c_{23} & \cdots & c_{2n} & 1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & \cdots & c_{nn} & 1 \\ 1 & 1 & 1 & \cdots & 1 & 0 \end{vmatrix} . \tag{5}$$

The vectors $\mathbf{C}_0$ are different for each new point to be interpolated, the coefficients $c_{i,j}$ have to be computed from the interpolated (theoretical) semi-variogram. The theoretical semi-variogram achieves its maximum value (or its 95%) at a distance H, called the *range* of the phenomenon. The coefficients belonging to points spaced on the range or farther become zero, the rest of them is calculated by the formula:

$$c_{i,j} = \gamma(H) - \gamma(h) \; ,$$

where h is the distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$.

As mentioned above, if the stationary hypothesis does not hold or with less sophisticated words if the phenomenon (e.g. the thickness of a coal stratum) has systematic changes in function of the location, then the systematic changes should be separated from the random ones. For this sake one can fit a plane or a surface of higher order to the sample data. This surface, of course, cannot pass through the points, it lies only near the points as a new reference frame. Now, the value of an interpolated point will consist of two parts: the systematic part that one computes from the equation of the fitted surface (often called *trend*), and the random part computed by kriging, related to the new reference frame. This complex method is called as universal kriging.

*C*o-kriging uses besides semi-variograms for each variable also cross-variograms for the variable couples. Let us consider the simple case of two variables. We have n sample points of the primary variable and m sample points of the single secondary variable (for simple writing we suppose $m = n+1$). For this case *Eq.* (4) and matrix (5) get the form as follows:

$$\mathbf{W}_0 = \mathbf{K}^{-1}\mathbf{V}_0 \; , \tag{4a}$$

where $\mathbf{W}_0^* = |\lambda_{1,0}s_{1,0} \ldots \lambda_{n,0}0s_{m,0}\mu_1\mu_2|$ (the notation $s_{i,0}$ stands for the weights of the secondary phenomenon, $\mu_1\mu_2$ are the Lagrange multipliers), $\mathbf{V}_0^* = |C_{0,1}Q_{0,1} \ldots C_{0,n}Q_{0,n}0Q_{0,m}11|$, and

$$\mathbf{K} = \left|\begin{array}{cccccccccc} c_{1,1} & 0 & c_{1,2} & cr_{1,2} & \cdots & c_{1,n} & cr_{1,n} & 0 & 0 & 1 & 0 \\ 0 & q_{1,1} & cr_{1,2} & q_{1,2} & \cdots & cr_{1,n} & q_{1,n} & 0 & q_{1,m} & 0 & 1 \\ \vdots & & & \vdots & & \vdots & & & & & \vdots \\ c_{n,1} & 0 & c_{n,2} & cr_{n,2} & \cdots & c_{n,n} & cr_{n,n} & 0 & 0 & 1 & 0 \\ 0 & q_{n,1} & cr_{n,2} & q_{n,2} & \cdots & cr_{n,n} & q_{n,n} & 0 & q_{n,m} & 0 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{m,1} & 0 & q_{m,2} & \cdots & 0 & q_{m,n} & 0 & q_{m,m} & 0 & 1 \\ 1 & 0 & 1 & 0 & \cdots & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & \cdots & 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right| . \tag{5a}$$

The coefficients in (5a) have to be computed as follows: $c_{i,j} = \gamma^{\text{primary}}(H) - \gamma^{\text{primary}}(h)$; $q_{i,j} = \gamma^{\text{secondary}}(H) - \gamma^{\text{secondary}}(h)$, the same is valid also for the coefficients in capital letters ($Q_{0,i}$ and $C_{0,i}$) related to the interpolated point; $cr_{i,j} = \gamma^{ps}(H) - \gamma^{ps}(h)$ where γ^{ps} is the regular representation of the empirical cross-variogram:

$$\gamma^{\text{ps}}(h) = \frac{1}{2n(h)}\sum_{i=1}^{n(h)}[F^{\text{primary}}(\mathbf{x}_i) - F^{\text{primary}}(\mathbf{x}_i + \mathbf{h})][F^{\text{secondary}}(\mathbf{x}_i) -$$

$$- F^{\text{secondary}}(\mathbf{x}_i + \mathbf{h})] \, . \tag{6}$$

In this brief review we cannot discuss more details. However, we should underline that the formulas above are related to the theoretical case of absolute stationarity. In the real cases the global shape of the phenomena is first modelled with some kind of functions, usually polynomials. In this practical case these functions are also included into *Eq.* (4a) instead of the rows and columns with numbers one, which are ensuring that the sum of weights related to a particular feature equals the unity. For example if the primary feature's trend surface is the plain $f(\mathbf{x}) = A + Bx + Cy$, and the secondary feature is approximated with another plain $g(\mathbf{x}) = D + Ex + Fy$, then the formulas (4a) and (5a) should be overwritten in the following way:

$$\mathbf{W}_0 = \mathbf{K}^{-1}\mathbf{T}_0 \, , \tag{4b}$$

where the right hand side vector $\mathbf{T}_0^* = |C_{0,1}Q_{0,1} \ldots C_{0,n}Q_{0n}0Q_{0,m}f(\mathbf{x}_0)g(\mathbf{x}_0)|$, and the matrix Krige takes the form

$$\mathbf{K} = \begin{vmatrix} c_{1,1} & 0 & c_{1,2} & cr_{1,2} & \cdots & c_{1,n} & cr_{1,n} & 0 & 0 & f(\mathbf{x}_1) & 0 \\ 0 & q_{1,1} & cr_{1,2} & q_{1,2} & \cdots & cr_{1,n} & q_{1,n} & 0 & q_{1,m} & 0 & g(\mathbf{x}_1) \\ \vdots & & & \vdots & & \vdots & & & & & \vdots \\ c_{n,1} & 0 & c_{n,2} & cr_{n,2} & \cdots & c_{n,n} & cr_{n,n} & 0 & 0 & f(\mathbf{x}_n) & 0 \\ 0 & q_{n,1} & cr_{n,2} & q_{n,2} & \cdots & cr_{n,n} & q_{n,n} & 0 & q_{n,m} & 0 & g(\mathbf{x}_n) \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{m,1} & 0 & q_{m,2} & \cdots & 0 & q_{m,n} & 0 & q_{m,m} & 0 & g(\mathbf{x}_m) \\ f(\mathbf{x}_1) & 0 & f(\mathbf{x}_2) & 0 & \cdots & f(\mathbf{x}_n) & 0 & 0 & 0 & 0 & 0 \\ 0 & g(\mathbf{x}_1) & 0 & g(\mathbf{x}_2) & \cdots & 0 & g(\mathbf{x}_n) & 0 & g(\mathbf{x}_m) & 0 & 0 \end{vmatrix} . \tag{5b}$$

The short summary of the kriging methods shows that this approach has several drawbacks. First of all I should mention the ambiguities by handling the trends and the unisotropies. That is the estimates can be biased due to the inappropriate choice of the trend expression and the regular model for variogram approximation.

Another problem is that for creation of an empirical cross-variogram only those spots are used where both the oversampled secondary variable and the undersampled primary variable have measured values. For taking into consideration the entire information content of the oversampled variable in the cross-variogram construction several research projects are in progress.

The model is a local one, but even so the processing takes a significant run time. Especially the inversion of the huge matrices **K** can cause numerical troubles and memory overflows. A new research of (LONG and MYERS, 1997) aims to split the co-kriging matrix into simple kriging matrices of the involved phenomena and express the co-kriging weights by the inverses of these smaller matrices (and some other terms).

For the visualization (mapping) purposes a second method, usually spline interpolation has to be used. The interpolated points can be stored in regular grid structure that is convenient for data base manipulations and also for different computational purposes.

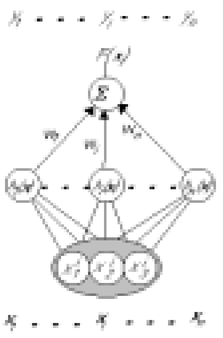

Fig. 7.

2.3. Methods Using Basis Functions

There are several methods that reconstruct the function using a linear combination of a set of basis functions with the closest fit to the samples. Depending on the type of the basis functions we can distinguish among others polynomial interpolation, splines with tension using radial basis functions, Fourier and wavelet transformations (F. SÁRKÖZY, J. ZÁVOTI, 1995). These latter ones transform the approximated function from the time or space domain to the frequency domain, that is the interpolation has to be made there. The numerical methods of transformation and inverse transformation, however, work for regularly located samples and in this form are unusable for interpolation of scattered points data. Only a few new research projects aim to solve the wavelet transformation of non-regularly distributed data. We have to wait some years until the wavelet interpolation could be involved in the solution of spatial interpolation problems.

2.3.1. Trend Surface Analysis

The GIS deals with spatial objects or spatial phenomena which are at least 2 dimensional, but the natural features are often modeled in 3 or 4 dimensions.

The TSA aims at the global interpolation of the phenomenon in question, it has no interest in detecting local irregularities. Of course, the global interpolation can be made also on different resolution level. The choice of this depends on the task to be performed and the behavior of the sample itself.

As we have seen in context of universal kriging the trend surface can be approximated by first order, second order, third order, in general n-th order polynomials. By choosing the order of the surface one should take into consideration the rule of thumb: any cross-section of a surface of order n can have at most $n - 1$ alternating maxima and minima.

If applied to the terrain, one can see in advance the main characteristics of the surface and choose the proper order. However, the surface of other phenomena (e.g. temperature) is not visible and one should make several attempts to find the appropriate order.

The polynomials are fitted to the sample data using the least squares adjustment (an alternative name of the fitting procedure is the regression).

This method minimizes the squares of differences of the given and interpolated values, that is denoting the i-th sample value by z_i, the interpolated value in the same point by $f(\mathbf{x}_i)$, the number of samples by n, the adjustment fulfills the condition: $\sum_{i=1}^{n}(f(\mathbf{x}_i) - z_i)^2 =$ minimum. Of course, the value of the minimum depends on the suitability of the chosen model for the particular data set. As a measure of the goodness of fitting we can use the quantity $m = \pm\sqrt{\frac{\sum_{i=1}^{n}(f(\mathbf{x}_i)-z_i)^2}{n-1}}$ called mean square error, abbreviated as MSE.

The flexibility of the polynomial depends on its order. If we choose higher orders, we can model more complicated surfaces. However, the higher the polynomial order, the more numerical problems might occur by risk of emerging huge, ill-conditioned matrices which should be inverted. As a conclusion for most of the practical tasks the order does not exceed $3 - 5$.

The method can be used also for three-dimensional modeling, however, in this case it was called as polynomial regression (COLLINS and BOLSTAD, 1996). The authors used for the estimation of the temperature t the expression

$$t = b + b_1x + b_2y + b_3x^2 + b_4y^2 + b_5xy + b_6x^3 + b_7y^3 + b_8x^2y + b_9xy^2 + b_{10}z\,.$$

In their opinion the difference between the TSA and the polynomial regression consists of the number of independent variables (the TSA as a surface can have only two) and the completeness of polynomial expansion (the TSA has complete expansion, the polynomial regression can be set up using the linear combination of independent functions). We can see that in the formula of the temperature the variable z is applied only in linear form.

The polynomial trend surface has the unpleasant behavior to increase or decrease very rapidly in the places where the density of samples is rare especially at the borders of the region.

2.3.2. Regularized Smoothing Spline with Tension

The first wide-spread commercial 3D modeling program package (SMITH and PARADIS, 1989) used the three-dimensional extension of the two-dimensional spline interpolation worked out by (BRIGGS, 1973). The original method defines a set of values at the points of a regular grid without finding first a continuous function of the space variables that takes the values measured at given positions. To cope with the problems of fitting two-dimensional piecewise polynomials to randomly located observation points Briggs solved the differential equation of a thin sheet that is equivalent to third-order spline. The solution was made numerically by the help of difference equations thus it puts the interpolated function values in a regular grid. The endproduct of the procedure was the drawing of contour lines; the intersections of the lines with the grid were interpolated by a four point cubic polynomial and then the intersections were joined by a cubic spline.

(MITAŠOVA and MITAŠ; MITAŠOVA et al., 1993) gave an explicit solution of variational conditions with direct estimation of first and second order derivatives for two-, three- and four-dimensional cases.

The regularized smoothing spline with tension is a radial basis function method for interpolation from scattered data. The interpolation is flexible through the choice of the tension parameter which controls the properties of the interpolation function and smoothing parameter which enables to filter out the noise.

The function derived by minimization of the squared smoothness functional containing all derivatives has the following form:

$$S(\mathbf{x}) = T(\mathbf{x}) + \sum_{j=1}^{N} \lambda_j R\left(\mathbf{x}, \mathbf{x}^{[j]}\right) \ , \tag{7}$$

where the index j denotes the measured points, λ_j the unknown coefficients, $R\left(\mathbf{x}, \mathbf{x}^{[j]}\right)$ the basis functions.

For the related cases $T(\mathbf{x}) = a_1 =$ constant. The basis functions for the two-, three- and four-dimensional cases are as follows:

$$R_2(r) = -[E_1(\rho) + \ln(\rho) + C_E] \ , \quad R_3(r) = \left[\sqrt{\frac{\pi}{\rho}}\mathrm{erf}(\sqrt{\rho}) - 2\right] \ ,$$

$$R_4(r) = \left(\frac{1 - e^{-\rho}}{\rho} - 1\right) \ ,$$

where $C_E = 0.577215...$ is the Euler constant, $E_1(.)$ is an exponential integral function, $r_j^2 = \sum_{i=1}^{d}\left(x_i - x_i^{[j]}\right)^2$, $\rho = \left(\frac{\varphi r}{2}\right)^2$, φ is an arbitrarily selected parameter, called 'tension' controlling the flexibility of the system,

$$\mathrm{erf}(x) \equiv \frac{2}{\sqrt{\pi}} \int_0^x e^{-x^2}\,\mathrm{d}x = \frac{2}{\sqrt{\pi}}\left(x - \frac{x^3}{3} + \frac{1}{2!}\frac{x^5}{5} - \frac{1}{3!}\frac{x^7}{7} \pm ...\right)$$

is the error function.
The $N+1$ unknown parameters $(\lambda_1, \dots, \lambda_N, a_1)$ are obtainable from the following system of $N+1$ linear equations, where $z^{[i]}$ is the measured value in the point $\mathbf{x}^{[i]}$;

$$a_1 + \sum_{j=1}^{N} \lambda_j R\left(\mathbf{x}^{[i]}, \mathbf{x}^{[j]}\right) = z^{[i]}, \quad i = 1, \dots, N \ , \sum_{j=1}^{N} \lambda_j = 0 \ . \tag{8}$$

In connection with this explicit method several questions require further analysis. Firstly the practical application of general global methods is problematical because of the long processing time growing proportional with the third power of the measured values. In (MITAŠOVA and MITAŠ, 1993) the authors recommend a segmented processing procedure that approximates the global method with several overlapping local ones.

From point of view of the GIS functions we should not forget that this global method does work in 3 dimensions, but its results are doubtful for most of natural phenomena, nevertheless its visualization possibilities are excellent.

2.3.3. Method of Local Polynomials

In 1992 the author of the present paper together with a young associate of the department of surveying Mr. P. GÁSPÁR published a simple interpolation method (SÁRKÖZY and GÁSPÁR, 1992). We can summarise the essence of our method as follows.

For each point $\mathbf{r}_i$ in which the value u_i of the in general unknown function $f(\mathbf{r})$ is given we can construct a $G_i(\mathbf{r})$ approximating function that is properly near to the original function in the neighborhood of the point. The function should be defined *everywhere* over the global model and give good approximation at least for the neighboring points. The simplest local function is the polynomial, in 3 D with three independent variables, or in the case of the given two-dimensional example a third-order polynomial with two independent variables: $z_k = z_i + a_1u_k + a_2v_k + a_3u_k^2 + a_4v_k^2 + a_5v_ku_k + a_6u_k^2v_k + a_7v_k^2u_k + a_8u_k^3 + a_9v_k^3$.

The co-ordinate transforms are as follows: $u_k = \text{arctg}\ (c(y_k - y_i))$, $v_k = \text{arctg}\ (c(x_k - x_i))$. For each polynomial $G_i(\mathbf{r})$ the coefficients a_j are calculated from the given values of the primary and (in case of need) secondary neighbors using weights considering the distance and the level of neighborhood as well.

The accuracy of approximation depends on the distances from the central point of the function. The discrepancies of the function values and the known data are computed using the expression $\delta_{ij} = G_i(\mathbf{r}_j) - u_j$. We can order the discrepancies into distance intervals and compute the squared dispersions belonging to each interval by the formula $\sigma_d^2 = \frac{1}{K}\sum_{i=1}^{K}\delta_i^2$, where K is the number of discrepancies in the particular interval.

In general for isotropic fields $\sigma_d^2(d) \approx \sigma_m^2 + a \cdot d^b + c$, $a, b, c \geq 0$, where σ_m is the dispersion of the function values in the nodal points. Parameters a, b, c can be computed from the related pairs d, σ_d. We can handle also the anisotropy and the special behavior of each local model.

From the dispersions we compute the weights $s(d) = \frac{\sigma_0^2}{\sigma_d^2}$, and as the weighted average (or linear combination of basis functions) the interpolated value of the function:

$$\hat{u}_r = \frac{\sum_{i=1}^{M} s_i \cdot u_{i,r}}{\sum_{i=1}^{M} s_i}\,. \tag{9}$$

For the sake of easier comparison the 500 m$\times$500 m simulated study area was covered by the regular function $z = 200 + 100\frac{\sin d}{d}$, with the notation $d = = \frac{\pi}{100}\sqrt{(y-100)^2 + (x-100)^2}$. First of all we computed the true values of the 20 m$\times$20 m grid points, as shown in part *a* of *Fig. 3*. After that, we randomly generated the co-ordinates of 200 spots and computed their heights (part *b*, *Fig. 3*).

Using the heights of the scattered points we interpolated the grid by the methods of inverse distances, kriging and polynomials (*Fig. 4*, parts *a*, *b* and *c* respectively).

2.4. Method of Artificial Neural Networks

The method of ANN is relatively, but absolutely new in the domain of spatial interpolation. This is the reason why we have to explain some basic ideas of this approach before discussing its application for our purposes.

2.4.1. What is an Artificial Neural Network?

The research of the human neural system showed that it could response to the signals from the sensing organs by a highly interconnected large system of relatively simple activation elements – the neurons. That is if we have a large number of interconnected simple processing elements we can solve very complex tasks.

This idea stimulated initially a small part of researchers in artificial intelligence to try to solve their tasks by artificial neural networks.

The theory and basic algorithms of ANN were worked out mostly in the last decade, software products have appeared since 1992–93, the first applications for spatial interpolation emerged in 1997.

There are different types of ANN architecture, however, for interpolation tasks it is satisfactory to discuss the default structure: the multilayer feedforward network.

In *Fig. 5* we sketched out a simple MLP (MultiLayer Perceptron) network (for the clarity we plotted only 1 hidden layer, however, the *m*ultilayer model allows arbitrary number of those).

As we can see, the structure consists of nodes (neurons or perceptrons) organized in layers and links. There are three principally different types of layers: the *i*nput layer (only one), the *h*idden layer (arbitrary number), and the *o*utput layer (only one).

The number of nodes in the input layer corresponds to the number of independent variables of the particular task, while the number of nodes in the output layer depends on the number of scalar functions involved in the common evaluation, or if we have to approximate a vector function, the output nodes can represent the components of the vector.

As a default, each node is connected to each node of the forward next layer. There are algorithms which can prune the ineffective links and there are network architectures with not only sequential but also backward (recurrent) links.

The nodes of hidden layers and output layers own activation functions which can differ from each other by nodes and/or by layer types. These functions are mostly nonlinear, easily differentiable functions as sygmoid or hyperbolic tangent or the Gaussian bell curve.

The formulae for these functions are as follows:

For the sygmoid $y = \frac{1}{1+e^{-Ds}}$; $D > 1$. The function values are in the domain $[0, 1]$, if $s = 0$, $y = 0.5$. The hyperbolic tangent is determined by the formula $y = \frac{1-e^{-Ds}}{1+e^{-Ds}}$; $D > 1$. The interval of output values is in the range $[-1, +1]$, if $s = 0$, $y = 0$. Nowadays the Gaussian activation function with the formula: $y = e^{-D^2s^2}$ gains more and more popularity. It has output values among 0 and 1, if $s = 0$, $y = 1$.

The diagrams for these functions for $D = 1$ and $D = 1.5$ are shown in *Fig. 6*. Pay attention that the 'useful' inputs (for which different inputs involve different outputs), range about $[(-2.7) - (+2.7)]$ and $[(-4) - (+4)]$, depending on D.

2.4.2. How Does an Artificial Neural Network Work

The objective of a particular ANN is to give desired outputs in response to the inputs of the user. For example if we want to interpolate the heights, we can choose for input locations and hope that the outputs of the network will contain the heights in the given spots. But how can the network find out the heights. This is made by learning from known input–output couples of data.

What can change in the network according to the information extracted from the known data, called training set? The answer is: weights denoted in *Fig. 5* by w_{ij}.

The procedure adjusting the weights to fit the given data is called training or learning. To understand this process first we should cast a glance at the way the network processes the input data.

For the network in *Fig. 5* the t-th sentence (record) of training data consists of the input vector x_j^t, $(j = 1, 2)$ and the desired output y^t. The input passing forward (from left to right) results in the output:

$$\begin{aligned} o_{21}^t \quad = \quad & f_1^2 \left[w_{11}^{1(t-1)} f_1^1 \left(x_1^t w_{11}^{0(t-1)} + x_2^t w_{21}^{0(t-1)} \right) + \right. \\ & + w_{21}^{1(t-1)} f_2^1 \left(x_1^t w_{12}^{0(t-1)} x_2^t w_{22}^{0(t-1)} \right) + \\ & \left. + w_{31}^{1(t-1)} f_3^1 \left(x_1^t w_{13}^{0(t-1)} x_2^t w_{23}^{0(t-1)} \right) \right] \end{aligned}$$

or

$$o_{21}^t = f_1^{(2)}\left[\sum_{i=1}^{3} w_{i1}^{1(t-1)} f_i^{(1)}\left(\sum_{j=1}^{2} x_j^t w_{ji}^{0(t-1)}\right)\right] . \tag{10}$$

In *Eq.* (10), $f_i^{(l)}$ denotes the activation function of the i-th node (from top to bottom) in the l-th layer ($l = 0, 1, 2$).

The output value o_{21}^t in general is not equal to the desired value y^t. To approach this equality we should perform a backward pass to adjust the network's weights. From this processing step has obtained this training method the name of *backpropagation*.

First we have to determine an expression characterizing the error in the output, called error function. This function has to be minimized in the training process. Usually the expression

$$E(w) = \frac{1}{2}\sum_{j=1}^{k}(y_j^t - o_j^t)^2 \tag{11}$$

is chosen for error function, where k is the number of output nodes (in our simple case $k = 1$).

One of the ways of minimizing a nonlinear objective function is the application of the so-called *g*radient descent method. The negative gradient $-\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}}$ is a vector that shows the *l*ocal direction of the descent. If we pass along these directions with small steps we can hope to reach the global minimum. Passing along the directions practically means that we change the weights (independent variables) proportionally to the components of the negative gradient. The step size η is called *learning rate* in the backpropagation algorithms. Thus

$$\mathbf{w}^{t+1} = \mathbf{w}^t + \Delta\mathbf{w} \tag{12}$$

and

$$\Delta\mathbf{w} = -\eta\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} . \tag{13}$$

Do not forget that in (13) both the error and the weights have the values computed in the t-th cycle.

The derivatives of a complex function have to be computed applying the chain law. For the derivative of E with respect to weights w_{ij} associated with the links between the i-th node of the last hidden layer and the j-th node of the output layer (in our example with respect to w_{11}^1, w_{21}^1, w_{31}^1) we have

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial o_j}\frac{\partial o_j}{\partial p_j}\frac{\partial p_j}{\partial w_{ij}} , \tag{14}$$

where the input of the j-th node of the output layer $p_j = \sum_i w_{ij} o_i$. The derivatives in (14) are as follows:

$$\frac{\partial E}{\partial o_j} = \frac{\partial\left(\frac{1}{2}(y_j - o_j)^2\right)}{\partial o_j} = -(y_j - o_j) ,$$

$$\frac{\partial o_j}{\partial p_j} = \frac{\mathrm{d} f(p_j)}{\mathrm{d} p_j} = f'(p_j) ,$$

$$\frac{\partial p_j}{\partial w_{ij}} = \frac{\partial \left(\sum_i w_{ij} o_i\right)}{\partial w_{ij}} = o_i .$$

Substituting the derivatives into (14) we get

$$\frac{\partial E}{\partial w_{ij}} = -(y_j - o_j) f'(p_j) o_i , \tag{14a}$$

thus, the weights between the last hidden layer and the output layer should be altered with the value

$$\Delta w_{ij} = \eta (y_j - o_j) f'(p_j) o_i = \eta \delta_j o_i . \tag{15}$$

The term $\delta_j = (y_j - o_j) f'(p_j)$ in *Eq.* (15) is called local error, and the expression itself is often cited as *delta rule*.

The terms in (15) are known from the training set (y_j), and from the forward pass of the t-th cycle. For example for the network in *Fig. 5*, o_j is computed from the entire formula (10), p_j is the term in square brackets in the same formula and $o_i = f_i^{(1)} \left(\sum_{j=1}^{2} x_j^t w_{ji}^{0(t-1)} \right)$.

The delta rule in form (15) is only suitable for the weights of connections ending in the output nodes because term δ_j is computed from the actual output o_j. To find out the changes of weights associated with links pointing to hidden layer we should 'backpropagate' the error.

Let us suppose we have n hidden layers. For computing the changes of weights w_{ij} associated with connections between the $(n-1)$, n layers (that is i is a node number in layer $(n-1)$, j in layer n) we have to express the local error in the node j of the n-th layer. For this sake we can use the already computed local errors of the output layer's nodes. Indeed, the error backpropagation is proportional to the local errors of the successor layer's nodes. Using index k for these nodes we can express the local error of nodes in a hidden layer

$$\delta_j^{\text{hidden}} = f'(p_j) \sum_k \delta_k w_{jk} . \tag{16}$$

Thus, the weight updates also in this case are computed by formula (15) but the local errors have to be considered as expressed in formula (16).

The most critical point for backpropagation is the choice of the learning rate η. It can be chosen from the range [0.05, 0.5]. If the rate is too small, the learning process is slow, if it is too large, the training oscillates around a local or global minimum. There are algorithms which can automatically control the learning rate in function of the slope of the error surface.

The backpropagation can be realized in two ways: the *on-line* backpropagation changes the weights after processing of each sample, the *batch* backpropagation performs the weight improvement only after the entire training set is processed. This latter works slower.

For improving the convergence of the learning process numerous modifications of the standard backpropagation are in use.

For elimination of flat spots on the error surface the *backpropagation with momentum* gives some hope. The idea of this method is that the weights are improved with a linear combination of the recent and former weight changes:

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} + \mu \Delta w_{ij}^{(t-1)} , \tag{17}$$

where the value of momentum μ lies in the interval [0, 1].

The *method of weight decay* tries to prevent the weights to grow too large. The weight updates for this method are computed as follows:

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} - \gamma \Delta w_{ij}^{(t-1)} , \tag{18}$$

coefficient γ ranges from 0 until 1.

The resilient backpropagation is essentially not a gradient descent method but somehow or other is similar to it. The method uses batch learning, that is the gradient is computed as the sum of the gradients belonging to each sample of the training set.

The method uses only the *signs* of the gradient and orders in *opposition* to those signs to the updates. The absolute value of the update factor depends on the relation of the actual and former gradient. If their signs are equal, this value is 1.2, if not, the value is 0.5. The absolute value of the update itself equals the product of the update factor and the absolute value of the former update. For the exception when the product of the new and old gradient is equal to nil, the new update has the same absolute value as the old one. The explanation above can be compressed in the following formulae:

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)}, & \text{if} \quad \dfrac{\partial E^{(t)}}{\partial w_{ij}} > 0 , \\[2ex] +\Delta_{ij}^{(t)}, & \text{if} \quad \dfrac{\partial E^{(t)}}{\partial w_{ij}} < 0 , \\[2ex] 0, & \text{else,} \end{cases} \tag{19}$$

$$\Delta_{ij}^{(t)} = \begin{cases} 1.2\Delta_{ij}^{(t-1)}, & \text{if} \quad \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} \dfrac{\partial E^{(t)}}{\partial w_{ij}} > 0 , \\[2ex] 0.5\Delta_{ij}^{(t-1)}, & \text{if} \quad \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} \dfrac{\partial E^{(t)}}{\partial w_{ij}} < 0 , \\[2ex] \Delta_{ij}^{(t-1)}, & \text{else.} \end{cases} \tag{20}$$

A very special variation of the MLP networks is the *r*adial basis function (RBF) network. This network type has a basically fixed architecture shown in *Fig. 7*.

The network consists of three layers: the input layer, the only hidden layer and the output layer. In the case of *Fig. 7* the input vector has three components that is we have three input nodes, one summing up output node and n training patterns.

The activation function in the hidden layer is some type of radial basis function, usually the Gaussian one, expressed by the formula

$$h(\mathbf{x}) = e^{-\left(\frac{(\mathbf{x}-\mathbf{c})^T(\mathbf{x}-\mathbf{c})}{r^2}\right)}, \tag{21}$$

where the vector $\mathbf{c}$ represents the center, the scalar r the radius of the function. If the input patterns x are one-dimensional, then the center c is also a one-dimensional shift and the Gaussian takes the form:

$$h(x) = e^{-\left(\frac{x-c}{r}\right)^2}. \tag{22}$$

The output can work in two ways: without activation function, only summing up the input, or with a sygmoid type activation function with bias that is with an unknown shift b added to the input of the output neuron.

When the output operates without activation as in *Fig. 7*, the network can work in the following way. First we create as many neurons in the hidden layer as training patterns are present. The centers could be set equal to the corresponding input scalars (or vectors), the radii depend on the distances of the neighboring input patterns, for normalized input data one can select 1. In this case the number of unknown weights is equal to the training patterns. Because of the lack of nonlinear changes in the process of solution the weights can be computed directly solving a system of linear equations.

In the case of activation function with bias in the output we cannot use the linear approach, however, the solution even in this case is much more simple as in the case of general type MLP feedforward networks because the *delta rule* can be used here in its simple form (we deal only with weight changes linked to the output neuron(s)), expressed in formula (15). The number of hidden nodes in this second case should be less than the number of training patterns, consequently the determination of the centers is not straightforward any more, but the details of this case are out of our discussion.

The training itself is not the goal of the network but the tool to prepare it for the interpolation. This statement should be emphasized because of the fact that several textbooks and software products as well are inclined to forget the basic task of the network: to estimate the values of the particular phenomenon in unmeasured spots.

That is after the training is performed successfully we should apply as input the independent variables of the unknown points and get on the output the corresponding interpolated function values. This procedure in opposition to the training is direct and very fast.

At the end of this section a very important practical remark should be added. The activation functions dependending on D are sensitive to the input values only in range about $[-1.5, +1.5]$, therefore the input data should be scaled to this interval.

With exception of the linear activation function all others can give only outputs which have smaller absolute values than 1, therefore the target values should be also scaled corresponding to the output value range of the output activation function $[-1, +1]$ or $[0, +1]$.

2.4.3. Some Remarks on the Architecture of Artificial Neural Networks

The main question, how many layers with how many nodes should be included in a particular network, cannot be answered directly yet. There are strategies, algorithms, rules of thumb related to this question, however, for a theoretically consistent answer we have to wait for a while.

In fact the more layers and nodes are included in the network, the less more flexible it is in training. On the other hand, the larger the network, the more processing time is required for one training cycle and the more probable is its overtraining.

The compromise can be found by one of the following strategies: beginning the training with a small two hidden layer net with e.g. 5 nodes in each. If the training is too slow (the MSE does not decrease in the proper way), then we add gradually new nodes (until a reasonable limit e.g. 15) and new layers until the training improves essentially. The algorithm called cascade correlation performs automatically the network expansion.

The other strategy starts with an extended network and eliminates gradually the superfluous nodes. A large number of pruning algorithms can cope with this task automatically.

2.4.4. Artificial Neural Networks in GIS applications

Only a few applications are present in the GIS papers so far. Xingong Li (XINGONG, 1997) reports about the neural network used for precipitation estimation. His MLP feedforward network had three layers, the input layer with 9 nodes, the only hidden layer with 10 nodes and the output layer with one node.

The most interesting point of this case study is the choice of the nine input parameters. These are the precipitation, the heights of the three closest weather stations and the distances from those to the interpolating point.

I wonder whether the relative co-ordinates and the heights as input data could not give a more general solution.

The training set consisted of 50 weather station data for four months, the validation set of 18 similar station records. The interpolation was made on the validation set points in order to be able to check the results. The interpolation by neural network was compared to the truth value and to the results of other interpolating methods (Voronoy cells, TSA, inverse distance weighted and ordinary kriging) and it was found that it performs consistently well the interpolation, in comparison to other methods, in all months.

A more theoretic paper of D. Pariente (PARIENTE, 1994) deals with the use of neural networks connecting the object oriented spatial data model with the function field data model (SÁRKÖZY, 1994). This idea can be developed further by associating the data of a function field with a properly composed and trained neural network.

The neural networks have very extended applications in classification issues. This kind of application can be used in processing of multispectral remote sensing scenes, pattern extraction from images and also for classification of land quality in GIS as explained in (R. MUTTIAH at al., 1996). Though the interpolation and classification are related fields (the classification is nothing else as interpolation between predefined patterns) we have no possibility in this paper to dwell on the applications in this field in more details. We only want to call the attention of the readers to the fact that the authors of the referred paper developed a neural network interface for the GRASS GIS to solve classification tasks. I suppose that without essential changes the interface can be also used for interpolation.

3. Discussion

The main questions for discussion are as follows:

1. Where is the border between the interpolation procedures of data production and interpolation procedures as standard GIS functions.
2. For both cases which particular method has the most benefit.
3. What is common and what is different in classification and interpolation using artificial neural networks.
4. Aspects of data model in connection with artificial neural networks.

I hope that my answer to the majority of these questions is clear from the paper. However, to be sure in it I will give their summary in the conclusions. But let us see first some alternatives.

1. There are three possibilities. In the first case we try to include in the GIS as standard function at least those interpolation methods which were discussed in the paper. In the second case we include in the GIS only some local interpolation methods. The third case ignores the interpolation procedures as GIS functions.
2. Here we have several answers. There are people in favor of geostatistical methods for both data preparation and GIS. Others are adherent of geometrical methods based an Voronoy tesselation, first of all for GIS. The specialists with concern of computer resources often support the inverse distance weighting. Mathematicians prefer the RBF polynomials, etc. The motivation of every group may be accepted for particular circumstances, however,

to include a method into the GIS as a standard function the method and its implementation should fulfill a couple of conditions. Among others the method should be local, as precise as possible, the implementation should accept the GIS data model and possess easy-to-use control tools, it should run fast without exaggerated employment of computer resources.

3. The question can be approached from different directions. As far as ANN is concerned, the MLP feedforward networks are equally usable for interpolation and classification. Other types of ANN are exclusively or mainly suited to unsupervised (SOM) or supervised (LVQ) classification. The main difference between interpolation and classification MLPs lies in the context of input and output. The interpolation network uses point coverages for learning and estimation, the classification networks deal also with raster (pixel) data. As for functionality classification is nothing else as interpolation of the membership in given clusters. With other words we define a staircase like function where a group of input variables' intervals corresponds to a stair in the function. An even more interesting GIS application of neural networks is the environmental (erosion, pollution, etc.) modelling by neural networks. This is essentially an approximation of a complex multivariate function with reduced number of input variables. The input and output variables are usually raster data.
4. The data model was several times mentioned already in the previous paragraph. At this place we only want to remind the reader of the fact that a trained network is nothing else as the approximation of an arbitrary function valid for the region determined as the convex hull of training points.

4. Conclusions

- The fact that the production of digital spatial data has become an independent industry results in the separation of data capture and GIS. However, even in the case of excellent geospatial infrastructure one cannot avoid local interpolations in the GIS workflow. Thus, *some local interpolation procedures should get the status of standard GIS functions*.
- The Voronoy approach and its developments are substantially local spatial interpolation methods and should be included in every GIS software.
- For visualization the GIS, especially in 3D requires some type of spline interpolation, too.
- The GIS software of a high standard has to be completed with the procedure of implementing the method of local polynomials.
- The ANN simulators are tools of interpolation, classification, environmental modelling. It is obvious that the future GIS should have ANN modules. It has not been cleared so far which architectures, learning rules, etc. can combine all ANN functions with easy-to-use control tools. Answering these questions demands a lot of research.

- The ANN is related to the data model in two ways. First it has to reach the input and output data in form of the particular GIS data model, secondly it can be a data model itself substituting for example the DEM grid or TIN triangles for the elevation data. Although this approach seems to be very promising one should remember that it requires strict standardization of network architecture and very flexible choice of learning algorithms and parameters on the side of data production. At the same time the GIS can use the learned networks very easy running in recall mode forward propagation of the standardized network architecture.

References

[1] BRIGGS, I. C. (1974): Machine Contouring Using Minimum Curvature. *Geophysics.* Vol. 39, No. 1. February 1974. pp. 39–48.

[2] COLLINS, F. C. – BOLSTAD, P. V. (1996): A Comparison of Spatial Interpolation Techniques in Temperature Estimation. *Third International Conference/Workshop on Integrating GIS and Environmental Modeling*. Santa Fe, New Mexico, January 21-25, 1996. Proceedings CD-ROM.

[3] GOLD, CH. M. (1991): Problems with Handling Spatial Data – the Voronoy Approach. *CISM Journal ACSGC*. Vol. 45, No. 1. Spring 1991. pp. 65–80.

[4] ISAAKS, E. H. – SRIVASTAVA, R. M. (1989): An Introduction to Applied Geostatistics. Oxford University Press, New York, Oxford, 1989.

[5] LONG, A. E. – MYERS, D. E. (1997): A New Form of Cokriging Equations. *Mathematical Geology*. Vol. 29. No. 5, 1997. pp. 685–702.

[6] MITAŠOVA, H. – MITAŠ, L. (1993): Interpolation by Regularized Spline with Tension: I. Theory and Implementation. *Mathematical Geology*. Vol. 25. No. 6, 1993. pp. 641–655.

[7] MITAŠOVA, H. – BROWN, W. – GERDES, D. P. – KOSINOVSKY, I. – BAKER, T. (1993): Multidimensional Interpolation, Analysis and Visualization for Environmental Modeling. *GIS/LIS '93 Annual Conference November 2-4, 1993*. Minneapolis, Minnesota. Proceedings. Volume 2. pp. 550–556.

[8] PARIENTE, D. (1994): Geographic Interpolation and Extrapolation by Means of Neural Networks. *EGIS/MARI'94 Proceedings*. Vol. 1. pp. 684–693.

[9] SÁRKÖZY, F. (1994): The GIS Concept and the 3-Dimensional Modeling. *Computers, Environment and Urban Systems*. Vol. 18. No. 2, 1994. pp. 111–121.

[10] SÁRKÖZY, F. (1999): GIS Functions. *Periodica Polytechnica*

[11] SÁRKÖZY, F. – GÁSPÁR, P. (1992): Modelling of Scalar Fields Represented by Scattered 3D Points. *Periodica Polytechnica Civil Engineering*. Vol. 36, No. 2. 1992. pp. 187–200.

[12] SÁRKÖZY F. – ZÁVOTI, J. (1995): Conceptional Data Model for Modeling of Scalar Fields and One Compression Method Usable for its Implementation. *Proceedings of the Fourth International Symposium of LIESMARS titled: Toward Three Dimensional, Temporal and Dynamic Spatial Data Modeling and Analysis*. LIESMARS, WTUSM, P.R. China, Oct. 25–27, 1995. pp. 1–10.

[13] SMITH, D. R. – PARADIS, A. R. (1989): Three-Dimensional GIS for the Earth Sciences. *AUTO-CARTO 9. Proceedings*. Baltimore, 1989. pp. 324–335.

[14] XINGONG, LI (1997): Development of a Neural Network Spatial Interpolator for Precipitation Estimation. *GIS/LIS '97 Annual Conference*, October 28–30, 1997. Cincinnati, Ohio. Proceedings CD-ROM. pp. 667–676.

[15] YANG, W. – GOLD, CH. (1995): Dynamic Spatial Object Condensation Based on the Voronoy Diagram. *Proceedings of the Fourth International Symposium of LIESMARS titled: Toward Three Dimensional, Temporal and Dynamic Spatial Data Modeling and Analysis*. LIESMARS, WTUSM, P.R. China, Oct. 25–27, 1995. pp. 134–145.