

Adaptation, Performance and Vapnik-Chervonenkis Dimension of Straight Line Programs

José L. Montaña¹, Cruz E. Borges¹, César L. Alonso², and José L. Crespo¹

¹ Departamento de Matemáticas, Estadística y Computación,
Universidad de Cantabria

{montanj1, borgesce, luis.crespo}@unican.es

² Centro de Inteligencia Artificial, Universidad de Oviedo
Campus de Viesques, 33271 Gijón, Spain
calonso@aic.uniovi.es

Abstract. We discuss here empirical comparison between model selection methods based on Linear Genetic Programming. Two statistical methods are compared: model selection based on Empirical Risk Minimization (ERM) and model selection based on Structural Risk Minimization (SRM). For this purpose we have identified the main components which determine the capacity of some linear structures as classifiers showing an upper bound for the Vapnik-Chervonenkis (VC) dimension of classes of programs representing linear code defined by arithmetic computations and sign tests. This upper bound is used to define a fitness based on VC regularization that performs significantly better than the fitness based on empirical risk.

Key words: Genetic Programming, Linear Genetic Programming, Vapnik-Chervonenkis dimension

1 Introduction

Throughout these pages we study some theoretical and empirical properties of a new structure for representing computer programs in the GP paradigm. This data structure –called *straight line program* (slp) in the framework of Symbolic Computation ([1])– was introduced for the first time into the GP setting in [2]. A slp consists of a finite sequence of computational assignments. Each assignment is obtained by applying some functional (selected from a specified) to a set of arguments that can be variables, constants or pre-computed results. The slp structure can describe complex computable functions using less amount of computational resources than GP-trees. The key point for explaining this feature is the ability of slp’s for reusing previously computed results during the evaluation process. Another advantage with respect to trees is that the slp structure can describe multivariate functions by selecting a number of assignments as the output set. Hence one single slp has the same representation capacity as a forest of trees (see [2] for a complete presentation of this structure).

Linear Genetic Programming (LGP) is a GP variant that evolves sequences of instructions from an imperative programming language or from a machine language. The structure of the program representation consists of assignments of operations over constants or memory variables called registers, to another registers (see [3] for a complete overview on LGP). The GP approach with slp's can be seen as a particular case of LGP where the data structures representing the programs are lists of computational assignments.

We study the practical performance of *ad-hoc* recombination operators for slp's. We apply the SLP-based GP approach to solve some instances of the symbolic regression problem. Experimentation done over academic examples uses a weak form of structural risk minimization and suggests that the slp structure behaves very well when dealing with bounded length individuals directed to minimize a compromise between empirical risk and non-scalar length (i.e. number of non-linear operations used by the structure). We have calculated an explicit upper bound for the Vapnik-Chervonenkis dimension (VCD) of some particular classes of slp's. This bound constitutes our basic tool in order to perform structural risk minimization of the slp structure.

2 Straight Line Programs: Basic Concepts and Properties

Straight line programs are commonly used for solving problems of algebraic and geometric flavor. An extensive study of the use of slp's in this context can be found in [4]. The formal definition of slp's we provide in this section is taken from [2].

Definition 1. Let $F = \{f_1, \dots, f_n\}$ be a set of functions, where each f_i has arity a_i , $1 \leq i \leq n$, and let $T = \{t_1, \dots, t_m\}$ be a set of terminals. A straight line program (slp) over F and T is a finite sequence of computational instructions $\Gamma = \{I_1, \dots, I_l\}$, where for each $k \in \{1, \dots, l\}$,

$$I_k \equiv u_k := f_{j_k}(\alpha_1, \dots, \alpha_{a_{j_k}}); \text{ with } f_{j_k} \in F, \\ \alpha_i \in T \text{ for all } i \text{ if } k = 1 \text{ and } \alpha_i \in T \cup \{u_1, \dots, u_{k-1}\} \text{ for } 1 < k \leq l.$$

The set of terminals T satisfies $T = V \cup C$ where $V = \{x_1, \dots, x_p\}$ is a finite set of variables and $C = \{c_1, \dots, c_q\}$ is a finite set of constants. The number of instructions l is the length of Γ .

Usually an slp $\Gamma = \{I_1, \dots, I_l\}$ will be identified with the set of variables u_i introduced at each instruction u_i , thus $\Gamma = \{u_1, \dots, u_l\}$. Each of the non-terminal variables u_i can be considered as an expression over the set of terminals T constructed by a sequence of recursive compositions from the set of functions F . Following [2] we denote by $SLP(F, T)$ the set of all slp's over F and T .

Example 1. Let F be the set given by the three binary standard arithmetic operations $F = \{+, -, *\}$ and let $T = \{1, x_1, x_2, \dots, x_n\}$ be the set of terminals. Any slp Γ in $SLP(F, T)$ represents a n -variate polynomial with integer coefficients.

An output set of a slp $\Gamma\{u_1, \dots, u_l\}$ is any set of non-terminal variables of Γ , that is $O(\Gamma) = \{u_{i_1}, \dots, u_{i_t}\}$. The function computed by a slp $\Gamma = \{u_1, \dots, u_l\}$ over F and T with set of terminal variables $V = \{x_1, \dots, x_p\}$ and with output set $O(\Gamma) = \{u_{i_1}, \dots, u_{i_t}\}$, denoted by $\Phi_\Gamma : I^p \rightarrow O^t$, is defined recursively in the natural way and satisfies $\Phi_\Gamma(a_1, \dots, a_p) = (b_1, \dots, b_t)$, where b_j stands for the value of the expression over V of the non terminal variable u_{i_j} when we replace each variable x_k with a_k ; $1 \leq k \leq p$.

3 Vapnik-Chervonenkis Dimension of Families of *slp*'s

In the last years GP has been applied to a range of complex learning problems including that of classification and symbolic regression in a variety of fields like quantum computing, electronic design, sorting, searching, game playing, etc. A common feature of both tasks is that they can be thought of as a supervised learning problem (see [5]) where the hypothesis class \mathcal{C} is the search space described by the genotypes of the evolving structures. In the seventies the work by Vapnik and Chervonenkis ([6], [7], [8]) provided a remarkable family of bounds relating the performance of a learning machine (see [9] for a modern presentation of the theory). The Vapnik- Chervonenkis dimension (VCD) is a measure of the capacity of a family of functions (or learning machines) as classifiers. The VCD depends on the class of classifiers. Hence, it does not make sense to calculate VCD for GP in general, however it makes sense if we choose a particular class of computer programs as classifiers. Our aim is to study in depth the formal properties of GP algorithms focusing on the analysis of the classification complexity (VCD) of straight line programs.

3.1 Estimating the VC dimension of slp's parameterized by real numbers

The following definition of VC dimension is standard. See for instance [7].

Definition 2. *Let \mathcal{C} be a class of subsets of a set X . We say that \mathcal{C} shatters a set $A \subset X$ if for every subset $E \subset A$ there exists $S \in \mathcal{C}$ such that $E = S \cap A$. The VC dimension of \mathcal{C} is the cardinality of the largest set that is shattered by \mathcal{C} .*

Through this section we deal with concept classes $\mathcal{C}_{k,n}$ such that concepts are represented by k real numbers, $w = (w_1, \dots, w_k)$, instances are represented by n real numbers, $x = (x_1, \dots, x_n)$, and the membership test to the family $\mathcal{C}_{k,n}$ is expressed by a formula $\Phi_{k,n}(w, x)$ taking as inputs the pair concept/instance (w, x) and returning the value 1 if " x belongs to the concept represented by w " and 0 otherwise

We can think of $\Phi_{k,n}$ as a function from \mathbb{R}^{k+n} to $\{0, 1\}$. So for each concept w , define:

$$C_w := \{x \in \mathbb{R}^n : \Phi_{k,n}(w, x) = 1\}. \quad (1)$$

The goal is to obtain an upper bound on the VC dimension of the collection of sets

$$\mathcal{C}_{k,n} = \{C_w : w \in \mathbb{R}^k\}. \quad (2)$$

Now assume that the formula $\Phi_{k,n}$ is a boolean combination of s atomic formulas, each of them having the following forms:

$$\tau_i(w, x) > 0 \quad (3)$$

or

$$\tau_i(w, x) = 0 \quad (4)$$

where $\{\tau_i(w, x)\}_{1 \leq i \leq s}$ are infinitely differentiable functions from \mathbb{R}^{k+n} to \mathbb{R} . Next, make the following assumptions about the functions τ_i . Let $\alpha_1, \dots, \alpha_v \in \mathbb{R}^n$. Form the sv functions $\tau_i(w, \alpha_j)$ from \mathbb{R}^k to \mathbb{R} . Choose $\Theta_1, \dots, \Theta_r$ among these, and define

$$\Theta : \mathbb{R}^k \rightarrow \mathbb{R}^r \quad (5)$$

as

$$\Theta(w) := (\Theta_1(w), \dots, \Theta_r(w)) \quad (6)$$

Assume there is a bound B independent of the α_i , r and $\epsilon_1, \dots, \epsilon_r$ such that if $\Theta^{-1}(\epsilon_1, \dots, \epsilon_r)$ is a $(k-r)$ -dimensional C^∞ -submanifold of \mathbb{R}^k then $\Theta^{-1}(\epsilon_1, \dots, \epsilon_r)$ has at most B connected components.

With the above setup, the following result is proved in [10].

Theorem 1. *The VC dimension V of a family of concepts $\mathcal{C}_{k,n}$ whose membership test can be expressed by a formula $\Phi_{k,n}$ satisfying the above conditions satisfies:*

$$V \leq 2 \log_2 B + 2k \log_2 (2es) \quad (7)$$

Next we state our main result concerning the VCD of a collection of subsets accepted by a family of slp's. We will say that a subset $C \subset \mathbb{R}^n$ is accepted by a slp Γ if the function computed by Γ , Φ_Γ , expresses the membership test to C . For slp's $\Gamma = (u_1, \dots, u_l)$ of length l accepting sets we assume that the output is the last instruction u_l and takes values in $\{0, 1\}$.

Theorem 2. *Let $T = \{t_1, \dots, t_n\}$ be a set of terminals and let $F = \{+, -, /, \text{sign}\}$ be a set of functionals where $\{+, -, *, /\}$ denotes the set of standard arithmetic operations and $\text{sign}(x)$ is a function that outputs 1 if its input $x \in \mathbb{R}$ satisfies $x \geq 0$ and outputs 0 otherwise. Let $\Gamma_{n,L}$ be the collection of slp's Γ over F and T using at most L non-scalar operations (i.e. operations in $\{*, /, \text{sign}\}$) and a free number of scalar operations (i.e. operations in $\{+, -\}$) whose output is obtained by applying the functional sign either to a previously computed result or to a terminal t_j , $1 \leq j \leq n$. Let $\mathcal{C}_{n,L}$ be the class of concepts defined by the subsets of \mathbb{R}^n accepted by some slp belonging to $\Gamma_{n,L}$. Then*

$$VC - \dim(\mathcal{C}_{n,L}) \leq 2(n+1)(n+L)L(2L + \log_2 L + 9) \quad (8)$$

Sketch of the proof. The first step in the proof consist of constructing of a universal slp Γ_U , over sets F_U an T_U , that parameterizes the elements of the family $\Gamma_{n,L}$. The definition of F_U and T_U depends only on the parameters n and L and will be clear after the construction. The key idea in the definition of Γ_U is the introduction of a set of parameters α, β taking values in $\{0, 1\}^k$, for a suitable natural number k , such that each specialization of this set of parameters yields a particular slp belonging to $\Gamma_{n,L}$ and conversely, each slp in $\Gamma_{n,L}$ can be obtained specializing the parameters α, β . For this purpose define $u_{-n+m} = t_m$ for $1 \leq m \leq n$. Note that any non-scalar assignment u_i , $1 \leq i \leq L$, in a slp Γ belonging to $\Gamma_{n,L}$ is a function of $t = (t_1, \dots, t_n)$ that can be parameterized as follows.

$$u_i = U_i(\alpha, \beta)(t) = \alpha_{-n}^i \left(\sum_{j=-n+1}^{i-1} \alpha_j^i u_j \right) * \left(\sum_{j=-n+1}^{i-1} \beta_j^i u_j \right) + \quad (9)$$

$$+ (1 - \alpha_{-n}^i) \left[\beta_{-n}^i \frac{\sum_{j=-n+1}^{i-1} \alpha_j^i u_j}{\sum_{j=-n+1}^{i-1} \beta_j^i u_j} + (1 - \beta_{-n}^i) \operatorname{sgn} \left(\sum_{j=-n+1}^{i-1} \beta_j^i u_j \right) \right], \quad (10)$$

for some suitable values $\alpha = (\alpha_j^i)$, $\beta = (\beta_j^i)$, with $\alpha_j^i, \beta_j^i \in \{0, 1\}$.

Let us consider the family of parametric slp's $\{\Gamma_{(\alpha, \beta)}\}$ where for each (α, β) the slp $\Gamma_{(\alpha, \beta)} := (U_1(\alpha, \beta), \dots, U_L(\alpha, \beta))$. Next replace the family of concepts $\mathcal{C}_{n,L}$ with the class of subsets of \mathbb{R}^n $\mathcal{C} := \{C_{(\alpha, \beta)}\}$ where for each (α, β) , the set $C_{(\alpha, \beta)}$ is given as follows.

$$C_{(\alpha, \beta)} := \{t = (t_1, \dots, t_n) \in \mathbb{R}^n : t \text{ is accepted by } \Gamma_{(\alpha, \beta)}\} \quad (11)$$

In the new class \mathcal{C} parameters α_j^i, β_j^i are allowed to take values in \mathbb{R} . Since $C_{n,L} \subset \mathcal{C}$ it is enough to bound the VC dimension of \mathcal{C} .

Claim A The number of parameters α, β is exactly

$$(n+1)(n+L)L \quad (12)$$

Claim B For each i , $1 \leq i \leq L$, the following holds:

(1) The function $U_i(\alpha, \beta)(t)$ is a piecewise rational function in the variables α, β, t of formal degree bounded by $3 \cdot 2^i - 2$.

(2) U_i is defined up to a set of zero measure and there is a partition of the domain of definition of U_i by subsets $(\Omega_j^i)_{1 \leq j \leq n_i}$ with $n_i \leq 2^i$ such that each Ω_j^i is defined by a conjunction of i rational inequalities of the form $p \geq 0$ or $p < 0$ with degree $\deg p \leq 3 \cdot 2^i - 2$. Moreover, the restriction of U_i to the set Ω_j^i , $U_i|_{\Omega_j^i}$, is a rational function of degree bounded by $3 \cdot 2^i - 2$.

(3) Condition $U_L(\alpha, \beta)(t) = 1$ can be expressed by a boolean formula of the following form:

$$\bigvee_{1 \leq i \leq 2^L} \wedge_{1 \leq j \leq L} p_{i,j} \epsilon_{i,j} 0; \quad (13)$$

where for each i, j , $p_{i,j}$ is a rational function in the variables α, β, t of degree bounded by $3.2^L - 2$ and $\epsilon_{i,j}$ is a sign condition in $\{\geq, <\}$.

Proof. Claim A follows by counting parameters. Claim B follows by induction on i .

In order to achieve the result in Equation 8, according Theorem 1 we have to estimate the number of connected components of a set defined by r equations of the form:

$$\Theta_1(t^1, \alpha, \beta) - \epsilon_1 = 0, \dots, \Theta_r(t^r, \alpha, \beta) - \epsilon_r = 0, \quad (14)$$

where $\epsilon_i \in \mathbb{R}$, $t^i \in \mathbb{R}^n$, and from Claim B, item 3, the $\Theta_i(t^i, \alpha, \beta)$ are polynomials in the variables (α, β) of degree bounded by $d = 3.2^L - 2$. Fortunately the solution to this problem is given by the following result by J. Milnor ([11]).

Lemma 1. *Assume that $\Theta_1(w), \dots, \Theta_r(w)$ are polynomials in k variables with degree at most d . Then, if $\epsilon = (\epsilon_1, \dots, \epsilon_r)$ is a regular value of $\Theta(w) := (\Theta_1(w), \dots, \Theta_r(w))$ as defined above, then*

$$B \leq (2d)^k \quad (15)$$

To conclude it is enough to apply Theorem 1 with $k = (n+1)(n+L)L$, $s = 6.L.2^L$ and $B \leq (2(3.2^L - 2))^k$.

3.2 Estimating the Average Error of slp's

We show how to apply the bound in Equation 8 to estimate the average error with respect to the unknown distribution from which the examples are drawn.

Let $\Gamma_{(\alpha, \beta)} := (U_1(\alpha, \beta), \dots, U_L(\alpha, \beta))$ be a family of parametric slp's as defined in the former Section . The average error of a classifier with parameters (α, β) is

$$\varepsilon(\alpha, \beta) = \int Q(t, \alpha, \beta; y) d\mu, \quad (16)$$

where Q measures the loss between the semantic function of $\Gamma_{(\alpha, \beta)}$ and the target concept, and μ is the distribution from which examples $\{(t_i, y_i)\}_{1 \leq i \leq m}$ are drawn to the GP machine. For classification problems, the error of misclassification is given taking $Q(t, \alpha, \beta; y) = |y - \Gamma_{(\alpha, \beta)}(t)|$. Similarly, for regression tasks one takes $Q(t, \alpha, \beta; y) = (y - \Gamma_{(\alpha, \beta)}(t))^2$. Since μ is usually unknown, one replaces theoretical error $\varepsilon(\alpha, \beta)$ by empirical error $\varepsilon_m(\alpha, \beta) = \frac{1}{m} \sum_{i=1}^m Q(t_i, \alpha, \beta; y_i)$. Now, the results by Vapnik state that the average error $\varepsilon(\alpha, \beta)$ can be estimated independently of the distribution of $\mu(t, y)$ due to the following formula.

$$\varepsilon(\alpha, \beta) \leq \varepsilon_m(\alpha, \beta) + \sqrt{\frac{h(\log(2m/h) + 1) - \log(\eta/4)}{m}}, \quad (17)$$

Here h must be substituted by the upper bound given in Equation 8. The constant η is the probability that the bound is violated.

4 SLP-Based Genetic Programming

For the construction of each individual $\Gamma \in SLP(F, T)$ of the initial population we adopt the process described in [2]. For each instruction $u_k \in \Gamma$ first an element $f \in F$ is random selected and then the function arguments of f are also randomly chosen in $T \cup \{u_1, \dots, u_{k-1}\}$, if $k > 1$ and in T if $k = 1$. We keep homogeneous populations of equal length slp's. Next, we describe the recombination operator.

Definition 3. (*slp-crossover*)(see [2]) Let $\Gamma = \{u_1, \dots, u_L\}$ and $\Gamma' = \{u'_1, \dots, u'_L\}$ be two slp's over F and T . First, a position k in Γ is randomly selected; $1 \leq k \leq L$. Let $S_{u_k} = \{u_{j_1}, \dots, u_{j_m}\}$ be the piece of the code of Γ related to the evaluation of u_k with the assumption that $j_1 < \dots < j_m$. Next randomly select a position t in Γ' with $m \leq t \leq L$ and modify Γ' by making the substitution of the subset of instructions $\{u'_{t-m+1}, \dots, u'_t\}$ in Γ' , by the instructions of Γ in S_{u_k} suitably renamed. The renaming function \mathcal{R} over S_{u_k} is defined as $\mathcal{R}(u_{j_i}) = u'_{t-m+i}$, for all $i \in \{1, \dots, m\}$. With this process we obtain the first offspring from Γ and Γ' . For the second offspring we symmetrically repeat this strategy, but now we begin by randomly selecting a position k' in Γ' .

Example 2. Let us consider the following slp's:

$$\Gamma \equiv \begin{cases} \mathbf{u}_1 := \mathbf{x} + \mathbf{y} \\ u_2 := u_1 * u_1 \\ \mathbf{u}_3 := \mathbf{u}_1 * \mathbf{x} \\ u_4 := u_3 + u_2 \\ u_5 := u_3 * u_2 \end{cases} \quad \Gamma' \equiv \begin{cases} \mathbf{u}_1 := \mathbf{x} * \mathbf{x} \\ \mathbf{u}_2 := \mathbf{u}_1 + \mathbf{y} \\ u_3 := u_1 + x \\ \mathbf{u}_4 := \mathbf{u}_2 * \mathbf{x} \\ u_5 := u_1 + u_4 \end{cases}$$

If $k = 3$ then $S_{u_3} = \{u_1, u_3\}$, and t must be selected in $\{2, \dots, 5\}$. Assumed that $t = 3$, the first offspring will be:

$$\Gamma_1 \equiv \begin{cases} u_1 := x * x \\ \mathbf{u}_2 := \mathbf{x} + \mathbf{y} \\ \mathbf{u}_3 := \mathbf{u}_2 * \mathbf{x} \\ u_4 := u_2 * x \\ u_5 := u_1 + u_4 \end{cases}$$

For the second offspring, if the selected position in Γ' is $k' = 4$, then $S_{u_4} = \{u_1, u_2, u_4\}$. Now if $t = 5$, the offspring will be:

$$\Gamma_2 \equiv \begin{cases} u_1 := x + y \\ u_2 := u_1 * u_1 \\ \mathbf{u}_3 := \mathbf{x} * \mathbf{x} \\ \mathbf{u}_4 := \mathbf{u}_3 + \mathbf{y} \\ \mathbf{u}_5 := \mathbf{u}_4 * \mathbf{x} \end{cases}$$

Next we describe mutation. The first step when mutation is applied to a slp Γ consists of selecting an instruction $u_i \in \Gamma$ at random. Then a new random selection is made within the arguments of the function $f \in F$ that constitutes the instruction u_i . The final step is the substitution of the selected argument for another one in $T \cup \{u_1, \dots, u_{i-1}\}$ randomly chosen.

4.1 Fitness based on Structural Risk Minimization

Under Structural Risk Minimization a set of possible models C forms a nested structure $C_1 \subset C_2 \subset \dots \subset C_L \subset \dots \subset C$. Here C_L represents the set of models of complexity L , where L is some suitable parameter depending on the problem. We require that VC-dimension V_L of model C_L is an increasing function of L . In our particular case C is the class of slp's having length bounded by some constant l with set of terminals $T =: \{t_1, \dots, t_n\}$ and set of functionals $F = \{+, -, *, /, sign\}$ and C_L is the class of slp's in C which use at most L non-scalar operations. In this situation one chooses the model that minimizes the right side of Equation 17.

For practical use of Equation 17 we adopt the following formula with appropriately chosen practical values of theoretical constants (see [12] for the derivation of this formula).

$$\varepsilon_m(h) \left(1 - \sqrt{p(h) - p(h) \ln p(h) + \frac{\ln m}{2m}} \right)^{-1}, \quad (18)$$

where $\varepsilon_m(h)$ means empirical risk, $p(h) = \frac{h}{m}$ and h is a function such that for each SLP Γ of length bounded by l with set of terminals T and set of functionals F the following holds: $h(\Gamma) = V_L$ where $L = \min\{k : \Gamma \in C_k\}$, that is, h is the VC-dimension of the class of models using as many non-scalar operations as Γ .

4.2 Experimentation

We consider instances of Symbolic Regression for our experimentation. The symbolic regression problem has been approached by Genetic Programming in several contexts. Usually, in this paradigm a population of tree-like structures encoding expressions, is evolved. We adopt slp's as the structures that evolve within the process. We will keep homogeneous populations of equal length individuals along the process.

Experiment 1. We compare four crossover methods: uniform, one-point crossover, 2-point crossover and slp-crossover as defined in Definition 3. For this purpose we have performed 200 executions of the algorithm for each instance and crossover operator. We have run our implemented algorithm based on GP with straight line programs on the following three target functions:

$$f_1(x) = x^4 + x^3 + x^2 + x; \quad f_2(x) = \cos(2x); \quad f_3(x) = 2.718 x^2 + 3.1416 x$$

These functions are also proposed in the book by John Koza, as illustration examples of the tree-based GP approach ([13]). The sample set contains only 20 points for the three target functions. The following values for the control parameters are the same for all tested functions: population size $M = 500$; number of generations $G = 50$; probability of crossover $p_c = 0,9$; probability of mutation $p_m = 0,01$; length of the homogeneous population of slp's $L = 20$. The set of functions is $F = \{+, -, *, //\}$ for f_1 and f_3 and $F = \{+, -, *, //, sin\}$ for

f_2 . In the above sets “//” indicates the protected division i.e. $x//y$ returns x/y if $y \neq 0$ and 1 otherwise. The basic set of terminals is $T = \{x, 1\}$. We include in T a constant c_0 for the target function f_3 . The constant c_0 takes random values in the interval range $[-1, 1]$ and is fixed before each execution. In table 2 we show the corresponding success rates for each crossover method and target function. The success rate (SR) is defined as the ratio of the successful runs with respect to the total number of runs. In our case one run will be considered successful if an individual with a empirical risk lower than 0.2 is found. The umbral of 0.2 is obtained by multiplying the number of test points, 20, by 0.01, that is the maximum error allowed to consider a point as a success point. We can see that uniform crossover and slp-crossover are the best recombination operators for the studied target functions. On the other hand the one point crossover is the worst of the studied recombination operators. Note that uniform crossover and two point crossover are not defined for the tree structure.

Experiment 2. This experiment was designed to test the effectiveness of VC regularization of the slp model using the fitness based in Structural Risk Minimization as given by formula in Equation 18 (VC-fitness) instead of fitness based on Empirical Risk Minimization (ERM-fitness). We describe first experimental comparison using the methodology and data sets from [12]. First we describe the experimental setup.

Target functions. The following target functions where used:

Discontinuous piecewise polynomial function defined as follows:

$$g_1(x) = 4(x^2(3 - 4x)), x \in [0, 0.5],$$

$$g_1(x) = (4/3)x(4x^2 - 10x + 7) - 3/2, x \in (0.5, 0.75],$$

$$g_1(x) = (16/3)x(x - 1)^2, x \in (0.75, 1].$$

Sine-square function: $g_2(x) = \sin^2(2\pi x)$, $x \in [0, 1]$

Two-dimensional *sinc* function: $g_3(x) = \frac{\sin\sqrt{x_1^2+x_2^2}}{x_1^2+x_2^2}$, $x_1, x_2 \in [-5, 5]$

Polynomial function: $g_4(x) = x^4 + x^3 + x^2 + x$, $x \in [0, 1]$

Estimators used. We use slp’s of fixed length l with functionals in $\{+, -, *, /\}$ for target functions g_2, g_3, g_4 . For target function g_1 we add the *sign* operator. In the experiments we set $l = 6, 10, 20$ and 40 . The model complexity is measured by the number of non-scalar operations in the considered slp, that is, the number of instructions which do not contain $\{+, -\}$ -operators. This is a measure of the non-linearity of the regressor as suggested by Theorem 2. Each estimator has being obtained running a GP algorithm with the following values for the control parameters: population size $M = 500$ or $M = 100$; number of generations

$G = 50$; probability of crossover $p_c = 0,9$; probability of mutation $p_m = 0,01$. In all trials, slp-crossover, as described in Definition 3, was used.

Experimentation procedure. A training set of fixed size n is generated. The x -values follows from uniform distribution in the input domain. The prediction risk $\varepsilon_{n_{test}}$ of the model chosen by the GP algorithm based on SLP evolution is estimated by the mean square error (MSE) between the model (estimated from training data) and the true values of target function $g(x)$ for independently generated test inputs $(x_i, \hat{y}_i)_{1 \leq i \leq n_{test}}$, i. e.:

$$\varepsilon_{n_{test}} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (g(x_i) - \hat{y}_i)^2 \quad (19)$$

The above experimental procedure is repeated 100 times using 100 different random realizations of n training samples (from the same statistical distribution). Experiments were performed using a small training sample ($n = 20$) and a large test set ($n_{test} = 200$). Table 1 shows comparison results for fitness based on ERM (ERM-fitness) and fitness based on VC-regularization (VC-fitness). Experiments for each target function $g = g_i, 1 \leq i \leq 4$ are divided into four groups corresponding to different values of the length of the evolved slp's ($l = 6, l = 10, l = 20, l = 40$). For each length two possible population sizes are considered (100 and 500 individuals). For each population size two fitness are considered: ERM-fitness and VC-fitness. The values in the fitness rows represent the estimated prediction error of the selected model. The values in the comparative rows represent the ratio between the prediction risk for the regressor obtained using the ERM-fitness and the corresponding value for the regressor obtained using the VC-fitness. Accordingly, the values in the comparative rows that are bigger than or equal to 1 represent a better performance of VC-fitness. If we consider an experiment successful when the comparative value is ≥ 1 , then the success rate is greater than 70%. If we consider an experiment strictly successful when the comparative value is > 1 , then the strict success rate is greater than 30%.

5 Conclusions and Future Research

We have calculated a sharp bound for the VC dimension of the GP genotype defined by computer programs using straight line code. We have used this bound to perform VC-based model selection under the GP paradigm showing that this model selection method consistently outperforms LGP algorithms based on empirical risk minimization. A next step in our research is to compare VC-regularization of slp's with other regularization methods based on asymptotical analysis like Akaike information criterion (AIC) or Bayesian information criterion (BIC). A second goal in our research on SLP-based GP is to study the experimental behavior of the straight line program computation model under Vapnik-Chervonenkis regularization but without assuming previous knowledge

	population size 100				population size 500			
	g_1	g_2	g_3	g_4	g_1	g_2	g_3	g_4
	length 6							
ERM-fitness	0.1559	0.1502	0.0456	0.1054	0.0670	0.1501	0.0456	0.1054
VC-fitness	0.0396	0.1502	0.0456	0.1019	0.0706	0.1443	0.0456	0.1019
Comparative	3.9368	1	1	1.0343	0.9490	1.0401	1	1.0343
	length 10							
ERM-fitness	0.1559	0.2068	0.0456	0.1054	0.0396	0.1502	0.0456	0.0377
VC-fitness	0.1559	0.1434	0.0456	0.1791	0.0396	0.1502	0.0456	0.1054
Comparative	1	1.4421	1	0.5884	1	1	1	0.3576
	length 20							
ERM-fitness	0.1566	0.1324	0.0456	0.4982	0.0396	0.1396	0.0456	0.0870
VC-fitness	0.1559	0.2068	0.0456	0.1852	0.0661	0.1827	0.0456	0.0633
Comparative	1.0004	0.6402	1	2.6900	0.5990	0.7640	1	1.3744
	length 40							
ERM-fitness	0.1029	0.1439	0.0456	0.1357	0.0745	0.1502	0.0456	0.2287
VC-fitness	0.1559	0.2068	0.0456	0.5709	0.0326	0.1502	0.0456	0.0805
Comparative	0.6600	0.6958	1	0.2376	2.2852	1	1	2.8409

Table 1. Prediction Risk: VC-fitness vs. ERM-fitness

Function	1-point	2-point	uniform	slp
f_1	100	100	100	100
f_2	30	35	53	53
f_3	42	71	91	93

Table 2. SR over 200 independent runs for the crossover operators

of the length of the structure. This investigation is crucial in practical applications for which the GP machine must be able to learn not only the shape but also the length of the evolved structures. To this end new recombination operators must be designed since the crossover procedure employed in this paper only applies to populations having fixed length chromosomes.

6 Acknowledgements

This work was partially supported by Spanish Grants TIN2007-67466-C02-02, MTM2007-62799 and FPU program.

References

1. Giusti, M., Heintz, J., Morais, J.E., Pardo, L.M.: Straight line programs in Geometric elimination Theory. *Journal of Pure and Applied Algebra* **124** (1997) 121–146
2. Alonso, C.L., Montaña, J.L., Puente, J.: Straight line programs: a new Linear Genetic Programming Approach. In: Proc. 20th IEEE International Conference on Tools with Artificial Intelligence, ICTAI (2008) 517–524
3. Brameier, M., Banzhaf, W.: *Linear Genetic Programming*. Springer (2007)
4. Aldaz, M., Heintz, J., Matera, G., Montaña, J.L., Pardo, L.: Time-space tradeoffs in algebraic complexity theory. *Journal of Complexity* **16** (1998) 2–49
5. Teytaud, O., Gelly, S., Bredeche, N., Schoenauer, M.A.: Statistical Learning Theory Approach of Bloat. In: Proceedings of the 2005 conference on Genetic and Evolutionary Computation. (2005) 1784–1785
6. Vapnik, V., Chervonenkis, A.: Ordered risk minimization. *Automation and Remote Control* **34** (1974) 1226–1235
7. Vapnik, V.: *Statistical learning theory*. John Willey & Sons (1998)
8. Vapnik, V., Chervonenkis, A.: On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications* **16** (1971) 264–280
9. Lugosi, G.: Pattern classification and learning theory. In: *Principles of Nonparametric Learning*. Springer (2002) 5–62
10. Karpinski, M., Macintyre, A.: Polynomial bounds for VC dimension of sigmoidal and general Pffafian neural networks. *J. Comp. Sys. Sci.* **54** (1997) 169–176
11. Milnor, J.: On the Betti Numbers of Real Varieties. *Proc. Amer. Math. Soc.* **15** (1964) 275–280
12. Cherkassky, V., Yunkian, M.: Comparison of Model Selection for Regression. *Neural Computation* **15** (2003) 1691–1714
13. Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press (1992)