Prediction of Dynamic Systems through the Identification of Basic Patterns in Multivariate Mathematical Landscapes Using a Genetic Algorithm

Angel Kuri
Centro de Investigación en Computación del IPN
Unidad Profesional Zacatenco, Av. Juan de Dios Batis S/N
Mexico, D.F.
email:akuri@pollux.cic.ipn.mx

ABSTRACT: We describe an order-based genetic algorithm which is able to extract the most relevant features of mathematical landscape by selecting the form of the approximant as well as determining the numerical values of the coefficients which best characterize the relationship between a set of dependent variables and a set of independent ones. We show that the data may be arbitrary and are not required to comply with typical statistical assumptions such as homoscedasticity, independence or normality. We also show that, due to the said independence from statistical restrictions, the method allows us to apply it to time series which may be characterized so that the resulting model exhibits predictive properties..

KEYWORDS: multivariate approximation, order-based genetic algorithms, pattern recognition, forecasting

I. INTRODUCTION

The characterization of a system is usually based on the knowledge derived from its prior behavior: a series of observations from some process may be expressed as $[(\vec{x}_1, y_1),\ldots,(\vec{x}_n, y_n)]$, where $\vec{x}_i = (v_1,\ldots,v_\Pi)_i$ and $y_i = (v_{\Pi+1})_i$, with y_i considered a dependent parameter and $\vec{x}_i$ being a vector of independent ones. Typically, the researcher has access to a limited set of the system's measurable parameters from which to extract such knowledge. This limitation arises because a full set is usually too large to be manipulated in practice. One is, therefore, forced to decide which subset to consider as relevant. In theory, there is a smallest complete defining subset ($\Lambda = (l_1,\ldots,l_{p+1})$) which we would like to *identify* and *categorize*. To "*identify* Λ" means that we want to determine the smallest $p \in \Pi$ which permits us to describe the system so that if any of the l_i is excluded the description would be incomplete. To "*categorize* Λ" means that we want to determine the relative importance of each of the l_i in Λ. One possibility is to find a set of relationships of the form $f_k = f(l_1,\ldots,l_{k-1},l_{k+1},\ldots,l_{p+1}) \forall k$. The problem (which we call Λ - *determination* or $\Lambda\Delta$) has been approached in various essentially different ways, depending on the particular intended goal.

The approach we follow is to try to obtain the mathematical relationship between subsets of Λ, thus,

$$f(v_{j_1},\ldots,v_{j_m}) = f(v_{k_1},\ldots,v_{k_n}) \qquad j_i \neq k_i \forall i \tag{1}$$

Usually the problem of finding the expression in (1) is addressed in more limited terms, namely, by expressing a dependent variable in terms of several independent ones. Multivariate interpolation [1], [2], [3] and approximation [4], [5], [6] have been widely treated and are still subject of much research. Superficially, the problem of multivariate approximation does not seem to bear a direct relation to the $\Lambda\Delta$ problem. However, the problem may be stated in terms of finding a set of coefficients in (1) which satisfy certain a priori conditions. For instance, one may specify that (1) *collocates* the data [7] or that it *approximates* the data in some sense [8]. The variables considered in (1), in this approach, constitute the l_i's and the form of (1) determines the categorization of Λ.

In this paper we address the problem of modeling a system by finding the best (*Tchebyshev*, *minimax* or L_∞) approximant in a multidimensional landscape. We are particularly interested in solving the $\Lambda\Delta$ problem for phenomena which occur in time with the ultimate goal of achieving reliable forecasting. This is done by finding a system of the form

$$f_j(v_1,\ldots,v_p)=\sum_{i_1=0}^{g_1}\cdots\sum_{i_p=0}^{g_p} m_{i_1\ldots i_p} C_{i_1\ldots i_p} v_1^{i_1}\cdots v_p^{i_p} \qquad (2)$$
$$j=1,\ldots,p$$

where

p is the number of parameters,

$$m_k=\begin{cases}0 \text{ if } C_k\notin\Psi\\ 1 \text{ if } C_k\in\Psi\end{cases}$$

and

Ψ is the set of relevant terms

We find Λ, and categorize it by finding the mathematical relationship between its elements. We arbitrarily define the *order* of Ψ, i.e. its cardinality. Thereafter, a Genetic Algorithm (GA) determines: a) which terms are included and b) the values of the coefficients of (2). We also discuss how to obtain an approximant of the form

$$f(v_1,\ldots,v_p)=\sum_{i_1=0}^{g_1}\cdots\sum_{i_p=0}^{g_p} C_{i_1\ldots i_p} v_1^{i_1}\bullet\bullet\bullet v_p^{i_p} \qquad (3)$$

in the minimax sense. This problem has been previously discussed in [9]. There, however, it is approached as the one of solving overdetermined systems whose data set is originally in $\Re^P$. We recognize that the problem of multivariate approximation in p space is equivalent to the approximation of overdetermined systems in m space ($m = 1 + \prod_{i=1}^{p}(g_i + 1)$ provided that linear independence may be guaranteed. We present our method in a more intuitive way, not resorting to topological concepts. We find a bound on the deviation of the approximant derived from such conditioning. In part III we discuss an order-based GA which allows for the identification of all the elements of (2). In part IV we discuss the application of the method to forecasting. The method is illustrated by applying it to data taken from the stock market. Finally, in part V we make some remarks on the $\Lambda\Delta$ problem and offer some conclusions.

II. MATHEMATICAL FOUNDATIONS

The general problem was stated before: a series of observations from some process (for instance, a physical system or a formal dynamical system) takes the form of a set of pairs, $[(\bar{x}_1,y_1),\ldots,(\bar{x}_n,y_n)]$, where $\bar{x}_i=(v_1,\ldots,v_p)_i$ and $y_i=(v_{p+1})_i$ Then:

a) The data which characterize the system under study may be put in the form of a bidimensional table, where every column corresponds to one of the parameters of such system. We do not make any *a priori* consideration as to the relevance of any of the variables in the characterization. The number of columns will be denoted as $p+1$ where p is the number of independent variables. The number of rows will be denoted by n, and the data set by N.

b) We wish to express one of the variables as a function of the remaining p. By convention, we shall assume that the $(p+1)$-th variable is the dependent variable and that the remaining p are independent.

c) We choose, *a priori* the basic form of the approximant: a full polynomial form of the kind in (3)

d) We choose a metric which allows us to precisely define the distance between the approximant and the table's values.

In the problems of interest, typically, the systems are overdetermined. Hence, it is necessary to define a precise way to measure the distance of the candidate curve representing the relationship between the variables and the data itself. The most frequently used norm is L_2, the least squares approximation. Unfortunately, L_2 has the drawback of giving rise to Hilbert matrices [10] which are badly ill-conditioned. Usually this inconvenience is circumvented by using an orthogonal basis [11]. This approach works well iff the data are found on a grid [12]. When data is experimental, we cannot guarantee such characteristic. It imposes a very restrictive sampling space, particularly when such space is multidimensional. Therefore, we have chosen the L_∞, *minimax* or *Tchebyshev* norm [13], which does not present any of the limitations just mentioned. We wish to find the coefficients $C_{0\ldots0}, C_{0\ldots1},\ldots,C_{g_1\ldots g_p}$ in (3).

A. Solution for a Set of Size m.

We arbitrarily select *m* vectors from the table, where

$$m = 1 + \prod_{i=1}^{p} (g_i + 1) \tag{4}$$

and, from (3):

$$e_i = | f_i - (\sum C \quad v)_i |$$

The last equation has two solutions:

$$e_i = \pm [f_i - (\sum C \quad v)_i]$$

Without loss of generality, we may solve for

$$e_i + (\sum C \quad v)_i = f_i \tag{5}$$

Let $e_q = \max \{ |e_1|, \ldots, |e_m| \}$ and $e_i = h_i e_q$. Clearly, $h_i \leq 1$. It is e_q that we wish to minimize. From (5) we may put

$$h_i \; e_q + (\sum C \quad v)_i = f_i \tag{6}$$

If we write the system of (6) in matrix form we have

$$\begin{bmatrix} h_1 & (v_1^0 \ldots v_p^0)_1 & \cdots & (v_1^{g_1} \ldots v_p^{g_p})_1 \\ . & \ldots & . & . \\ h_m & (v_1^0 \ldots v_p^0)_m & \cdots & (v_1^{g_1} \ldots v_p^{g_p})_m \end{bmatrix} \begin{bmatrix} e_q \\ C_{0 \ldots 0} \\ \ldots \\ C_{g_1 \ldots g_p} \end{bmatrix} = \begin{bmatrix} f_1 \\ \ldots \\ f_m \end{bmatrix} \tag{7}$$

Which we may write as

$$\mathbf{A} \; \mathrm{x} \; \mathbf{C} = \mathbf{f} \tag{8}$$

From Cramer's rule, the value of e_q is

$$e_q = \frac{\begin{vmatrix} f_1 & (v_1^0 \ldots v_p^0)_1 & \cdots & (v_1^{g_1} \ldots v_p^{g_p})_1 \\ . & \ldots & . & . \\ f_m & (v_1^0 \ldots v_p^0)_m & \cdots & (v_1^{g_1} \ldots v_p^{g_p})_m \end{vmatrix}}{\begin{vmatrix} h_1 & (v_1^0 \ldots v_p^0)_1 & \cdots & (v_1^{g_1} \ldots v_p^{g_p})_1 \\ . & \ldots & . & . \\ h_m & (v_1^0 \ldots v_p^0)_m & \cdots & (v_1^{g_1} \ldots v_p^{g_p})_m \end{vmatrix}} \tag{9}$$

If we denote the numerator of (9) with Δ_f and the cofactors of the first column of the denominator with Δ_i we can write equation (9) as

$$e_q = \frac{\Delta_f}{+ h_1 \Delta_1 + \ldots + h_m \Delta_m} \tag{10}$$

Let $s = sgn(x)$ be the sign of *x*. Since we wish to minimize the absolute value of e_q we must choose the values which *maximize* the absolute value of the denominator of (10). This is achieved by: a) Selecting the greatest absolute value of the h_i's and b) Making $s_i = sgn(\Delta_i)$ or $s_i \neq sgn(\Delta_i)$.

We know that $\max \{ h_1, \ldots, h_m \} = 1$. Hence, the minimax fit for the set implies that the e_i's have the same absolute value and that the signs of such e_i's are all equal to those signs of the Δ_i's. That is, e_q is minimized if

$$e_q = \frac{\Delta_f}{+ s_1 \Delta_1 + \ldots + s_m \Delta_m} \tag{11}$$

It is impossible to attain a Vandermonde [14] ordering when $p > 1$. Therefore, the easiest way to calculate the s_i's follows from the following theorem [15].

Cofactor Theorem.

If the elements of one column of a determinant are multiplied by the cofactors of a different column and summed the result is zero.

□

Let us denote the elements of the determinant in the denominator of (9) with $d_{11},...,d_{1m}\,;d_{21},...,d_{2m}\,;\ ...\ ;d_{m}1,...,d_{mm}$. We can put

$$\begin{aligned}
&\Delta_1 d_{12} + \Delta_2 d_{22} + ... + \Delta_m d_{m2} = 0\\
&\Delta_1 d_{13} + \Delta_2 d_{23} + ... + \Delta_m d_{m3} = 0\\
&..\\
&..\\
&..\\
&\Delta_1 d_{1m} + \Delta_2 d_{2m} + ... + \Delta_m d_{mm} = 0
\end{aligned} \tag{12}$$

This is a system with *m* unknowns and only *m-1* conditions. We may choose, however, the value of any of the Δ_i's (say Δ_m) arbitrarily and solve system (12) for the remaining *m-1* Δs. For this to be possible we must assume that the system satisfies Haar's condition [16].

If we make that $\Delta_m = -1 = \Im_m$ the system in (12) will be

$$\begin{bmatrix} (v_1^0 ... v_p^0)_1 & \cdots & (v_1^0 ... v_p^0)_{m-1} \\ ... & ... & ... \\ (v_1^{g_1} ... v_p^{g_p})_1 & \cdots & (v_1^{g_1} ... v_p^{g_p})_{m-1} \end{bmatrix} \times \begin{bmatrix} \Im_1 \\ ... \\ \Im_{m-1} \end{bmatrix} = \begin{bmatrix} (v_1^0 ... v_p^0)_m \\ ... \\ (v_1^{g_1} ... v_p^{g_p})_m \end{bmatrix} \tag{13}$$

where $\Im_i = k\ \Delta_i$. *k* may take any positive or negative value different from zero. The s_i's are obtained, then, from $s_i = sgn(\Im_i)$. That s_m may be arbitrary follows from the fact that only the sign of $\Im_i$ is relevant, and not its magnitude and that, on the other hand, the denominator of (9) is maximized choosing either $s_i = sgn(\Delta_i)\forall\ i$ or $s_i \neq sgn(\Delta_i)\forall\ i$.

We may then write matrix **A** of (8) as

$$A = \begin{bmatrix} s_1 & (v_1^0 ... v_p^0)_1 & \cdots & (v_1^{g_1} ... v_p^{g_p})_1 \\ . & ... & . & . \\ s_m & (v_1^0 ... v_p^0)_m & \cdots & (v_1^{g_1} ... v_p^{g_p})_m \end{bmatrix} \tag{14}$$

and the coefficients resulting from $\mathbf{C} = \mathbf{f}\,\mathbf{A}^{-1}$ are the minimax coefficients that we look for in an arbitrary set of *m* vectors.

B. Solution for a Set of Size n.

To find the global e_q, that is, the one which solves the minimax fit for the *n* original vectors we establish an iterative ascent algorithm, analogous to Remes' [17] algorithm.

Ascent Algorithm

1. Make $i \leftarrow 1$.
2. Choose an arbitrary set of vectors M_i of size *m*.
   This set will be called the *inner set*; those vectors not included in this set will be included in another set E_i which we call the *external set* and that is, clearly, of size *n - m*.
3. Find matrix $\mathbf{A}_i$ as per (14).
4. Find the minimax coefficients $\mathbf{C}_i$ and the maximum fit error $(e_q)_i$ for the internal set.
5. Calculate the maximum fit error for the external set, where

$$[e_f]_i = \max\ \{\ |(f_{1...p})_j - (\sum_{i_1=0}^{g_1} ... \sum_{i_p=0}^{g_p} C_{i_1...i_p} v_1^{i_1} ... v_p^{i_p})_j|\ \} \tag{15}$$

$$\bar{v}_j \in E_i$$

6. If $[e_q]_i \geq [e_f]_i$, *end algorithm.*
7. {If $[e_q]_i < [e_f]_i$}

Exchange a vector in M_i for one in E_i such that $|[e_q]_{i+1}| > |[e_q]_i|$. (This algorithm is sometimes called the *exchange algorithm*).

8. Do $i \leftarrow i + 1$.
9. Go to step *3*.

□

The algorithm converges since $i < t \Rightarrow |[e_q]_i| < |[e_f]_i|$ where t denotes the index of the target set (M_t). This notion may be formalized in the following theorem [18].

C. Generalization of the Ascent Algorithm.

The method outlined in section *A* is based in the obtention of a map of each of the vectors of the original problem in $\Re^p$ to $\Re^m$ and the subsequent obtention of the s_i to conform **A**. From the exchange theorem it is implied that every set of vectors in **A** does satisfy Haar's condition. The probability that any of the sets does not satisfy it is different from zero. In such cases, we wish to establish a method to find the solution to the so called *singular systems* [19].

Once matrix **A** whose vectors are in $\Re^m$ has been conformed, it will not comply with Haar's condition if

$$A^i = K\ A^j ; (i,\ j) \leq p ; i \neq j ; K \equiv constant$$

or

$$A_i = K\ A_j ; (i,\ j) \leq p ; i \neq j ; K \equiv constant$$

If we can guarantee that no such files or columns are linearly dependent Haar's condition may be maintained. To achieve this, we induce random perturbations in the vectors $v_i = [(v_1)_i , ..., (v_p)_i , (v_{p+1})_i]$ of the original set *N* making

$$(V_1)_i = (v_1)_i (1 + d_i)$$

$$(V_2)_i = (v_2)_i (1 + d_{n+i})$$

.....

$$(V_{p+1})_i = (v_{p+1})_i (1 + d_{np+i})$$

$$d_i = d_H \times r_i$$

where:

$$r_i = random\#\ [i = 1, ..., n(p + 1)]$$

and

$$0 < r_i < 1 ; d_H << 1$$

d_H is constant and, therefore, each vector $\boldsymbol{v}_i$ in set *N* is replaced by another vector $\boldsymbol{V}_i = [(V_1)_i , (V_2)_i , ..., (V_{p+1})_i]$ which exhibits a random percentual displacement relative to $\boldsymbol{v}_i$ which is bounded by d_H. The resulting set N^* satisfies Haar's condition for an appropriate d_H even if *N* does not. No subset of the points in $\Re^m$ must be collinear in none of the *m* dimensions. We displace slightly each point randomly. By definition, the displacements do not maintain the proportionality that gives rise to the linear dependence and we avoid that any of the minors in (11) be equal to zero. The replacement of $\boldsymbol{v}$ by $\boldsymbol{V}$ in some cases does not achieve the linear independence directly. Hence, we apply the following *Data Set Algorithm*. This algorithm receives four input parameters:

a) The value of the initial perturbation $(d_H)_0$.

b) The maximum acceptable perturbation (m_p).

c) An adjustment factor (Φ).

d) The maximum number of stabilization iterations (h).

Data Set Algorithm.

```
for i = 1 to h
  while true
        read N original data
        from N original data obtain N*
        if linearly dependent system
              d_H ← d_H × Φ
              if d_H > m_p
                    r ← rand()        (0 < r < 1)
                    Φ ← Φ × (1 + r)
                    d_H ← (d_H)_0 × Φ
                    loop while
              endif
        else
              exit for
        endif
  endwhile
endfor
```

Subindex t denotes the target set. M_t denotes, therefore, the internal target set, E_t the external target set, etc. Here v_t and V_t denote any vector in the target sets M_t and M_t^* for *N* and N^* respectively.

Clearly, when we fit to these new data we are merely approximating the required solution. We now establish a bound to the difference between the minimax approximation to N^* in V_t and in v_t.

Let

$$F_t^* = \sum_{i_1=0}^{g_1} \cdots \sum_{i_p=0}^{g_p} C_{i_1,\ldots,i_p} (V_1^{i_1})_t \cdots (V_p^{i_p})_t$$

be the value of the minimax polynomial obtained from *N* and evaluated in set M_t^*; let

$$F_t = \sum_{i_1=0}^{g_1} \cdots \sum_{i_p=0}^{g_p} C_{i_1,\ldots,i_p} (v_1^{i_1})_t \cdots (v_p^{i_p})_t$$

be the value of the polynomial obtained from *N* and evaluated in any point of M_t; and $TOD(x) \equiv$ *terms of degree x in* F_t; $TOO(x) \equiv$ *terms of order x in* F_t. Then,

$$F_t^* = (1 + d_H)^0 \times TOD(0) + (1 + d_H)^1 \times TOD(1) + \ldots + (1 + d_H)^{g_1 + \ldots + g_p} \times TOD(g_1 + \ldots + g_p)$$

where, for simplicity, we have assumed the maximum value of d_i, that is, d_H. Expanding the value of the binomials, we have

$$F_t^* = TOD(0) + \ldots + TOD(g_1 + \ldots + g_p) + d_H [1 \times TOD(1) + \ldots + (g_1 + \ldots + g_p) \times TOD(g_1 + \ldots + g_p)] + TOO(d_H^2) + \ldots + TOO(d_H^{g_1 + \ldots + g_p})$$

Since $d_H << 1$ we may write

$$F_t^* \approx TOD(0) + \ldots + TOD(g_1 + \ldots + g_p) + d_H [1 \, x \, TOD(1) + \ldots + (g_1 + \ldots + g_p) \times TOD(g_1 + \ldots + g_p)]$$

$$F_t^* - F_t = d_H [1 \times TOD(1) + \ldots + (g_1 + \ldots + g_p) \times TOD(g_1 + \ldots + g_p)]$$

and, successively,

$$d_H \sum_{i=1}^{g_1 + \ldots + g_p} i \times TOD(i) = d_H \{ [F_t - TOD(0)] + [F_t - TOD(0) - TOD(1)] + \ldots +$$

$$[\, F_t - TOD(0) - TOD(1) - \ldots - TOD(g_1 + \ldots + g_p - 1)\,] \,\}$$

$$= d_H \sum_{i=1}^{g_1+\ldots+g_p} \{\, F_t - \sum_{j=0}^{i-1} TOD(j) \,\}$$

Taking absolute values,

$$|F_t^* - F_t| \approx \left| d_H \sum_{i=1}^{g_1+\ldots+g_p} [\, F_t - \sum_{j=0}^{i-1} TOD(j) \,] \right| \tag{16}$$

Since $d_H > 0$ and $|\sum_i a_i| \leq \sum_i |a_i|$ then

$$|F_t^* - F_t| \leq d_H \sum_{i=1}^{g_1+\ldots+g_p} \left| F_t - \sum_{j=0}^{i-1} TOD(j) \right|$$

$$|F_t^* - F_t| < d_H \times (g_1 + \ldots + g_p)\,|F_t|$$

From which we conclude

$$\boxed{\left| \frac{F_t^* - F_t}{F_t} \right| < \left[\sum_i g_i \right] d_H} \tag{17}$$

On the other hand, we may write

$$(i)\ f_t - F_t = e_q$$

$$(ii)\ f_t + f_t d_H - F_t^* = e_q^*$$

Subtracting *(ii)* from *(i)*

$$F_t^* - F_t - f_t d_H = e_q - e_q^*$$

We know, from (16) that

$$F_t^* - F_t - f_t d_H = d_H [\sum_{i=1}^{g_1+\ldots+g_p} \{\, F_t - \sum_{j=0}^{i-1} TDG(j) \,\} - f_t \ ]$$

hence

$$|e_q - e_q^*| \leq d_H \ \{ \sum_{i=1}^{g_1+\ldots+g_p} |F_t - \sum_{j=0}^{i-1} TDG(j)| + |-f_t| \,\}$$

$$|e_q - e_q^*| < d_H \{\, |(g_1 + \ldots + g_p)\,|F_t| + |f_t| \,\}$$

and since $|f_t| < |F_t| \, x \, 2$; $e_q > 0$ *and* $e_q^* > 0$ then

$$\boxed{|e_q^* - e_q| < \left(2 + \left[\sum_i g_i \right] \right) d_H \left| F_t \right|} \tag{18}$$

□

Equations (17) and (18) allow us to conclude that the process of conditioning leads to a relative difference between the function evaluated in N^* and the function evaluated in N of the order of d_H. On the other hand, the difference of the minimax errors (e_q) in N^* and N is, also, of the order of d_H. Since the value of d_H is, by definition, very small, the approximant obtained from the altered set is very close to the original approximant obtained from N.

III. ORDER-BASED GENETIC ALGORITHM

Using the method outlined in III it is possible to characterize a set of arbitrary data. However, the number of coefficients from (3) grows exponentially with the number of independent variables and the maximum degrees stipulated. For 10 variables and $g_1 = \ldots = g_{10} = 6$ the number of coefficients is $7^{10} \approx 282.475 \cdot 10^6$. One solution whose characterization implies such number of data is useless. To avoid such problem we replace the system in (3) by the following:

$$f(v_1,\ldots,v_p)=\sum_{i_1=0}^{g_1}\cdots\sum_{i_p=0}^{g_p} m_{i_1\ldots i_p} C_{i_1\ldots i_p} v_1^{i_1}\ldots v_p^{i_p} \tag{19}$$

where the variables $m_{i_1,\ldots,i_p}$ can only take the values *0* or *1* and their purpose is to exclude/include the coefficient which they multiply. We introduce a new parameter to the problem: the maximum number of desired coefficients (g). Now we must find a) the g most convenient coefficients to minimize e_q and b) the values of the chosen coefficients. We seek, therefore, two solution vectors. Vector m has $\prod_{i=1}^{p}(g_i+1)$ elements; vector **C** has g elements. These are highly non-linear problems and *NP complete* [20].

Our problem may be solved if we apply a GA whose genome follows the next conventions:

a) It is of length $\Gamma=\prod_{i=1}^{p}(1+g_i)$.

b) Each of the bits in the genome corresponds to a coefficient of (3). The coefficients are indexed from left to right in a way such that these indices run from 1 to Γ. For example, if p = 3 and g_1 = 1, g_2 = 2 y g_3 = 2, then the relation between powers and indices is illustrated in diagram 1.

Powers associated to the Coefficients

001	002	003	...	120	121	122

Indices of the Coefficients

1	2	3	...	16	17	18

Diagram 1. Relation between powers and indices.

c) There are g positions in which the genome takes the value *1*; Γ - g positions where it takes the value *0*.

d) A *1* in the genome means that the corresponding coefficient exists (m = 1); on the contrary, a *0* in the genome means that the coefficient does not exist (m = 0).

Let $G=(g_1,g_2,\ldots,g_p)$ be a set of maximum degrees of cardinality p and $f(v_1,\ldots,v_p)$ the minimax approximant of (3). Given the index *N* of the coefficients of the genome, it is possible to find the powers associated to the terms (t_i) from the following expression:

$$t_i=\left|\frac{(N-1)-\sum_{j=i}^{i-1} t_j \prod_{k=j+1}^{p}(g_k+1)}{\prod_{j=i+1}^{p}(g_j+1)}\right| \tag{20}$$

$$i=1,\ldots,p$$

where $a > b \Rightarrow \{\sum_{a}^{b} x\Delta 0\ ;\ \prod_{a}^{b} x\Delta 1\}$. The terms of the powers ($t_1,\ldots,t_p$) correspond to positions $1,\ldots,p$ of the monomial. Similarly, given the terms $t_1,\ldots,t_p$ we find the corresponding index from the expression

$$N=1+\sum_{i=1}^{p} t_i \prod_{j=i+1}^{p}(g_j+1) \tag{21}$$

These genomes may be genetically processed in a way such that the fitness function is the error e_q and the process becomes one of *minimization* of such error. In each generation two components are derived per individual:

a) The minimax fitness error (e_t).

b) The coefficient vector **C** for the ascent algorithm.

Let E_t be the global e_t, i.e. the minimax error for the population of the GA given that the number of *1*s in the genome remains equal to g. The population evolves towards the combination of coefficients that minimizes E_t. The GA must maintain a genome whose number of *1*s (w) is always equal to g. The genetic operators of crossover and mutation

are limited in the sense that they cannot give rise to valid genomes in all cases. Consequently, we must apply a new operator, called the *order preserving (* $\Omega\Pi$) operator, to be discussed in the sequel. The parametric fit of fixed order does not seek the solution to the problem (vector **C**) directly. When looking for the form of the solution (the best individual of order g) we are led to the solution of the true problem: the set of coefficients which best adjust the data.

A. The Eclectic Genetic Algorithm.

The GA applied to the solution of the problem herein is not a *Canonical* GA (CGA). Rather, we used an *Eclectic* GA (EGA) which has been discussed in detail elsewhere [21]. The *EGA* incorporates the following:

i) *Full elitism*
ii) *Deterministic selection*
iii) *Annular crossover*
iv) *Random mutation hillclimbing*
v) *Adaptative determination of the following parameters:*
a) n: the number of offspring.
b) P_c*: the crossover probability*
c) P_m*: the mutation probability.*
d) P_H*: the probability of hillclimbing.*
e) h_H *: the hillclimber's function evaluations.*

Notice that the EGA proper is an independent module which outputs a population and inputs the fitness for every generation.

The *External Fitness Evaluation Module (EFEM)* receives, as parameters, the *i-th* population's (*PA*) and fitness' (*FA*) arrays. It calculates the values for *FA* for every individual and returns this result to the EGA. The operators in the EGA do not preserve the cardinality of **C** (g). Therefore, an external operator was implemented to this effect.

B. The $\Omega\Pi$ Operator.

Letting Γ be the length of the genome and $\Gamma(i)$ its *i-th* gene, the preservation of g is achieved by applying the following scheme.

a) Obtain the order of the genome (w).

b) If $w = g$ stop.

c) $\begin{matrix} w > g \Rightarrow s \leftarrow 1 ; t \leftarrow 0 \\ w < g \Rightarrow s \leftarrow 0 ; t \leftarrow 1 \end{matrix}$.

d) Generate a random number *k*, where $1 \le k \le \Gamma$.

e) Setting, $i \leftarrow k$ scan the genome until $\Gamma(i) = s$ (If the end of the string is reached, set $i \leftarrow 1$ and continue scanning). Then $\Gamma(i) \leftarrow t$ and go to (d).

Repeat steps (d) and (e) until $w = g$.

□

This procedure is applied to every individual in the population. The adjusted individual consists of a string of g *1*s and $\Gamma - g$ *0*s. The position of each of the *1*s is then mapped into the corresponding combination of powers of the variables as per (26). The resulting polynomial is the summation of all such monomials. It is interesting to note that this is possible because the terms in A_k (from equation (7)) for $k \neq 1$ may be arbitrary. Here we start by assuming that these terms are derived from the right hand side of (3) and then leaving out some of them.

The *EFEM* then calculates the fitness (e_t *) for each of the adjusted individuals, according to the FAA.* Each one of these e_t *'s determines the desirable components of the genome which will lead the EGA to* Ψ *. When the EGA receives the FA* it does not only update the fitnesses but the genomes as well. Thus, it must also incorporate the adjusted population from *PA*. This last is the one that is incorporated to the EGA's mechanism and is subject to further genetic treatment. The $\Omega\Pi$ operator, therefore, individually adjusts g *and* allows for the required update of the adjusted population.

IV. FORECASTING.

We are interested in the case where data are obtained in different instants of time. When the arguments lie outside the range of observation we may predict the behavior of the system. Of course, once **C** is obtained, the resulting polynomial is susceptible of accepting any argument. However, due to the highly nonlinear relationships between the terms of the

approximant, it is unlikely that the arguments outside of the data domain yield significant results. There is, however, a simple strategy which allows us to avoid this limitation.

In what follows, we shall assume that there are 5 independent variables which we denote with $v_1,\ldots,v_5$ and that represent snapshots of the system in time. The data corresponding to $v_1,\ldots,v_5$ is, in fact, a historical record where every row in the table represents a state of the system. It is possible to express one of the variables as a function of itself, displaced with respect to the entries in the data table. For instance, $f(v_1,\ldots,v_5)$ can be made to correspond to the value of v_5 at time $t+dt$. That is,

$$f(v_1,\ldots,v_5)=v_5(t+dt)=f(v_1(t),\ldots,v_5(t)) \tag{22}$$

where dt determines how many time intervals we wish to let pass between the moment of observation of the phenomenon and a hypothetical moment of prediction. In this case, the dependent variable is one of the independent variables displaced in time. We call this variable the *prospection variable*. The EGA together with the ascent algorithm and the data conditioning method permit us to find a mathematical expression for the relation that exists between an undefined number of variables and the *prospection variable*.

A. Experiments.

We illustrate the method employed in the next example.

Here we selected a table of 61 data vectors. We considered 5 independent variables. Each variable represents a sample of the values (at close) of one of five stocks from the American stock market. For each of the independent variables we followed the next procedure:

a) Each of the independent variables was put as a function of the other four and *itself* with a fixed time offset, as in equation (1), that is,

$$\begin{aligned} v_1(t+dt) &= f_1(v_1(t),\ldots,v_5(t)) \\ &\ldots \\ v_5(t+dt) &= f_5(v_1(t),\ldots,v_5(t)) \end{aligned} \tag{23}$$

b) The values of the coefficients and the m's were calculated as described above (the genome size is 1024).

In each generation we defined a population size of 50. Here $n = 61$, $p = 5$ and $g = 20$. Thus, we establish a map from $\Re^5$ to $\Re^{20}$. In order to condition the data, we specified $(d_H)_0 = 10^{-6}$, $\Phi = 8$ and $h = 5$. The last three parameters are input to the *Data Set Algorithm*. We also specified $k = 0.01$ and no sampling, since n is comparatively small.

c) The genetic algorithm settings for the were successively used in (19) five times, yielding $(v_1(t+dt),\ldots,v_5(t+dt))$.

The Genetic Algorithm we utilized is self-adaptive and we, therefore, specified the following ranges: $a)\ 0.9 \le P_c \le 1$, $b)\ P_m = 0.005$, $c)\ 4 \le No.\ of\ descendants \le 5$

We calculated four different fits, yielding 1 forecasting function set each, as shown in diagram 2.

Option	v_1	v_2	v_3	v_4	v_5	Dt
a	2	3	2	3	2	5
b	3	3	3	3	3	5
c	2	3	2	3	2	10
d	3	3	3	3	3	10

Diagram 2. Conditions for Experimental Fit.

In cases (a) and (c) the highest degree a monomial may reach is $3\times4\times3\times4\times3=432$; in cases (b) and (d) it is. $4\times4\times4\times4\times4=1024$ □

Once each of the five functions is calculated, we proceed to find the combination of stocks to be bought/sold (z) which optimizes the *projected* gain.

This is done by maximizing the expected gain (the one derived from (23)) subject to a maximum amount (c) to invest. The optimization is also carried out with the EGA, as follows:

From the *prospection functions* ($v_i(t+dt)$) already found for each of the variables:

a) An initial population of 100 individuals where every individual represented a combination of stocks was randomly generated. Every element in the combination stood for the amount of stocks to negotiate.

b) Every individual was tested to see if it complied with c. If it did not, it was qualified proportionally to the c-*compliance* of every element; if it did, it was qualified according to the projected gain.

c) An individual which maximized the expected gain was evolved.

The only restriction is that the net value of the stocks to be operated should not exceed c. Therefore, negative as well as positive values for $v_i(t+dt)$ are acceptable; a negative $v_i(t+dt)$ means a sale of stocks. A detailed description of the optimization process is described in [23]. The genome size indicated there corresponds to the concatenated binary representation of 5 numbers $n_1,\ldots,n_5$: the amount of stocks to negotiate for each stock. We also ran this optimization process using the actual values of the stocks. We are then able to compare the predicted best combination (z) and the actual best combination (z') of stocks in every case.

B. Results.

The results of the experiments are shown in figures 16. In all cases we specified a maximum amount of $c = 50{,}000$ monetary units to be invested. In the *Gain* axis we may find the value of the operated stocks. In the *Displacement* axis we graphed the time units (days) into the system. For instance, a displacement of *15* marks the behavior of the system 15 time units after its initiation.

In figure 1 we show the graph where fits for an offset (dt) of 5 were found. That is, we obtained the expressions for $v_i(t+5)\,\forall\, i$ and then use this expressions to extrapolate out of the data domain. Of course, in our experiments we knew such data beforehand.

Three lines are shown: a) The *Projected* line corresponds to the gains which the polynomial has predicted by forecasting the values of $v_i(t)$ and finding z; b) The *Actual* line shows the actual gain which would have been obtained by operating the amount of stocks in z on the true data; c) The *True* line corresponds to the optimization process applied to the actual values. That is, we obtained the optimal combination of stocks (z') for the true data.

As shown in the boxes, we used degrees *2-3-2-3-2* for the 5 variables, giving a maximum order of 432. The combination of degrees was arbitrary.

In figure 2 we illustrate the case where the size of the genome is increased to $4^5 = 1024$. The experiment just described was repeated. Notice that projected and actual gains are closer to each other than in the former experiment.

In figures 3 and 4 similar results for an offset of 10 ($dt = 10$) are shown. If we compare figures 1 and 3 (degrees *2-3-2-3-2* with $d_H = 5$ and $d_H = 10$, respectively), we find that the actual values are lower for $d_H = 10$ than for $d_H = 5$ for low range displacements but higher in the high range displacements.

Similar results are seen from figures 2 and 4. In figure 5 we display the actual gain for the displacements of 15, 25, 35, 45 and 55 time units, as well as the average gain. Here, we see that best gains were obtained in the following order: *D33333D10*, *D23232D10*, *D33333D5* and *D23232D5* (where *D12345Dyy* stands for *maximum degrees: 1,2,3,4,5* and $dt = yy$). As expected, combinations of potentially higher degree monomials yield better fits and, consequently, better forecasts. However, contrary to our initial expectations, longer data offsets resulted in better forecasts. With the benefit of hindsight, we may remark that, since we are including longer range data, the predictive process is derived from a richer Λ. Our initial intuition was that larger offsets would imply greater uncertainty.

In figure 6 we show the average actual and true gains compared. It is interesting to see how close the true values are to the predicted values. It is also striking that for long range displacements z was actually better than z'. This, of course, is due to chance.

V. CONCLUSIONS.

Several assumptions have been tacit in the previous discussion:

a) That the relationship between the independent and the dependent variables (P) is amenable to be expressed as a summation of monomials.

b) That the set of maximum degrees has been selected in a manner that guarantees that the monomials are able to express P in a satisfactory way.

c) That the minimax norm is appropriate to express P.

d) That the g terms found by the EGA are enough to approximate P properly.

Each of the assumptions reflects a decision on the part of the designer. When the system under study arises from a formal dynamical system it is possible to determine the convenience of the decisions implied in the above assumptions. In other cases one must rely on more general considerations.

(a) The method may be applied, without changes, to other kind of approximants. A notable alternative are the approximants of the form

$$f(v_1,\ldots,v_p)=\sum_{i_1=0}^{g_1}\cdots\sum_{i_p=0}^{g_p}\frac{C_{i_1\ldots i_p}}{v_1^{i_1}\cdots v_p^{i_p}} \tag{24}$$

which are, since they include poles, richer than polynomial approximants.

(b) We should be relatively certain that the set **G** is properly selected so as to allow for the relationship between the parameters to be properly expressed while, at the same time, precluding an impractical size of Γ. This compromise is problem dependent but it must also take into consideration the platform in which the algorithm is implemented since a very large Γ increases the chance of numerical instability.

(c) L_∞ approximants may be shown [24] to share many of the desirable properties of L_2 approximants. However, as is well known, they are sensitive to atypical or spurious data. In this sense it is interesting to point out that *sampling* E_i as mentioned before, does not only improve the efficiency of the Ascent Algorithm but also minimizes the said sensitivity.

(d) The size of g is, intuitively, proportionally better the larger it is. However, our intuition is broken for very large g for numerical reasons. Once again, this calls for implementation considerations.

The method outlined solves $\Lambda\Delta$ finding the *form* of the approximant by disregarding those genomes which yield poor fitness (given the invariance of g). This takes care of the categorization part. The identification part is more subtle. Since the method yields the values of the coefficients of (2), it is relatively simple to identify those combinations of parameters which are relevant for the characterization of the phenomenon. As a simple example, we examine the set of terms/coefficients displayed next.

t_i	C_i		t_i	C_i
01)	**00000**	**-.2060387612E+03** -	02) 00001	+.1042358033E+02
03)	00002	-.3130780051E-01 -	04) 00033	+.9654923995E-09
05)	00133	-.4101563075E-10 -	06) 02100	+.1245050193E-02
07)	02211	-.2053642610E-08 -	08) 02222	+.2112579764E-11
09)	03100	-.1089647019E-04 -	10) 03332	-.8456768892E-18
11)	11103	-.1068709766E-07 -	12) 11121	-.2075532452E-08
13)	12003	-.3184965292E-07 -	14) 23212	+.1036968640E-15
15)	30011	+.3587274833E-06 -	16) 30121	-.1218516243E-09
17)	30122	+.2344535944E-11 -	18) 30123	-.6820513199E-14
19)	30130	+.8157530681E-11 -	20) **33331**	**-.9126712347E-23**

Here $g = 20$, and $g_i = 3$, $i = 1,\ldots,5$. The first and 4th columns enumerate the *C*s; the 2nd and 5th ones show the degrees for the selected monomials; the third and 6th ones display the coefficients. It is clear that the monomial $v_1^0 v_2^0 v_3^0 v_4^0 v_5^0 = 1$ is of primordial relevance. Similarly, we find that the monomial $v_1^3 v_2^3 v_3^3 v_4^3 v_5^1$ is relatively unimportant; its "weight" is of the order of 10^{-23}. These two entries have been boldfaced.

When applied to actual data, the method exhibits interesting predictive properties. The reason is, basically, that is has been able to extract the most relevant features from the mathematical landscape defined by the purported approximants: those that best solve $\Lambda\Delta$. In this particular mathematical landscape, one of the axis is time and, in this view, it should not be surprising that a good minimax fit would yield a characterization with the said properties.

REFERENCES

[1] de Boor, C., and Devore, R., "Approximation by smooth multivariate splines", Trans. Amer. Math. Soc., 276, pp. 775-788, 1983.

[2] Chui, C. K., and Lai, M. J., "Vandermonde determinants and Lagrange interpolation in $\Re^8$", in Nonlinear and Convex Analysis, Lin and Simmons, eds., Marcel Dekker, New York, pp. 23-36, 1987.

[3] Gasca, M. and Maetzu, J. I., "On Lagrange and Hermite interpolation in $\Re^k$", J. Approx. Theory, 42, pp. 1-14, 1982.

[4] Cheney, E. W., "Multivariate Approximation Theory: Selected Topics", CBMS-NSF Regional Series in Applied Mathematics, S.I.A.M., Philadelphia, 1986.

[5] Diliberto, S.P. and Straus, E.G., "On the approximation of a function of several variables by the sum of functions of fewer variables", Pacific J. Math. 1, pp. 195-210, 1951.

[6] Feng, G., "A counterexample on global Chebyshev approximation", J. Approx. Theory, 1987.

[7] Rice, J. R., "Numerical Methods, Software and Analysis", pp. 79-120, McGraw-Hill Book Company, 1983.

[8] Cheney, E. W.,, "Introduction to Approximation Theory", McGraw-Hill Book Company, 1966.

[9] Cheney, E. W., op. cit., pp. 45-50.

[10] Scheid, F., "Theory and Problems of Numerical Analysis", McGraw-Hill Book Company, 7th. edition, 1997.

[11] Shampine, L., and Allen, R., "Numerical Computing: an introduction", W.B. Saunders, pp. 43-53, 1974.

[12] Chui, C. K., "Multivariate Splines", Chap. 9, "Multivariate Interpolation", CBMS-NSF Regional Series in Applied Mathematics, S.I.A.M., Philadelphia, 1988.

[13] Cheney, E. W.,, "Introduction to Approximation Theory", Chapter 2, McGraw-Hill Book Company, 1966.

[14] Cheney, E. W., op. cit., pp. 58-64.

[15] Cheney, E. W., op. cit., pp. 39.
[16] Willard, S., "General Topology", Addison-Wesley, pp. 116-128, 1970.
[17] Remes, E., "Sur le calcul effectif des polynomes d'approximation de Tchebyshef", Comptes Rendus Hebdomadaires, Seances de l'Academie de Sciences, 199, pp. 337-340, Paris, 1934.
[18] Cheney, E. W., op. cit., pp. 45-46.
[19] Kuri, A., and Galavíz, J., "An eclectic genetic algorithm for function optimization with nonlinear constraints", IEEE Transactions on Evolutionary Computation, *to be published*.
[20] Kuri, A., "A Universal Eclectic Genetic Algorithm", Proceedings of Iberamia 1998, *to be published*.
[21] Kuri, A., and Galavíz, J., "An eclectic genetic algorithm for function optimization with nonlinear constraints", IEEE Transactions on Evolutionary Computation, *to be published*.
[22] Cheney, E. W.,, "Introduction to Approximation Theory", McGraw-Hill Book Company, chap. 3, 1966.

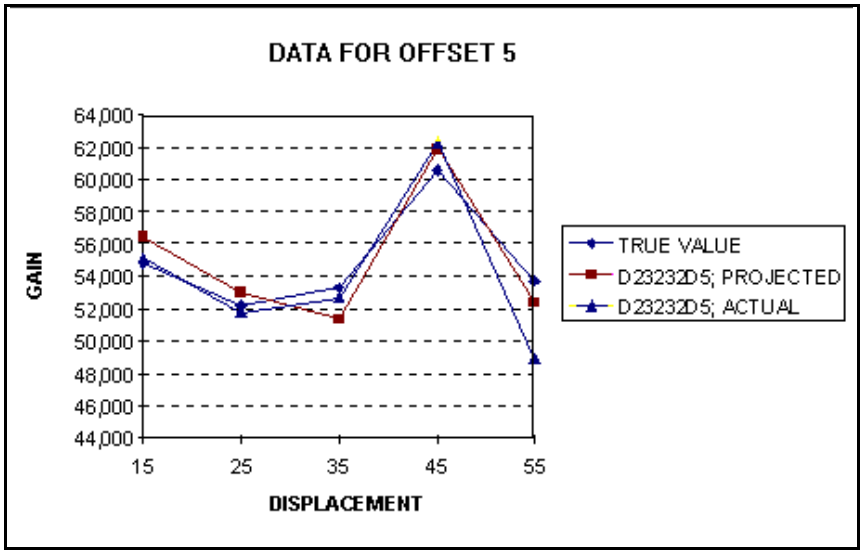

Figure 1: Runs for an Offset of 5 with 2-3-2-3-2 Degrees.

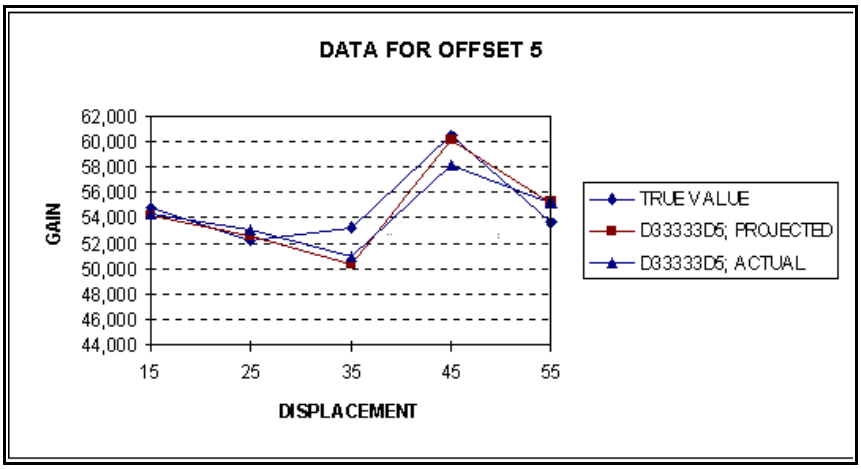

Figure 2: Runs for anOffset of 5 with 3-3-3-3-3 Degrees.

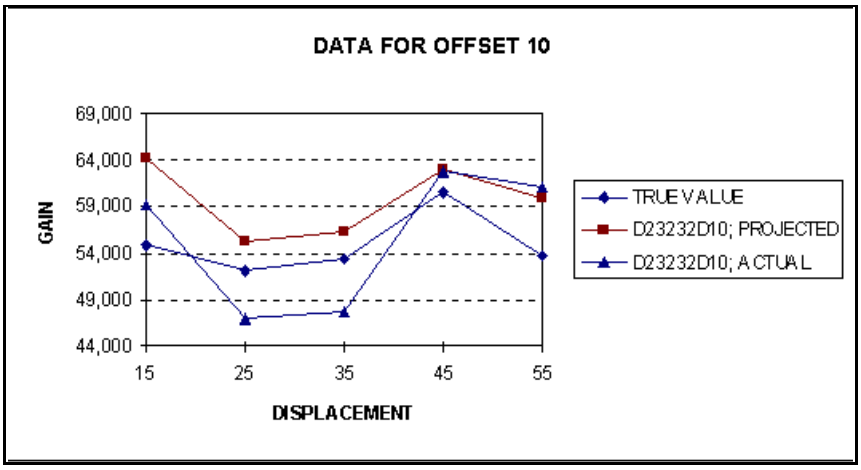

Figure 3: Runs for an Offset of 10 with 2-3-2-3-2 Degrees.

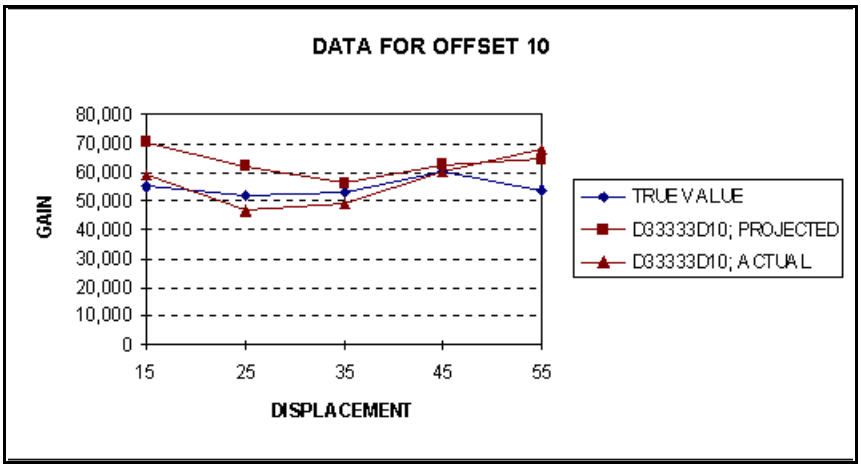

Figure 4: Runs for an Offset of 10 with 3-3-3-3-3 Degrees.

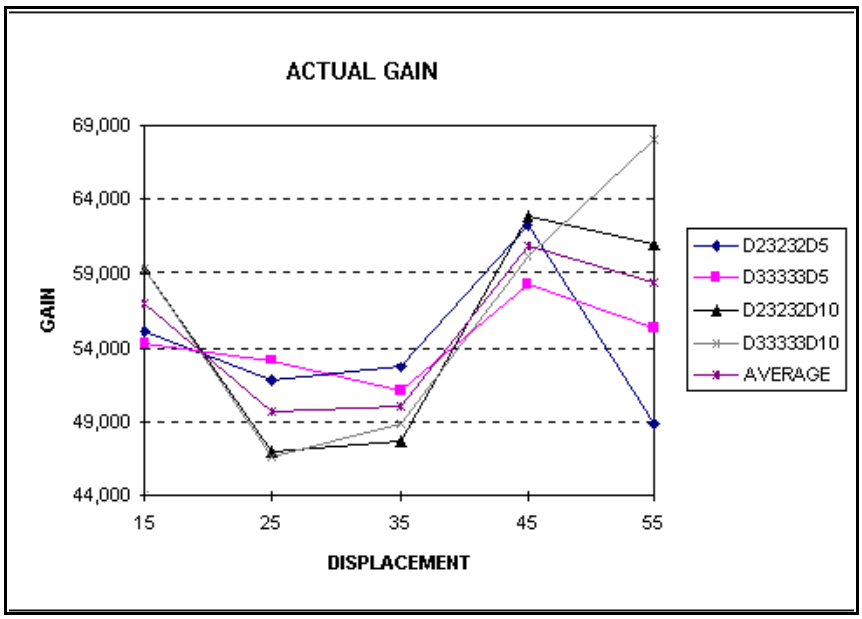

Figure 5: Actual Gain for all Sets of Experiments.

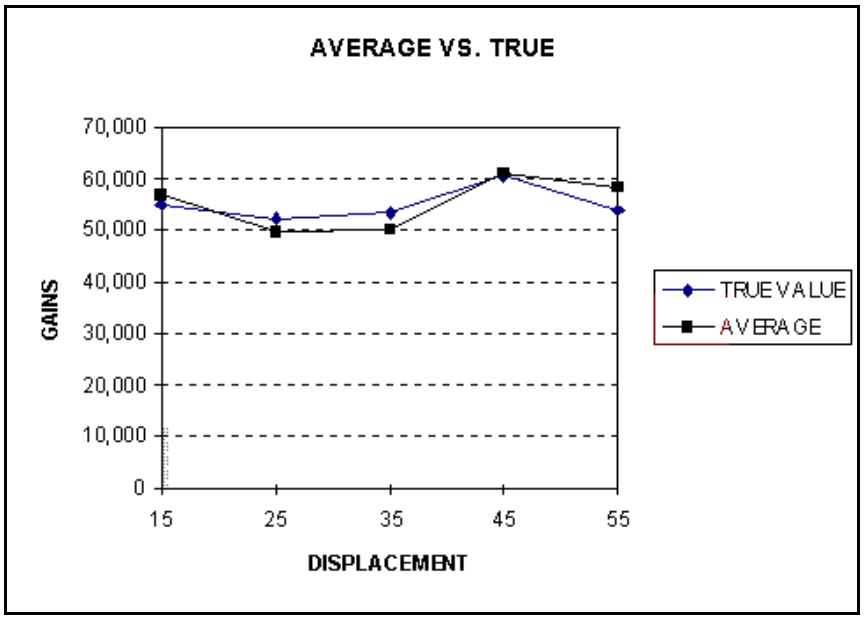

Figure 6: Average Gain and True Gain Compared.