Modelling Discrete Sequences with Fractal Interpolation Functions of higher order

P. Bouboulis*
Department of Informatics and Telecommunications,
Telecommunications and Signal Processing, University of Athens,
Panepistimiopolis 157 84, Athens, Greece
bouboulis@di.uoa.gr

February 6, 2010

Abstract

We present a method that increases the accuracy of approximation of discrete sequences from fractal interpolation functions. Instead of previously presented models which use affine mappings, we employ polynomials of order d. Comparative results confirm the improvements in the accuracy of the proposed method.

1 Introduction

Fractal interpolation functions have drawn a great deal of attention in many scientific areas. They pose as an alternative to traditional interpolation techniques, giving a broader set of interpolants. In fact, many traditional interpolation techniques (splines, hermite polynomials et.c.) are included as special cases. Its main differences from standard methods consist: a) in the presence of a self similarity in small scales, b) in the constructive way (through iterations), that it is used to compute the interpolant, instead of the descriptive one (usually a formula) provided by the classical methods and c) in the usage of some parameters, which are usually called vertical scaling factors, that are strongly related with the fractal dimension of the interpolant. They have been used to construct wavelets ([13], [14], [12]), to model one-dimensional signals ([18], [19], [21]), in medical applications ([10], [20]), in remote sensing ([15]), in computer graphics ([17], [16], [22]), in image compression ([7], [8], [9]) and several other applications.

This paper deals with the modelling of discrete sequences using fractal interpolation, a subject pioneered by Mazel and Hayes in [18]. They used piecewise self-affine fractal interpolation functions and devised an algorithm to compute the parameters of the model. Most applications dealing with approximation of rough objects using fractal interpolation still use variants of their algorithm. However, their approach has several drawbacks. The use of affine mappings for the construction of the fractal function that approximate the given sequence makes the model rather inflexible. Still, there hasn't been proposed a method that overcomes this obstacle. This is the objective of the present paper. We employ polynomials of order d and gain significant improvements.

The structure of the paper is as follows. In section 2 the main aspects of IFS and RIFS theory are outlined and a little more detailed description of the concept of fractal interpolation is given. The methodology of Mazel and Hayes in a broader context suitable for our purposes is also found there. Section 3 deals with fractal interpolation functions of higher order. The procedure of the computation of the new model parameters using convex optimization and some issues regarding the implementation of the method are given. Finally we conclude with a comparison between the new method and the Mazel-Hayes approach.

*Partially supported by the Special Account for Research Grants of the University of Athens #70/4/5626.

2 Background

2.1 IFS-RIFS

An *Iterated Function System* (or IFS for short) $\{X; w_{1-N}\}$ is defined as a pair consisting of a complete metric space (X, ρ), together with a finite set of continuous, contractive mappings $w_n : X \to X$, with respective contraction factors $|s_n| < 1$, for $n = 1, 2, \ldots, N$ ($N \geq 2$). The attractor of an IFS is the unique set E, for which $E = \lim_{k\to\infty} W^k(A_0)$ for every starting compact set A_0, where

$$W(A) = \bigcup_{n=1}^{N} w_n(A) \text{ for all } A \in \mathcal{H}(X),$$

and $\mathcal{H}(X)$ is the complete metric space of all nonempty compact subsets of X with respect to the Hausdorff metric h (for the definition of the Hausdorff metric, properties of $\langle \mathcal{H}(X), h\rangle$ and examples of IFS, see [3] and [2] among others). Since their introduction by Barnsley and Demko in the 80's (see [3]) they have been used in numerous applications, due to their ability to construct highly complex structures with only a handful of mappings. A notion closely related with IFS is that of the *Recurrent Iterated Function System*, or *RIFS* for short, that allows the construction of even more complicated sets, by utilizing an irreducible row-stochastic matrix P: $(p_{n,m} \in [0, 1] : n, m = 1, \ldots, N)$, such that

$$\sum_{m=1}^{N} p_{n,m} = 1, \quad n = 1, \ldots, N. \tag{1}$$

The concept of the RIFS resembles that of a discrete time Markov process. The matrix P contains the transition probabilities for the process (i.e. $p_{n,m}$ gives the probability of transfer into state m given that the process is in state n). Condition (1) says that whichever state the system is in (say n), a set of probabilities is available that sum to one and describe the possible states to which the system transits at the next step. We will return to these issues later, when we will use RIFS for interpolation (see algorithm 2). For the construction of the attractor of the RIFS a little more effort is needed. The interested reader is addressed to [2], [4], [5] and [6] for more details.

2.2 Fractal Interpolation

Barnsley in [1] was the first to introduce functions that are derived as attractors of IFSs or RIFSs (see [1], [4]) and interpolate given data points. Here we briefly describe this construction based on RIFSs. The resulting functions are called *Recurrent Fractal Interpolation Functions* (or RFIF for sort).

Let $X = [0, 1] \times \mathbb{R}$ and $\Delta = \{(x_n, y_n) : n = 0, 1, \ldots, N\}$ be an interpolation set with $N + 1$ interpolation points such that $0 = x_0 < x_1 < \cdots < x_N = 1$. The interpolation points divide $[0, 1]$ into N intervals $I_n = [x_{n-1}, x_n]$, $n = 1, \ldots, N$, which we call *domains*. In addition, let $Q = \{(\hat{x}_j, \hat{y}_j) : m = 0, 1, \ldots, M\}$ be a subset of Δ, such that $0 = \hat{x}_0 < \hat{x}_1 < \cdots < \hat{x}_M = 1$. We also assume that for every $m = 0, 1, \ldots, M - 1$ there is at least one n such that $\hat{x}_m < x_n < \hat{x}_{m+1}$. Thus, the points of Q divide $[0, 1]$ into M intervals $J_m = [\hat{x}_{m-1}, \hat{x}_m]$, $m = 1, \ldots, M$, which we call *regions*. Finally, let $\mathbb{J}$ be a labelling map such that $\mathbb{J}$: $\{1, 2, \ldots, N\} \to \{1, 2, \ldots, M\}$ with $\mathbb{J}(n) = m$. Let $x_n - x_{n-1} = \delta_n$, $n = 1, 2, \ldots, N$, and $\hat{x}_m - \hat{x}_{m-1} = \psi_m$, $m = 1, 2, \ldots, M$. It is evident that each region contains an integer number of domains. In the special case where the interpolation points are equidistant (that is $x_n - x_{n-1} = \delta$, $n = 1, 2, \ldots, N$, and $\hat{x}_m - \hat{x}_{m-1} = \psi$, $m = 1, 2, \ldots, M$), each region contains exactly $\psi/\delta \in \mathbb{N}$ domains.

We define N mappings of the form:

$$w_n \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} L_n(x) \\ F_n(x, y) \end{pmatrix}, \quad \text{for } n = 1, 2, \ldots, N, \tag{2}$$

where $L_n(x) = a_n x + b_n$ and $F_n(x, y) = s_n y + q_n(x)$ ($q_n(x)$ is a continuous function). Each map w_n is constrained to map the endpoints of the region $J_{\mathbb{J}(n)} \equiv J_m$ to the endpoints of the domain I_n (see figure 1). That is,

$$w_n \begin{pmatrix} \hat{x}_{m-1} \\ \hat{y}_{m-1} \end{pmatrix} = \begin{pmatrix} x_{n-1} \\ y_{n-1} \end{pmatrix}, w_n \begin{pmatrix} \hat{x}_m \\ \hat{y}_m \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \end{pmatrix}, \quad \text{for } n = 1, 2, \ldots, N. \tag{3}$$

Solving for a_n, b_n we obtain $a_n = (x_n - x_{n-1})/(\hat{x}_m - \hat{x}_{m-1})$ and $b_n = (\hat{x}_m x_{n-1} - \hat{x}_{m-1} x_n)/(\hat{x}_m - \hat{x}_{m-1})$. Vertical segments are mapped to vertical segments scaled by the factor s_n. The parameter s_n is called the *vertical scaling factor* of the map w_n. It is easy to show that if $|s_n| < 1$, then there is a metric d equivalent to the Euclidean metric, such that w_n is a contraction (i.e., there is $\hat{s}_n : 0 \leq \hat{s}_n < 1$ such that $d(w_n(\vec{x}), w_n(\vec{y})) \leq \hat{s}_n d(\vec{x}, \vec{y})$, see [2]).

To conclude, we must define the $N \times N$ stochastic matrix P:

$$p_{nm} = \begin{cases} \frac{1}{\gamma_n}, & \text{if } I_n \subseteq J_{\mathbb{J}(m)} \\ 0, & \text{otherwise.} \end{cases},$$

where γ_n is the number of positive entries of the line n, $n = 1, 2, \ldots, N$. This means that $p_{n,m}$ is positive, if and only if there is a transformation L_m, which maps the region containing the nth domain (i.e. I_n) to the mth domain (i.e. I_m). Usually, it is more efficient to describe the stochastic matrix through the connection vector $V = (\mathbb{J}(1), \mathbb{J}(2), \ldots, \mathbb{J}(N))$, that describes which region is mapped to each domain. Assuming that we have chosen a set of interpolation points Δ, its subset Q, the vertical scaling factors s_n, V, and some arbitrary mappings that satisfy (3) we can construct the RFIF using the Deterministic Iterative Algorithm (Algorithm 1) or the Random Iterative Algorithm (Algorithm 2).

Input: Δ, Q, $\{s_1, \ldots, s_N\}$, V, $\{q_1(x), \ldots, q_N(x)\}$, ν (the number of steps).
Output: f_ν
$f_0 = \Delta$;
for $k = 1$ *to* ν **do**
  $f_k = \emptyset$;
  foreach *domain* I_n **do**
    Find the corresponding region $J_{V(n)}$;
    Denote $R_{V(n)}$ as the set containing the points of f_{k-1} that lie inside $J_{V(n)}$;
    Map the points of $R_{V(n)}$ using the map w_n, i.e. $D_n = w_n\left(R_{V(n)}\right)$;
    $f_k = f_k \cup D_n$;
  end
end

Algorithm 1: Deterministic Iterative Algorithm (DIA)

Input: Δ, Q, $\{s_1, \ldots, s_N\}$, P, $\{q_1(x), \ldots, q_N(x)\}$, ν (the number of iterations).
Output: f
Let $(x^{(0)}, y^{(0)}) \in \Delta$ be one of the interpolation points chosen at random;
If $(x^{(0)}, y^{(0)}) \in I_n$, choose $n_0 = n$ as the initial state of the Markov process;
$f = \{(x^{(0)}, y^{(0)})\}$;
for $k = 1$ *to* ν **do**
  Using the set of probabilities $\{p_{n_{k-1},1}, \ldots, p_{n_{k-1},N}\}$, choose one of the mappings $w_1, \ldots, w_N$ (i.e. we choose w_n with probability $p_{n_{k-1},n}$) and denote it as w_{n_k}. The process is now at state n_k;
  Let $(x^{(k)}, y^{(k)}) = w_{n_k}\left((x^{(k-1)}, y^{(k-1)})^\top\right)$;
end
$f = \{(x^{(k)}, y^{(k)}),\ k = 0, \ldots, \nu\}$;

Algorithm 2: Random Iterative Algorithm (RIA)

As mentioned above, while the theory allows the use of arbitrary functions $\{q_1(x), \ldots, q_N(x)\}$ that satisfy (3), only the linear case has been considered in most applications, i.e. $q_n(x) = c_n x + d_n$, $n = 1, \ldots, N$. In this case (3) gives a linear system which can be easily solved to obtain:

$$c_n = \frac{y_n - y_{n-1}}{\hat{x}_m - \hat{x}_{m-1}} - s_n \frac{\hat{y}_m - \hat{y}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}},$$
$$d_n = \frac{\hat{x}_m y_{n-1} - \hat{x}_{m-1} y_n}{\hat{x}_m - \hat{x}_{m-1}} - s_n \frac{\hat{x}_m \hat{y}_{m-1} - \hat{x}_{m-1} \hat{y}_m}{\hat{x}_m - \hat{x}_{m-1}},$$

for all $n = 1, \ldots, N$. Thus, once the contraction factor s_n for each map has been chosen, the remaining parameters may be easily computed. This is the reason why the linear case is so popular. Mazel and Hayes (see [18]) used this model (which they called piecewise self-affine fractal model) to approximate discrete sequences. Their methodology is briefly described in algorithm 3 in a more general setting. The computation of the vertical scaling factor is done via unconstrained optimization of the mean square error or using some geometric criteria ([18]). Both methods, however, may well lead to non acceptable values for the vertical scaling factor ($|s_n| > 1$). This is another drawback of their approach which we attempt to fix.

Input: The discrete sequence $\{(t_i, f_i), i = 0, 1, \ldots, N_0\}$, δ, ψ such that $\psi/\delta \in \mathbb{N}$ and the error tolerance E.
Output: A list of interpolation points L_p, a list of domain numbers L_d, a list of address L_a (describing which region is mapped to each domain) and a list of vertical scaling factors L_s.
Store the points (t_0, f_0), $(t_{0+\delta}, f_{0+\delta})$, $(t_{0+2\delta}, f_{0+2\delta})$ e.t.c. to the interpolation points list L_p;
Store the numbers of all domains, defined by those points, to L_d;
foreach *domain* n **do**
  foreach *region* m **do**
    Compute the vertical scaling factor s associated with region m and domain n (this is done via some geometric arguments or optimization;
    if $|s_n| > 1$ **then**
      Continue to the next region;
    end
    Compute the rest of the parameters of the map w_n (i.e. c_n, d_n);
    Denote by D_n the points of the sequence that lie inside the domain n and by R_m the points of the sequence that lie inside the region m;
    Map R_m through w_n, i.e. $\tilde{D}_n = w_n(R_m)$;
    Compute the SNR between D_n and $\tilde{D}_n$;
    if *SNR* $\geq E$ **then**
      Continue to the next domain;
    end
  end
  Find the maximum of the computed SNRs for the domain n;
  if *maximum SNR* $\geq E$ **then**
    Store the corresponding m to the addresses-list L_a;
    Store the corresponding s to the vertical scaling factors list L_s;
  end
  if *maximum SNR* $< E$ **then**
    Split the domain D_n into ψ/δ smaller domains (adding the respective interpolation points to the list L_p) and repeat the region-search for each one;
  end
end

Algorithm 3: Mazel - Hayes Methodology

Note that Algorithms 1, 2 and 3 work even if $\psi/\delta \notin \mathbb{N}$. In the following, however, we limit our interest in the case where there are numbers $\alpha, \beta \in \mathbb{N}$ such that $\delta = \alpha^\beta$, $\psi = \alpha^{\beta+1}$. In this case, for the reconstruction step, the DIA will produce exactly the values of the discrete sequence. Otherwise, the algorithm will produce more points than needed and we have to add an additional step to the procedure (for details see [18], [8], [9]).

3 Fractal Interpolation Functions of higher order

3.1 Definition

Although the method of Mazel and Hayes is quite popular and has been already implemented in many applications it has some serious disadvantages. If we try to approximate a smooth signal, even as simple as the points of a parabola, with the piecewise self affine model, we will obtain in the output much more information than needed. This motivates us to wonder what will happen if, instead of affine functions the q_n's are polynomials of degree greater than one. Here we consider this case, i.e.

$$q_n(x) = \sum_{k=0}^{d} a_{n,k} x^k,$$

for $n = 1, \ldots, N$, $d \geq 1$. Considering s_n and $a_{n,k}$ for $k = 2, \ldots, d$, as free parameters and taking into account conditions (3) after some algebra we obtain:

$$a_{n,1} = \frac{y_n - y_{n-1}}{\hat{x}_m - \hat{x}_{m-1}} - \frac{\sum_{k=2}^{d} a_{n,k}(\hat{x}_m^k - \hat{x}_{m-1}^k)}{\hat{x}_m - \hat{x}_{m-1}} - s_n \frac{\hat{y}_m - \hat{y}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}}, \tag{4}$$

$$a_{n,0} = \frac{y_{n-1}\hat{x}_m - y_n \hat{x}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}} - \frac{\sum_{k=2}^{d} a_{n,k}(\hat{x}_{m-1}^k \hat{x}_m - \hat{x}_m^k \hat{x}_{m-1})}{\hat{x}_m - \hat{x}_{m-1}} - s_n \frac{\hat{y}_{m-1}\hat{x}_m - \hat{y}_m \hat{x}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}}. \tag{5}$$

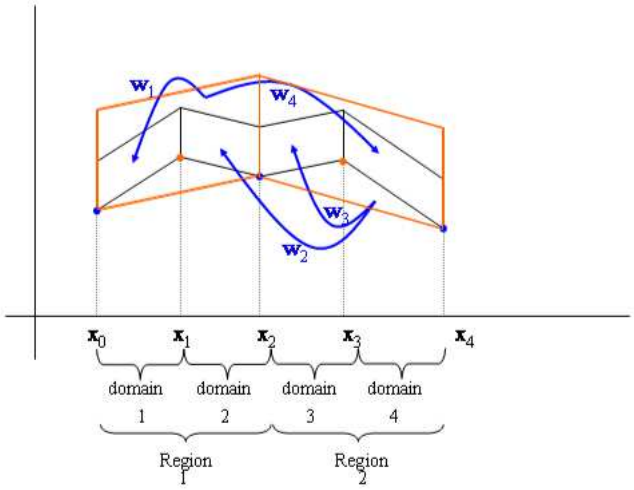


Figure 1: In the above figure, the set Δ consists of five interpolation points, while the set Q consists of three points. The corresponding connection vector is $V = (1, 2, 2, 1)$.

We will call the attractor of an RIFS of this form as RFIF of order d.

3.2 Modeling discrete sequences

In order to use RFIFs of order d to model discrete sequences, we need to devise a method for the efficient computation of the free parameters. To this end, we consider the discrete sequence $\{(t_i, f_i), i = 0, 1, \ldots, N_0\}$ and the sets $D_n = \{(t_{i(n)}, f_{i(n)}), \ldots, (t_{i(n)+\delta}, f_{i(n)+\delta})\}$ and $R_m = \{(t_{\hat{i}(m)}, f_{\hat{i}(m)}), \ldots, (t_{\hat{i}(m)+\psi}, f_{\hat{i}(m)+\psi})\}$, containing the points of the signal that lie inside the n-th domain and m-th region respectively. Since we limit our interest in the case where $\delta = \alpha^\beta$ and $\psi = \alpha^{\beta+1}$, we do not need all the points of R_m, but only those mapped at the corresonding points of D_n, therefore we redefine R_m as $R_m = \{(t_{\hat{i}(m)+\alpha\cdot l}, f_{\hat{i}(m)+\alpha\cdot l}), l = 0, \ldots, \delta\}$. Mapping the set R_m through a w_n of the form (2) we obtain the set $\tilde{D}_n = w_n(R_m) = \{(t_{i(n)+l}, \tilde{f}_{i(n)+l}), l = 0, \ldots, \delta\}$, where $\tilde{f}_{i(n)+l} = s_n f_{\hat{i}(m)+\alpha\cdot l} + q_n(t_{\hat{i}(m)+\alpha\cdot l})$. After some lines of algebra we conclude that $\tilde{f}_{i(n)+l} = A_{n,m,l} - \sum_{k=2}^{d} B_{m,l,k} \cdot a_{n,k} - s_n \Gamma_{m,l}$, where

$$A_{n,m,l} = A^{(2)}_{n,m} + A^{(1)}_{n,m} t_{\hat{i}(m)+\alpha\cdot l}, \qquad B_{m,l,k} = B^{(2)}_{m,k} + B^{(1)}_{m,k} t_{\hat{i}(m)+\alpha\cdot l} - t^k_{\hat{i}(m)+\alpha\cdot l},$$

$$A^{(1)}_{n,m} = \frac{y_n - y_{n-1}}{\hat{x}_m - \hat{x}_{m-1}}, \qquad A^{(2)}_{n,m} = \frac{y_{n-1}\hat{x}_m - y_n \hat{x}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}},$$

$$B^{(1)}_{m,k} = \frac{\hat{x}^k_m - \hat{x}^k_{m-1}}{\hat{x}_m - \hat{x}_{m-1}}, \qquad B^{(2)}_{m,k} = \frac{\hat{x}^k_{m-1}\hat{x}_m - \hat{x}^k_m \hat{x}^k_{m-1}}{\hat{x}_m - \hat{x}_{m-1}},$$

$$\Gamma_{m,l} = \Gamma^{(2)}_m + \Gamma^{(1)}_m t_{\hat{i}(m)+\alpha\cdot l} - f_{\hat{i}(m)+\alpha\cdot l}, \qquad \Gamma^{(1)}_m = \frac{\hat{y}_m - \hat{y}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}},$$

$$\Gamma^{(2)}_m = \frac{\hat{y}_{m-1}\hat{x}_m - \hat{y}_m \hat{x}_{m-1}}{\hat{x}_m - \hat{x}_{m-1}},$$

for $l = 0, \ldots, \delta$, $k = 2, \ldots, d$. We should choose the free parameters s_n, $a_{n,k}$, $k = 2, \ldots, d$ such that the square error

$$E(s_n, a_{n,2}, \ldots, a_{n,d}) = \sum_{l=0}^{\delta} \left(f_{i(n)+l} - \tilde{f}_{i(n)+l} \right)^2$$

is minimized. To simplify the notation, we define:

$$\boldsymbol{u} = \begin{pmatrix} s_n \\ a_{n,2} \\ \vdots \\ a_{n,d} \end{pmatrix}, \quad \boldsymbol{A} = \begin{pmatrix} \Gamma_{m,0} & B_{m,0,2} & \dots B_{m,0,d} \\ \Gamma_{m,1} & B_{m,1,2} & \dots B_{m,1,d} \\ \vdots & \vdots & \vdots \\ \Gamma_{m,\delta} & B_{m,\delta,2} & \dots B_{m,\delta,d} \end{pmatrix} \quad \boldsymbol{b} = \begin{pmatrix} A_{n,m,0} - f_{i(n)} \\ A_{n,m,1} - f_{i(n)+1} \\ \vdots \\ A_{n,m,\delta} - f_{i(n)+\delta} \end{pmatrix}.$$

therefore the error to be minimized is $E(\boldsymbol{u}) = \|\boldsymbol{A} \cdot \boldsymbol{u} - \boldsymbol{b}\|_2^2$. Considering some positive regularization coefficients $\gamma_1, \dots, \gamma_d$ and taking into account the fact that $|s_n| < 1$ we end up with the following convex optimization problem:

$$\begin{aligned} \textbf{minimize} \quad & E(\boldsymbol{u}) = \|\boldsymbol{A} \cdot \boldsymbol{u} - \boldsymbol{b}\|_2^2 + \sum_{k=1}^{d} \gamma_k u_k^2, \\ \textbf{subject to} \quad & u_1 \leq \epsilon, \\ & u_2 \geq -\epsilon, \end{aligned} \tag{6}$$

where ϵ should be chosen a priori close to 1 (say 0.99). We can easily check that problem (6) has a unique solution, which can easily be found. Before we give the solution we need the Lemma given below.

Lemma 3.1 *Consider the function $f : \mathbb{R}^n \to \mathbb{R} : f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c$, where A is a strictly positive matrix and the hyperplane $\boldsymbol{a}^\top \boldsymbol{x} = \zeta$. The aforementioned hyperplane is a supporting hyperplane to the level set $C_\eta = \{\boldsymbol{x} : \ f(\boldsymbol{x}) \leq \eta\}$ at the point $(\boldsymbol{x}_0, f(\boldsymbol{x}_0) = \eta)$, where*

$$\boldsymbol{x}_0 = \frac{1}{2}\boldsymbol{A}^{-1}(\lambda \boldsymbol{a} - \boldsymbol{b}), \quad \lambda = \frac{\zeta + \boldsymbol{a}^\top \boldsymbol{A}^{-1}\boldsymbol{b}}{\boldsymbol{a}^\top \boldsymbol{A}^{-1}\boldsymbol{a}}.$$

Proof. The supporting hyperplane to the set C_η at a point $(\boldsymbol{x}_0, \eta)$ is $\nabla f(\boldsymbol{x}_0)^\top \boldsymbol{x} = \nabla f(\boldsymbol{x}_0)^\top \boldsymbol{x}_0$. Therefore, since we want this hyperplane to be identified with the hyperplane $\boldsymbol{a}^\top \boldsymbol{x} = \zeta$, it is sufficient to find λ such that $2\boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{b} = \lambda \boldsymbol{a}$ and $(2\boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{b})^\top \boldsymbol{x}_0 = \lambda\zeta$. Solving for $\boldsymbol{x}_0$ gives the result.

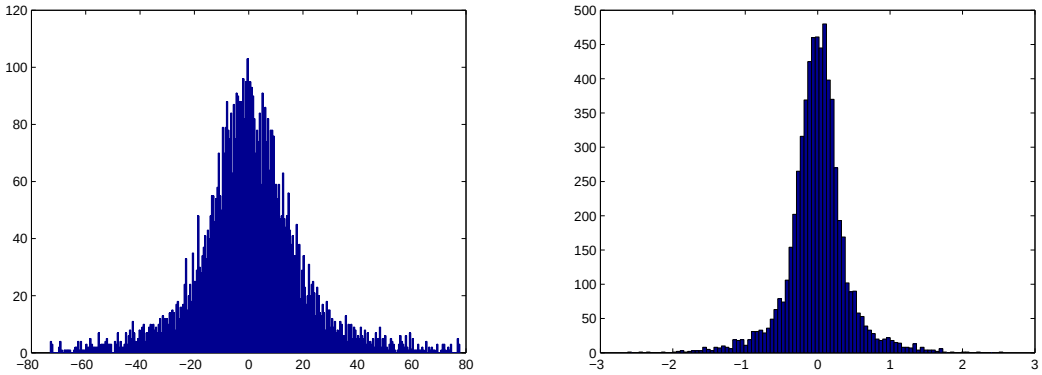
Now we focus our interest to problem (6). Since $E(\boldsymbol{u}) = \boldsymbol{u}^\top \left(\boldsymbol{A}^\top \boldsymbol{A} + \mathrm{diag}(\boldsymbol{\gamma})\right) \boldsymbol{u} - 2\boldsymbol{b}^\top \boldsymbol{A}\boldsymbol{u} + \boldsymbol{b}^\top \boldsymbol{b}$ is a strictly convex differentiable function (where $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_d)^\top$), it has a unique global minimum. Taking the gradient equal to 0, we can easily found that the global minimum is attained at $\boldsymbol{u}^{(m)} = \left(A^\top A + \mathrm{diag}(\boldsymbol{\gamma})\right)^{-1} \boldsymbol{b}^\top \boldsymbol{A}$. If $|u_1^{(m)}| \leq \epsilon$ then the solution of problem (6) is $\boldsymbol{u}^* = \boldsymbol{u}^{(m)}$. If $u_1^{(m)} > \epsilon$ then (using lemma 3.1)

$$\begin{aligned} \boldsymbol{u}^* &= \frac{1}{2}\left(\boldsymbol{A}^\top \boldsymbol{A} + \mathrm{diag}(\boldsymbol{\gamma})\right)^{-1} (\lambda \boldsymbol{a} + 2\boldsymbol{b}^\top \boldsymbol{A}), \\ \text{where } \lambda &= \frac{\epsilon - 2\boldsymbol{a}^\top \left(\boldsymbol{A}^\top \boldsymbol{A} + \mathrm{diag}(\boldsymbol{\gamma})\right)^{-1} \boldsymbol{b}^\top \boldsymbol{A}}{\boldsymbol{a}^\top \boldsymbol{A}^{-1}\boldsymbol{a}} \text{ and } \boldsymbol{a} = (1, 0, \dots, 0)^\top \in \mathbb{R}^d. \end{aligned}$$

Finally, if $u_1^{(m)} < -\epsilon$ we find similarly that

$$\begin{aligned} \boldsymbol{u}^* &= \frac{1}{2}\left(\boldsymbol{A}^\top \boldsymbol{A} + \mathrm{diag}(\boldsymbol{\gamma})\right)^{-1} (\lambda \boldsymbol{a} + 2\boldsymbol{b}^\top \boldsymbol{A}), \\ \text{where } \lambda &= \frac{-\epsilon - 2\boldsymbol{a}^\top \left(\boldsymbol{A}^\top \boldsymbol{A} + \mathrm{diag}(\boldsymbol{\gamma})\right)^{-1} \boldsymbol{b}^\top \boldsymbol{A}}{\boldsymbol{a}^\top \boldsymbol{A}^{-1}\boldsymbol{a}} \text{ and } \boldsymbol{a} = (1, 0, \dots, 0)^\top \in \mathbb{R}^d. \end{aligned}$$

Note that in the case where Algorithm 3 partitioned a domain to several smaller ones due to inadequate region mapping as described in the algorithm, the above relations still hold if we consider $\alpha = \psi/(\text{domain-width})$. Moreover, it is easy to modify the procedure to handle the case where the interpolation points are not equidistant.



(a) (b)

Figure 2: The histograms of the coefficients $a_{n,2}$ (a) and $a_{n,3}$ (b) found by algorithm 3 with $d = 3$ applied to a typical one dimensional signal. In (b) we can see some points that deviate significantly from the mean of the distribution (close to 3 or -3).

3.3 Implementation - Comparison

Using Algorithm 3, with the modifications in the computation of the map parameters which we described above (i.e. we replace the steps of the vertical scaling factor computation and the c_n, b_n computation with the minimization procedure) and removing the if statement regarding the value of the vertical scaling factor (which is not needed since the minimization procedure ensures that $|s| < 1$) one should expect to get better results than the Mazel-Hayes approach, since our methodology uses polynomials of degree higher than 1 and minimizes the error analytically. In fact, it is evident that in the trivial case where we try to approximate a sequence which has been derived as points of a piecewise self affine FIF, our approach will give the same parameters up to order one, filling the rest of the polynomial coefficients (up to order d) with zeros. In this section we demonstrate the efficiency of the proposed model, comparing it with the approach of Mazel and Hayes in the area of one-dimensional signal compression.

To achieve compression, we must come up with a quantization method for the model parameters. For $d = 1$, Mazel and Hayes showed that a uniform quantizer with 2^6 or 2^8 levels for the quantization of the vertical scaling factors gives satisfactory results. This is expected since we can see from examples that the vertical scaling factors s_n follow an almost uniform distribution. In the non-linear case a little more effort is needed. The positive (likewise for the negative) values of $a_{n,k}$ (i.e. the coefficients of x^k terms in the model) now follows exponential distributions and there are some cases (luckily only a few) where the values deviate from the mean significantly (see figure 2). To confront with this problem we add one more step to Algorithm 3, so that the domains with model parameters that deviate from the mean significantly (i.e. the values greater than a cutoff threshold for the positive coefficients or alternatively the values lower than a cutoff threshold for the negative coefficients) are feeded again to the algorithm. The cutoff threshold is taken as the value greater than the 99% of the positive coefficients (or alternatively the value lower than the 99% of the negative coefficients). The algorithm searches again for a region that is "best-mapped" to the respective domain. However, each time the minimization process yields model parameters that deviate from the mean as explained above, the deviated parameters are set equal to the respective cutoff threshold and the algorithm continues as before (i.e. the set $\tilde{D}_n$ is formed and the SNR is computed). Consequently, we apply different uniform quantizers for the model parameters of different orders. In addition for higher order coefficients we need more quantizer levels since the model becomes more sensitive to errors.

Figure 3 shows the results of the methodology applied to two sound samples for various values of error tolerance E (from 10 to 50). The first is a sample from a man singing a ballad and the second a sample from an electric guitar. We can see in the figure that there is a point which the model (for various values of d) gets its maximum compression ratio and the decrease of error tolerance E doesn't contribute. This is due

to the fact that as d increases, we need to store more coefficients for each domain. There is a value for E such that almost none of the domains get splitted. If we decrease E more, the algorithm finds less suitable regions for each domain, which results to a decrease of PSNR while the compression ratio remains the same, since the same number of coefficients is stored. For the quantization of the coefficients we used 8 bits for the vertical scaling factors s_n of the Mazel-Hayes approach. For the higher order model with $d = 2$, we used 8 bits for the s_n and 8 bits for the $a_{n,2}$ coefficients. For the same model with $d = 3$, we used 8 bits for the s_n and 9 bits for the $a_{n,2}$ and $a_{n,3}$ coefficients. Finally, when $d = 4$, we used 8 bits for the s_n and 11 bits for the $a_{n,2}$, $a_{n,3}$, $a_{n,4}$ coefficients. For the regularization coefficients we chose $\boldsymbol{\gamma} = (1, \ldots, 1)$ and $\epsilon = 0.99$. The values of δ and ψ were chosen to be 32 and 64 respectively.

Acknowledgments. The author would like to thank professors Leoni Dalla and Sergios Theodoridis for their continuous support.

References

[1] M. F. Barnsley, Fractal functions and interpolation, Constr. Approx., 2 (1986), 303–329.

[2] M. F. Barnsley, Fractals everywhere, 2nd ed., Academic Press Professional, 1993.

[3] M. F. Barnsley, S. Demko, Iterated function systems and the global construction of fractals, Proc. Roy. Soc. London Ser A, 399 (1985), 243–275.

[4] M. F. Barnsley, J. H. Elton and D. P. Hardin, Recurrent iterated function systems, Constr. Approx., 5 (1989), 3–31.

[5] P. Bouboulis and L. Dalla, Fractal interpolation surfaces derived from fractal interpolation functions, J. Math. Anal. Appl., 336 (2007), 919–936.

[6] P. Bouboulis, L. Dalla, A general construction of Fractal Interpolation Functions on grids of $\mathbb{R}^n$, European J. Appl. Math., 18 (2007), 449–476.

[7] P. Bouboulis, L. Dalla, V. Drakopoulos, Construction of Recurrent Bivariate Fractal Interpolation Surfaces and Computation of their Box-Counting Dimension, J. Approx. Theory, 141 (2006), 99–117.

[8] P. Bouboulis L. Dalla and V. Drakopoulos, Image Compression Using Recurrent Bivariate Fractal Int. J. Bifurcation Chaos Appl. Sci. Eng., 141 (7) (2006), 1–9.

[9] P. Bouboulis and V. Drakopoulos and S. Theodoridis, Image compression using affine fractal interpolation surfaces, Fractals, 14 (4) (2006), 1–11.

[10] O. I. Craciunescu, S. K. Das, J. M. Poulson and T. V. Samulski, Three-Dimensional Tumor Perfusion Reconstruction Using Fractal Interpolation Functions, IEEE Trans. Biom. Eng., 48 (4) (2001), 462–473.

[11] L. Dalla, Bivariate Fractal Interpolation Functions on Grids, Fractals, 10 (2002), 53–58.

[12] G. Donovan, J. S. Geronimo, D. P. Hardin, P. R. Massopust, Construction of orthogonal wavelets using fractal interpolation functions, SIAM Journal of Mathematical Analysis, 27 (1996), 1158–1192.

[13] D. P. Hardin, B. Kessler, P. R. Massopust, Multiresolution Analysis Based on Fractal Functions, J. Approx. Theory, 71 (1992), 104–120.

[14] J. S. Geronimo, D. P. Hardin, P. R. Massopust, Fractal functions and wavelet expansions based on several scaling functions, J. Approx. Theory, 78 (1994), 373–401.

[15] G. Kim, A. Baros, Downscaling of remotely sensed soil moisture with a modified fractal interpolation method using contraction mapping and ancillary data, Remote Sensing of Environment, 83 (3) (2002), 400–413.

[16] P. Manousopoulos, V. Drakopoulos, T. Theoharis, P. Stavrou, Effective Representation of 2D and 3D Data Using Fractal Interpolation, October 2007, Proceedings of the 2007 International Conference on Cyberworlds.

[17] P. Manousopoulos, V. Drakopoulos and T. Theoharis, Fractal Active Shape Models, in Kropatsch W. G., Kampel M. and Hanbury A. (eds), Computer Analysis of Images and Patterns, Springer-Verlag, Berlin and Heidelberg, 2007, 645Ű-652.

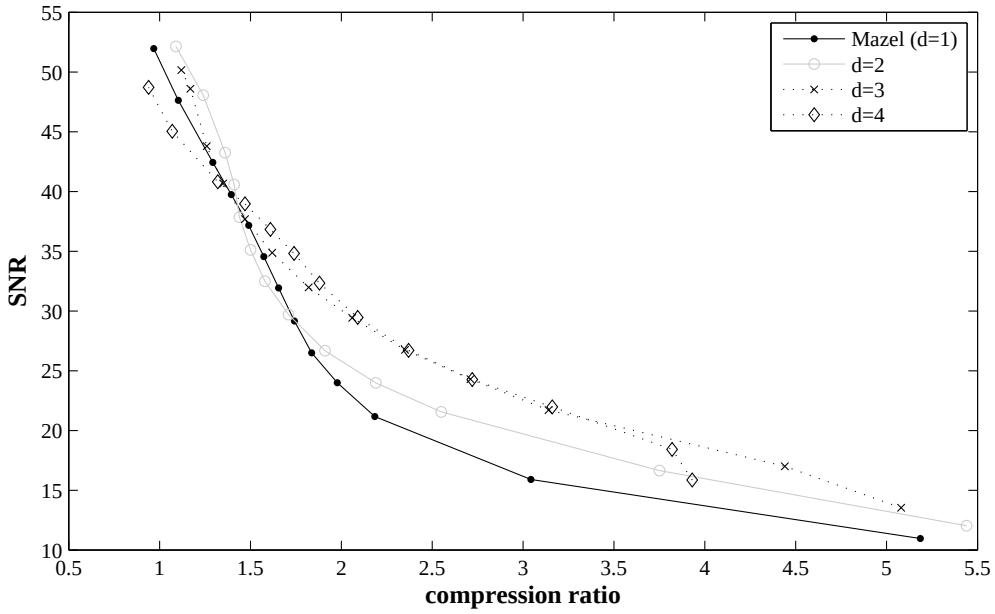
[18] D. S. Mazel and M. H. Hayes, Using iterated function systems to model discrete sequences, IEEE Trans. Signal Process., 40 (1992) 1724–1734.

[19] D. S. Mazel, Representation of discrete sequences with three-dimensional iterated function systems, IEEE Trans. Signal Process., 42 (1994), 3269–3271.

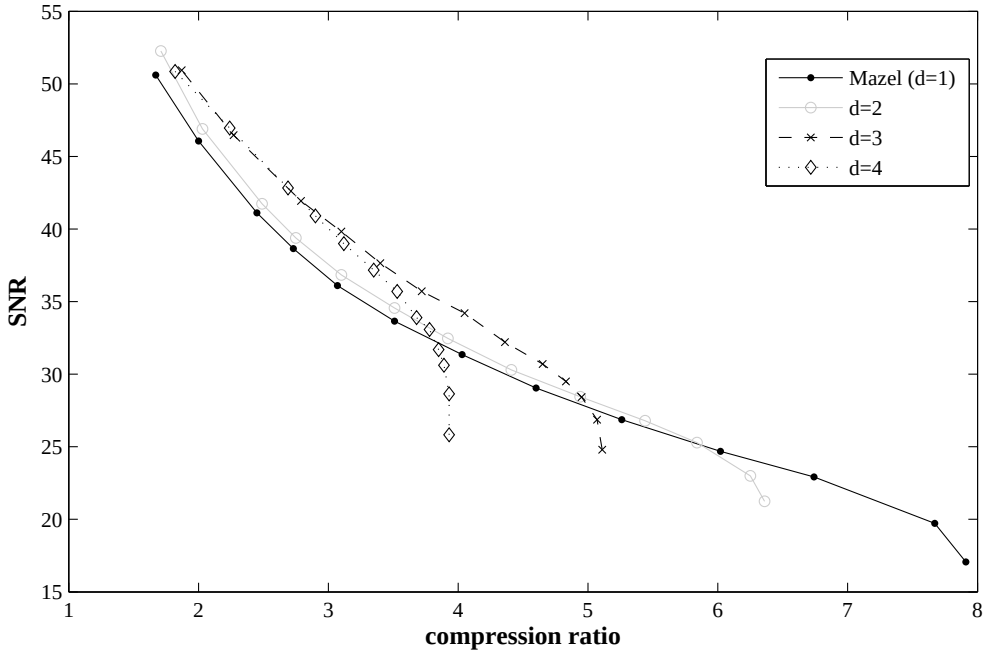
[20] A. I. Pennand and M. H. Loew, Estimating fractal dimension with fractal interpolation function models, IEEE Trans. Med. Imag., 16 (1997), 930–937.

[21] J. R. Price and M. H. Hayes, Resampling and reconstructing with fractal interpolation functions, IEEE Signal Process. Letters, 5 (1998), 228–230.

[22] Nailiang Zhao, Construction and application of fractal interpolation surfaces, The Visual Computer, 12 (1996), 132–146.



(a)



(b)

Figure 3: Two different wav files consisting of 32769 samples have been compressed with the Mazel-Hayes methodology and the RFIF of order d methodology, for various values of d. The second sound sample (b) is far more complex than the first (a).