ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2008, Vol.37, No.3

COLLAGE-BASED INTERPOLATION OF REAL-DATA SETS

Jonas Valantinas

Department of Applied Mathematics, Kaunas University of Technology
Studentų St 50-325c, LT-51368 Kaunas, Lithuania

Ramutis Rindzevičius

Department of Telecommunications, Kaunas University of Technology
Studentų St 50-423, LT-51368 Kaunas, Lithuania

Abstract. This paper presents a recurrent fractal interpolation method (approach) for one-dimensional sets of real-data. The method explores both the local collage idea, developed originally for image compression purposes, and the basic platform for generating of non-recurrent fractal interpolation functions – attractors of iterated function systems (IFS). The characteristic feature of the developed approach – the recurrent fractal interpolation functions are obtained by applying specialized correction procedures to the approximants of the real-data sets, i.e. to the attractors of local IFS, generated using self-similarities detected within the data under processing.

Keywords: recurrent iterated function systems, local collage, real-data sets, fractal interpolation.

1. Introduction

In engineering and science one often has a number of data points, as obtained by sampling or experiment, and tries to construct a function which closely fits those data points. This is called curve fitting or regression analysis. Interpolation is a specific case of curve fitting, in which the function must go exactly through the data points.

Methods for modelling (fitting, interpolating, approximating) piecewise smooth curves and surfaces are abundant. "Rough" curves (for instance, mountain range silhouettes, noisy radio-signals, etc.) that are differentiable almost nowhere are generally more difficult to model accurately with the economy of parameters (say, using polynomial or spline interpolation approach). Fractal interpolation and approximation functions are attractive for their compact representation of real-data - "rough" shapes (one-dimensional images or signals).

Fractal interpolation functions were founded over a decade ago and have been used mainly to represent uncertainty in the visualisation of scientific data and to imply the shape of a dropping leaf [1, 2]. Barnsley and Hutchinson were pioneers in the use of fractal functions to interpolate sets of data [3-5]. Soon it became clear that fractal interpolants can be defined for any continuous function defined on a real compact interval [6, 7]. This method constitutes an advance in the techniques of approximation, since all the classical methods of real-data interpolation can be generalized using fractal techniques [8, 9].

Fractal interpolation functions (FIF) are defined as fixed points of transformations between spaces of functions using iterated function systems (IFS). The theorem of Barnsley and Harrington proves the existence of differentiable fractal interpolation functions [10].

In the paper, a generalization of the FIF, called the recurrent FIF, is proposed. Like the FIF, the recurrent FIF represents a function (signal, one-dimensional image) with a set of affine shear transformations (local IFS). However, whereas the FIF models the graph of the function out of smaller copies of itself, the recurrent FIF models the graph out of smaller copies of sections of the function. Interestingly, the local IFS based approach originally was developed and applied to fractal encoding of two-dimensional digital images [11, 12].

2. Fractal interpolation functions

Suppose that a particular experiment consists in measuring values of a certain real-valued function F. Measurement results – the set of real-data $\{(x_i, F_i) \in \mathrm{R}^2 \mid i = 0,1,\ldots,N\}$, where $F_i = F(x_i)$, for all $i = 0,1,\ldots,N$; $x_0 < x_1 < \ldots < x_N$.

Classically, F is assumed to be smooth, and the input points are interpolated with a single degree N

polynomial, or a piecewise smooth low-degree polynomial (splines). Recent research has proved an alternative assumption that the function F is self-similar, and typically not smooth but fractal.

We construct an iterated function system (IFS) whose attractor is the graph of a function $F:[x_0,x_N]\to \mathrm{R}$. Such a function is called a fractal interpolation function (FIF) [3].

Let $\omega_i:[x_0,x_N]\times \mathrm{R}\to[x_0,x_N]\times \mathrm{R}$, $i=1,2,\ldots,N$, be affine (shear) transformations of the form

$$\omega_i(x,y)=\begin{pmatrix} a_i & 0\\ c_i & d_i\end{pmatrix}\cdot\begin{pmatrix} x\\ y\end{pmatrix}+\begin{pmatrix} e_i\\ f_i\end{pmatrix}, \qquad (1)$$

where $|d_i|<1$ is given as a parameter controlling the roughness of the function, and a_i, c_i, e_i and f_i are determined either by the constraints

$$\omega_i(x_0,F_0)=(x_{i-1},F_{i-1}),\ \omega_i(x_N,F_N)=(x_i,F_i)$$

or by the "reflected" constraints

$$\omega_i(x_N,F_N)=(x_{i-1},F_{i-1}),\ \omega_i(x_0,F_0)=(x_i,F_i),$$

for all $i=1,2,\ldots,N$. For instance, in the case of former constraints, coefficients a_i, c_i, e_i and f_i are defined as follows:

$$a_i=\frac{x_i-x_{i-1}}{x_N-x_0},\quad c_i=\frac{F_i-F_{i-1}}{x_N-x_0}-d_i\frac{F_N-F_0}{x_N-x_0},$$

$$e_i=\frac{x_Nx_{i-1}-x_0x_i}{x_N-x_0},$$

$$f_i=\frac{x_NF_{i-1}-x_0F_i}{x_N-x_0}-d_i\frac{x_NF_0-x_0F_N}{x_N-x_0}.$$

Given the metric

$$d((x_{i-1},F_{i-1}),(x_i,F_i))=|x_i-x_{i-1}|+\theta\cdot|F_i-F_{i-1}|, \qquad (2)$$

where $\theta=1$, for $c_1=c_2=\ldots=c_N=0$, and $\theta=\min\{1-|a_i|\}/\max\{2\cdot|c_i|\}$, otherwise, it is shown that each ω_i has contractivity factor

$$s_i=\max\{|a_i|+\min\{1-|a_i|\}/2,\max\{|d_i|\}\}<1.$$

Hence, by the fixed point theorem, there exists one and only one function F satisfying the relationship

$$F=\bigcup_{i=1}^{N}\omega_i(F),$$

i.e. F can be represented as the union (collage) of smaller copies of itself [3].

In the section below we present a newly developed procedure (method) for constructing a generalization of the FIF, called the recurrent (local) FIF. The procedure comprises three steps – construction of a local IFS for a given set of input points (real-data), generation of an intermediate recurrent real-data approximation function and formation of a recurrent fractal interpolation function. It is worth emphasizing that the developed approach provides much more flexibility in representing "rough" curves.

3. Generating of recurrent fractal interpolation functions

Like the FIF, the recurrent FIF models the graph F with a set of affine (shear) transformations, i.e. with an IFS $\{\mathrm{R}^2;\omega_1,\omega_2,\ldots,\omega_N\}$. However, whereas the FIF models the graph F out of smaller copies of F, the recurrent FIF models the graph of F out of smaller copies of sections of F.

For each $i\in\{1,2,\ldots,N\}$, we have indices j_i and k_i such that

$$\omega_i(x_{j_i},F_{j_i})=(x_{i-1},F_{i-1}),\ \omega_i(x_{k_i},F_{k_i})=(x_i,F_i),$$

and $|x_{k_i}-x_{j_i}|>|x_i-x_{i-1}|$. If all of the shears are contractive under (2), then there is one and only one graph F that satisfies the relationship (local collage condition)

$$F=\bigcup_{i=1}^{N}\omega_i(F[x_{j_i},x_{k_i}]).$$

Thus, given a sequence of input points (x_i,F_i), we can construct a recurrent IFS that interpolates these points by forcing the curve that lies between (x_i,F_i) and (x_{i+1},F_{i+1}) to be a contracted copy, say, of the curve that begins at (x_{i-1},F_{i-1}) and ends at (x_{i+2},F_{i+2}).

In the sections below, we shortly present a developed approach for generating of recurrent fractal interpolation functions for real-data sets, with the use of local IFS.

3.1. Constructing local iterated function systems

Let $S=\{(x_i,F_i)\in\mathrm{R}^2\mid i=0,1,\ldots,N\}$ be the set of real-data, i.e. the set of input points. Without loss of generality, we assume that $x_{i+1}-x_i=h=2^n$, for all $i=0,1,\ldots,N-1$; here $n\in\{1,2,\ldots\}$.

To construct a local collage-based IFS, associated with the given real-data set S, the latter set is partitioned at two scales (one twice the other), i.e. into the fixed size range subsets (blocks)

$$R_i=\{F_{\lambda i},F_{\lambda i+1},\ldots,F_{\lambda(i+1)}\},$$

supported by the segments $[x_{\lambda i},x_{\lambda(i+1)}]$, $i=0,1,\ldots,[\![N/\lambda]\!]-1$, and the domain subsets (blocks)

$$D_j=\{F_{\lambda j},F_{\lambda j+1},\ldots,F_{\lambda(j+2)}\},$$

supported by the segments $[x_{\lambda j},x_{\lambda(j+2)}]$, $j=0,1,\ldots,[\![N/\lambda]\!]-2$; here $\lambda=2^p$, $p\in\{1,2,\ldots\}$, and $[\![x]\!]$ stands for the integral part of the real number x. As it can be seen, the former (range) blocks are non-overlapping and contain every input point. The latter ones (domain blocks) may overlap and not necessarily contain every input point.

The essence of the approach is the pairing of each range block R to a domain block D such that the mean squared error $\delta = \delta(R, D)$ is minimal.

The search for the best domain block D_j ($j \in \{0,1,\ldots,[\![N/\lambda]\!]-2\}$) for a particular range block R_i ($i = 0,1,\ldots,[\![N/\lambda]\!]-1$) is complicated by the requirement that the range block matches the transformed version $\hat{D}_j$ of the domain block D_j, i.e. the block $\hat{D}_j = \{\hat{F}_{\lambda j}, \hat{F}_{\lambda j+1}, \ldots, \hat{F}_{\lambda(j+1)}\}$, where:

$$\hat{F}_{\lambda j} = F_{\lambda j},\ \hat{F}_{\lambda(j+1)} = F_{\lambda(j+2)},$$

$$\hat{F}_{\lambda j+r} = \frac{1}{3}\sum_{s=-1}^{1} F_{\lambda j+2r+s},\ r = 1,2,\ldots,\lambda-1.$$

The mean squared error $\delta = \delta(R_i, D_j)$ for any pair "range block R_i - domain block D_j" is found using an expression

$$\delta(R_i, D_j) = \left(\frac{1}{\lambda-1} \sum_{r=1}^{\lambda-1} \left(\Delta F_{\lambda i+r} - k_i \cdot \Delta \hat{F}_{\lambda j+r} \right)^2 \right)^{1/2};$$

here: $\Delta F_{\lambda i+r} = F_{\lambda i+r} - y_i(x_{\lambda i+r})$ and $\Delta \hat{F}_{\lambda j+r} = F_{\lambda j+r} - \hat{y}_j(x_{\lambda j+r})$, for all $r = 1,2,\ldots,\lambda-1$; $y = y_i(x)$ is the equation of a straight line connecting the points $(x_{\lambda i}, F_{\lambda i})$ and $(x_{\lambda(i+1)}, F_{\lambda(i+1)})$, whereas $y = \hat{y}_j(x)$ is the equation of a straight line connecting another two points, $(x_{\lambda j}, \hat{F}_{\lambda j})$ and $(x_{\lambda(j+1)}, \hat{F}_{\lambda(j+1)})$; k_i ($i = 0,1,\ldots,[\![N/\lambda]\!]-1$) is chosen to draw the range and the domain blocks closer, in other words, to minimize the value of δ, i.e.

$$k_i = \frac{1}{\alpha_j} \sum_{r=-1}^{\lambda-1} \Delta F_{\lambda i+r} \cdot \Delta \hat{F}_{\lambda j+r},$$

for $\alpha_j = \sum_{r=1}^{\lambda-1} (\Delta \hat{F}_{\lambda j+r})^2 \neq 0$, and $k_i = 0$, otherwise (Figure 1).

As a result of above calculations, a local IFS $\{R^2; \omega_1, \omega_2, \ldots, \omega_{[\![N/\lambda]\!]-1}\}$ is constructed. Each transformation ω_i ($i = 0,1,\ldots,[\![N/\lambda]\!]-1$) is determined by a triple of numbers $\langle i, j, \hat{k}_i \rangle$, where the coefficient $\hat{k}_i$ is assumed to be: $\hat{k}_i = -1$, if $k_i < -1$; $\hat{k}_i = k_i$, if $|k_i| \leq 1$; $\hat{k}_i = 1$, if $k_i > 1$. Let us note here that $\hat{k}_i = d_i$, for all $i = 0,1,\ldots,[\![N/\lambda]\!]-1$ (expression (1); Section 2). To say more, $\hat{k}_i$ is introduced to ensure convergence in generating attractor of the constructed local IFS.

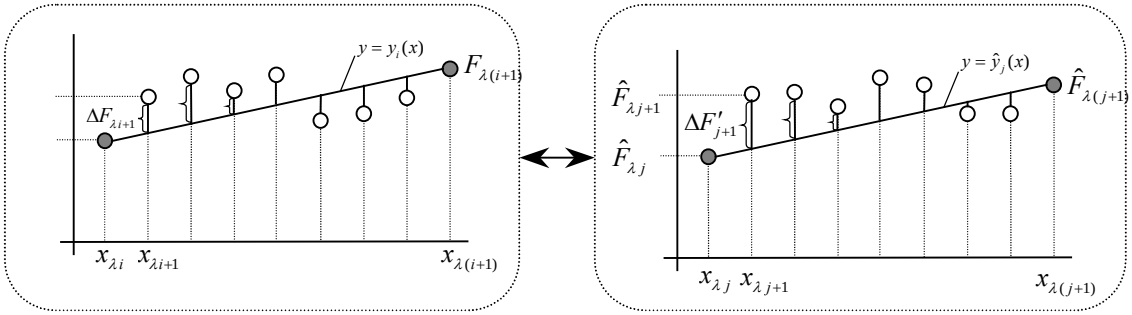

Figure 1. Organizing local collage for the set of real-data ($\lambda = 2^3 = 8$)

3.2. Generating recurrent approximation functions

The set of ordered triples $\langle i, j, \hat{k}_i \rangle$, $i = 0,1,\ldots,[\![N/\lambda]\!]-1$, obtained in the previous section, is processed in a consecutive order.

Since $\lambda = 2^p$, it becomes clear that precisely p iterations are needed to reconstruct the initial set of real-data, i.e. to get an approximant of the set $\{(x_i, F_i) \in \mathrm{R}^2 \mid i = 0,1,\ldots,[\![N/\lambda]\!]\}$. Since, for each i, $h = x_{i+1} - x_i = 2^n$, additional n iterations are necessary to fill in "gaps" between the input points (x_i, F_i), $i = 0,1,\ldots,[\![N/\lambda]\!]$.

So, if we denote the whole set of points on the number line (monitor's screen) by $\{(x'_i, F'_i) \in \mathrm{R}^2 \mid i = 0,1,\ldots,[\![N/\lambda]\!] \cdot h\}$, then the first iteration of the generation process implies (Figure 2):

$$(x'_{\lambda hi}, F'_{\lambda hi}) := \omega_i(x_{\lambda j}, F_{\lambda j}) = (x_{\lambda i}, F_{\lambda i}),$$

$$(x'_{\lambda h(i+1/2)}, F'_{\lambda h(i+1/2)}) := \omega_i(x_{\lambda(j+1)}, F_{\lambda(j+1)}),$$

$$(x'_{\lambda h(i+1)}, F'_{\lambda h(i+1)}) := \omega_i(x_{\lambda(j+2)}, F_{\lambda(j+2)}) = (x_{\lambda(i+1)}, F_{\lambda(i+1)}),$$

for all $i = 0,1,\ldots,[\![N/\lambda]\!]-1$.

The s-th iteration ($s = 2,3,\ldots,p+n$) yields the following results (for all $i = 0,1,\ldots,[\![N/\lambda]\!]-1$):

$$(x'_{\lambda h(i+(2t-1)/2^s)}, F'_{\lambda h(i+(2t-1)/2^s)}) := \omega_i(x'_{\lambda h(j+(2t-1)/2^{s-1})}, F'_{\lambda h(j+(2t-1)/2^{s-1})});$$

here $t = 1, 2, \ldots, 2^{s-1}$.

Application of the local collage to the above generation process leads to the conclusion that the resulting function (curve), in general, does not pass through the initially given input points $(x_i, F_i) \in \mathrm{R}^2$, whose indices $i \neq \lambda \cdot t$, $t = 0,1,\ldots,[\![N/\lambda]\!]$. This fact distorts the essence of the interpolation problem. In the next section we propose a correction procedure providing a way out of a difficulty.

3.3. Passing from recurrent approximation functions to fractal interpolation functions.

The final fractal interpolation function (curve) associated with the given set of real-data

$S = \{(x_i, F_i) \in \mathrm{R}^2 \mid i = 0,1,\ldots,[\![N/\lambda]\!] \cdot \lambda\}$ is obtained by evaluating anew ordinates of the points in the blocks supported by the segments $[x_{hi}, x_{h(i+1)}]$, $i = 0,1,\ldots,[\![N/\lambda]\!] \cdot \lambda - 1$.

The given below procedure serves the purpose – for each value of i ($i = 0,1,\ldots,[\![N/\lambda]\!] \cdot \lambda - 1$) the necessary corrections are made:

$$F''_{hi+r} := F'_{hi+r} + \frac{a(h-r) + br}{h},$$

for all values of $r = 0,1,\ldots,h$; here: $a = F_{hi} - F'_{hi}$, $b = F_{h(i+1)} - F'_{h(i+1)}$.

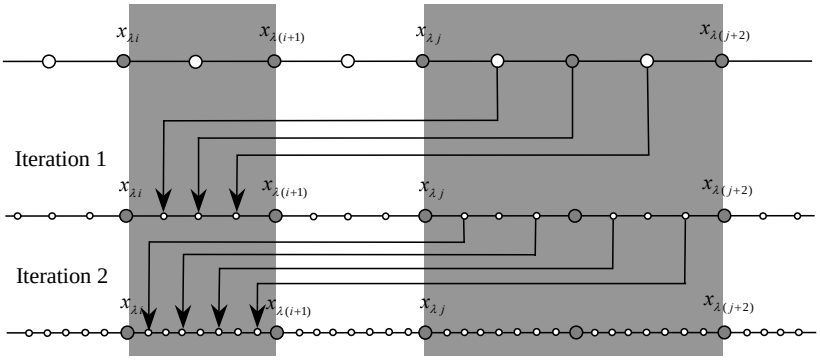

Figure 2. Generating recurrent approximation functions (curves)

The sought-for fractal interpolation function (curve), associated with the initial set of real-data S (Section 3.1), now is represented by the newly generated set of points

$$S' = \{(x'_i, F''_i) \in \mathrm{R}^2 \mid i = 0,1,\ldots,[\![N/\lambda]\!] \cdot h\}.$$

4. Experimental results

To examine application of the proposed theoretical developments to practice, an exploratory test has been carried out. Computer realization (Programming language – .NET C#) was done by Edvinas Medišauskas – Master of Science in Applied Mathematics.

The objective set of real-data was formed using a seismogram of the earthquake, as recorded on the British Geological Survey (BGS) seismograph network (Figure 3). Herewith, the earthquake measuring 3.5 on Richter scale has shaken parts of Dumfries and Galloway (Scotland; December 26, 2006; 10:40 GMT).

Worth emphasizing that the "shape" of an earthquake, it was observed some time ago, is self-reflective. Warning tremors and aftershocks are like miniquakes. This sort of behaviour is difficult to model with traditional curves but seems well suited for fractals.

To construct the working set of real-data (input points) S, every fourth pixel value of the seismogram (Figure 3) has been retrieved, i.e. $S = \{(x_i, F_i) \in \mathrm{R}^2 \mid i = 0,1,\ldots,316\}$. So, the step size $h = x_{i+1} - x_i = 2^2 = 4$, $i = 0,1,\ldots,315$, and two iterations were needed to fill in gaps between the interpolation points.

Initially, for organizing of a local collage, the parameter λ was equated to $2^4 = 16$ (Figure 4). After six iterations and necessary corrections a final recurrent fractal interpolation function has been generated (Figure 5).

One can easily see that the generated fractal interpolation curve follows the overall shape of the real earthquake (Figure 6; the real-data points are coloured in black).

To make comparative analysis results more precise, the mean squared error δ and the proportionate mean squared error $\Delta\delta$ were found, namely: $\delta = 10.61$ and $\Delta\delta = \delta / B \cdot 100\,\% = 3.4\,\%$ (here B indicates the range of the seismogram). Tolerable modelling results were obtained also for $\lambda = 8$, i.e. $\delta = 10.61$ and $\Delta\delta = 4.1\,\%$.

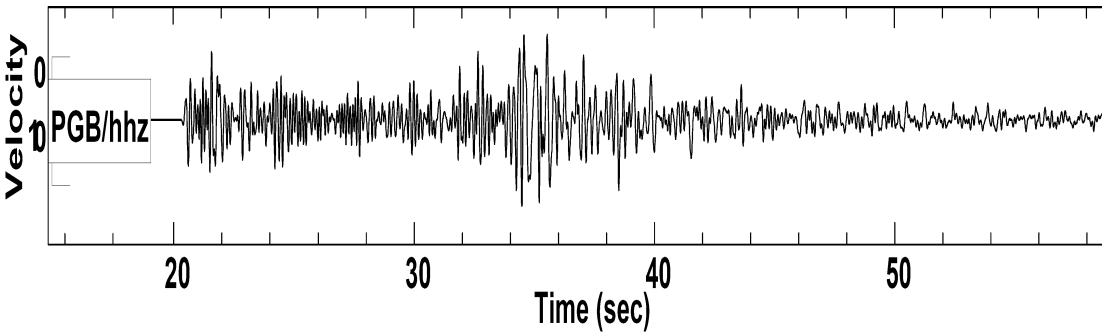

Figure 3. The seismogram of the earthquake (bitmap image of size 3792×402), as recorded on the BGS seismograph network

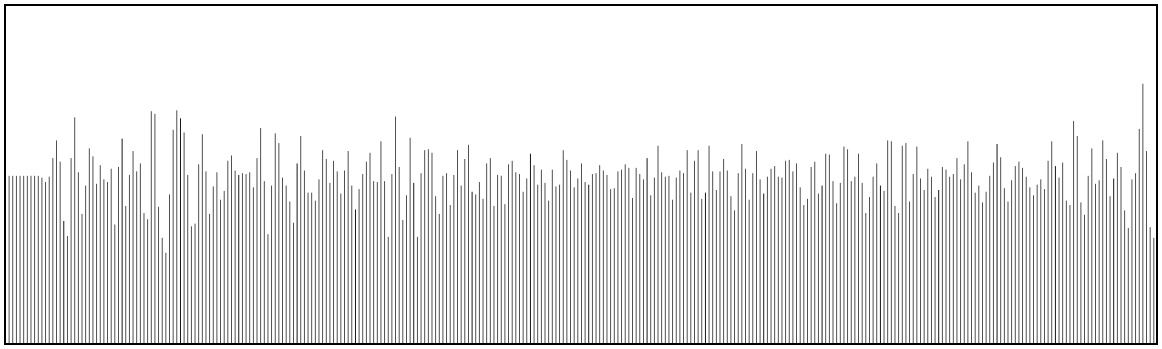

Figure 4. The digitized third of the seismogram ($N = 316$, $h = 2^2 = 4$) and sampling points for organizing of a local collage ($\lambda = 2^4 = 16$)

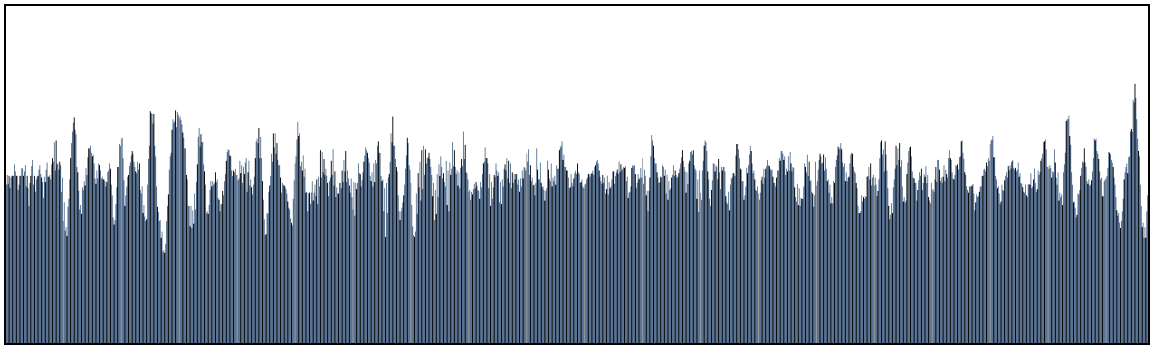

Figure 5. The fractal interpolation function, obtained after performing six iterations

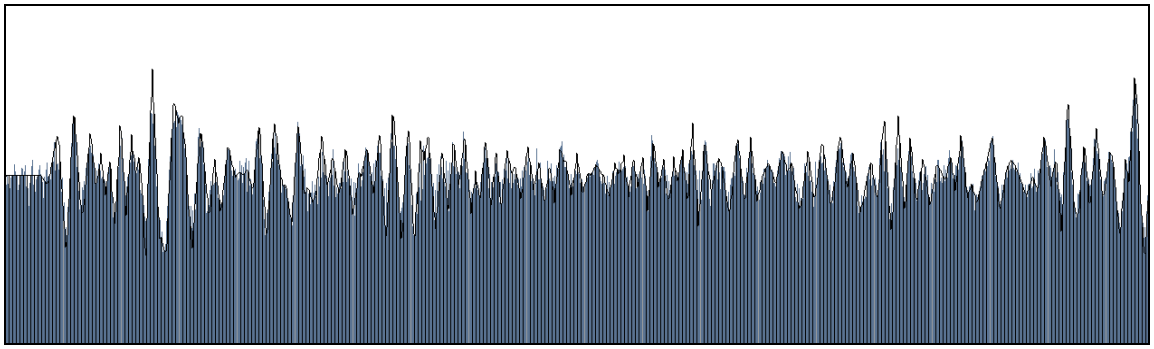

Figure 6. Visualization of comparative analysis results (the seismogram is coloured in black)

5. Conclusion

In the paper, a challenging approach (method) to interpolation (modelling) of "rough" curves (signals, one-dimensional images) by fractal techniques is proposed. The method is built on the analysis of self-similarities found within the sets of real-data under processing, explores recurrent iterated function systems, local collage relationships and a newly develo-

ped procedure for passage from fractal approximation models of "rough" shapes to fractal interpolation ones.

The obvious advantage of the developed approach – calculation complexity marginally depends on the amounts of real-data sets. Preliminary experimental analysis results are promising - shape modelling errors are tolerable.

Despite the fact that the recurrent FIF models are general enough and can be used to describe a wide variety of shapes, they also require that the "domain blocks" of each affine transformation were the union of "range blocks" (Section 3.1). In the future, we are to concentrate our attention on a special class of recurrent IFS which allow the "domain block" to be arbitrary.

Finally, we observe that the developed fractal "rough" curve modelling approach can be successively and profitably employed under circumstances where the measurement (experimental, observation) results are obtained at a great cost or, say, the duration of measurements themselves is time consuming.

References

[1] **C.M. Wittenbrink.** IFS fractal interpolation for 2D and 3D visualization. *Proceedings of the* 6th *IEEE Visualization Conference* (*Visualization* '95), *IEEE Computer Society*, 1995, 77-84.

[2] **N. Zhao.** Construction and application of fractal interpolation surfaces. *The Visual Computer*, *Vol*.12, *No*.3, 1996, 132-146.

[3] **M.F. Barnsley.** Fractal functions and interpolation. *Constructive Approximation*, *Vol*.2, *No*.1, 1986, 303-329.

[4] **M.F. Barnsley.** Fractals Everywhere. *Academic Press Professional, Cambridge*, 1993.

[5] **J.E. Hutchinson.** Fractals and self-similarity. *Indiana University Mathematics Journal*, *Vol*.30, *No*.5, 1981, 713-747.

[6] **M.A. Navascués, M.V. Sebastián.** Smooth fractal interpolation. *Journal of Inequalities and Applications*, *Vol*.2006, *Article ID* 78734, 2006, 1-20.

[7] **LI Xin-Fu, LI Xiao-Fan.** An Explicit Fractal Interpolation Algorithm for Reconstruction of Seismic Data. *Chinese Physics Letters*, *Vol*.25, *No*.3, 2008, 1157-1159.

[8] **M.A. Navascués, M.V. Sebastián.** Some results of convergence of cubic spline fractal interpolation functions. *Fractals*, *Vol*.11, *No*.1, 2003, 1-7.

[9] **M.A. Navascués, M.V. Sebastián.** Generalization of Hermite functions by fractal interpolation. *Journal of Approximation Theory*, *Vol*.131, *No*.1, 2004, 19-29.

[10] **M.F. Barnsley, A.N. Harrington.** The calculus of fractal interpolation functions. *Journal of Approximation Theory*, *Vol*.57, *No*.1, 1989, 14-34.

[11] **M.F. Barnsley, L.P. Hurd.** Fractal Image Compression. *AK Peters, Wellesley*, 1993.

[12] **J. Valantinas, T. Žumbakis.** The Use of Image Smoothness Estimates in Speeding Up Fractal Image Compression. *Lecture Notes in Computer Science* 3540, *Springer Berlin / Heidelberg*, 2005, 1167-1176.

Received April 2008.

DOI: 10.5755/j01.itc.37.3.11978