

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



José L. Montaña, Cruz E. Borges, César L. Alonso, José L. Crespo.

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

Centro de Inteligencia Artificial, Universidad de Oviedo.

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

1. Straight Line Programs: Basic Concepts and Properties

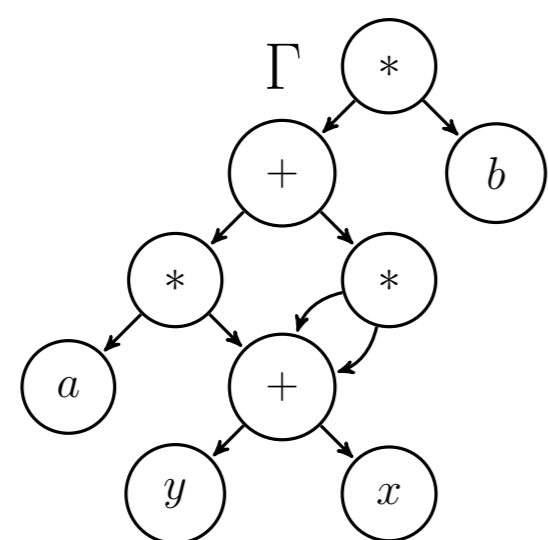
Definition 1.1 (SLP).

A slp is a finite sequence of computational instructions where it is allowed to reuse previous computations.

Figure 1: Sequential slp Γ

$$\Gamma \equiv \begin{cases} u_1 := x + y \\ u_2 := u_1 * u_1 \\ u_3 := u_1 * a \\ u_4 := u_3 + u_2 \\ u_5 := u_4 * b \end{cases}$$

Figure 2: Graph slp Γ



Note that Γ represent the polynomial $(ba + x + y)(x + y)$.

- Straight Line Programs (slp) are commonly used for solving problems of algebraic and geometric flavour. An extensive study of the use of slp's in this context can be found in [1, 2].
- The set of instructions F as well as the set of terminals T can be chosen arbitrary. In the previous example we use: $F := \{+, -, *, /\}$ and $T := \{x, y, a, b\}$.
- Note that, as we can reuse previous computations, the set of terminals of the computational node u_i are $T \cup \{u_1, \dots, u_{i-1}\}$.

As we will use this data structure in a Linear Genetic Programming problem we need to define a slp-crossover operator and a slp-mutation operator.

Algorithm 1.2 (slp-crossover).

Input $\Gamma_1 := \{u_1, \dots, u_n\}$ $\Gamma_2 := \{u'_1, \dots, u'_m\}$
Output Γ'_1, Γ'_2

1. Take a random number k between 0 and n .
2. Let $S_k^1 = \{u_{j_1}, \dots, u_{j_s}\}$ the computational nodes needed to evaluated the node u_k that belong to Γ_1 . Note that $j_1 < \dots < j_m$.
3. Take another random number l between s and m .
4. Replace $\{u'_{l-s+1}, \dots, u'_l\}$ in Γ_2 with S_k^1 suitably renamed.
5. Repeat, swapping Γ_1 with Γ_2 .

Figure 3: Parents ($k_1 = 3$ and $k_2 = 3$)

$$\Gamma_1 \equiv \begin{cases} u_1 := x + y \\ u_2 := u_1 * u_1 \\ u_3 := u_1 * a \\ u_4 := u_3 + u_2 \\ u_5 := u_4 * b \end{cases} \quad \Gamma_2 \equiv \begin{cases} v_1 := x * x \\ v_2 := v_1 + y \\ v_3 := v_1 + x \\ v_4 := v_2 * x \\ v_5 := v_1 + v_4 \end{cases}$$

Figure 4: Children ($l_1 = 4$ and $l_2 = 3$)

$$\Gamma'_1 \equiv \begin{cases} u'_1 := x * x \\ u'_2 := u'_1 + y \\ u'_3 := x + y \\ u'_4 := u'_3 * a \\ u'_5 := u'_1 + u'_4 \end{cases} \quad \Gamma'_2 \equiv \begin{cases} v'_1 := x * x \\ v'_2 := v'_1 + y \\ v'_3 := v'_2 * x \\ v'_4 := v'_3 + v'_2 \\ v'_5 := v'_4 * b \end{cases}$$

Algorithm 1.3 (slp-mutation).

Input $\Gamma_1 := \{u_1, \dots, u_n\}$
Output Γ'_1

1. Take a random number k between 0 and n .
2. Take an element v in $T \cup \{u_1, \dots, u_{k-1}\}$ at random.
3. Replace an argument of u_k (chosen at random) by v .

2. Vapnik-Chervonenkis Dimension of Families of slp's

- GP has been applied to a range of complex learning problems including that of classification and symbolic regression.
- Both tasks can be thought of as a supervised learning problem (see [6]) where the hypothesis class \mathcal{C} is the search space described by the genotypes of the evolving structures.
- The works by Vapnik and Chervonenkis ([7, 5]) provided bounds (VC dimension) relating the performance of a learning machine.
- The VC dimension depends on the class of classifiers, hence it makes sense the analysis of the VC dimension of slp's.

Theorem 2.1 ([4]).

The VC dimension V of a family of concepts \mathcal{C} whose membership test can be expressed by a formula Φ satisfies:

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

where B, e, s are constants that depends of the formula Φ and k is a constant that depends of the family of concepts.

Theorem 2.2 (Main Theorem).

Let $T = \{t_1, \dots, t_n\}$ be a set of terminals and let $F = \{+, -, *, /\}$ be a set of functionals where $\{+, -, *, /\}$ denotes the set of standard arithmetic operations and $\text{sgn}(x)$ outputs 1 if $x \geq 0$ and 0 otherwise.

Let $\Gamma_{n,L}$ be the collection of slp's Γ over F and T using at most L non-scalar operations and a free number of scalar operations.

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, the VC dimension of $\Gamma_{n,L}$ satisfied:

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

3. Estimating the Average Error of slp's

- The theoretical average error ε of a slp classifier Γ in the regression problem is

$$\varepsilon := \int (y - \Gamma(t))^2 d\mu,$$

where μ is the distribution from which examples $\{(t_i, y_i)\}_{1 \leq i \leq m}$ are generated.

- Since μ is generally unknown, we replaces ε with the empirical error

$$\varepsilon_m = \sum_{i=1}^n (y_i - \Gamma(t_i))^2.$$

- The result by Vapnik state that:

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

where h is the upper bound given in theorem 2.2 and η is the probability that the bound is violated.

- For practical use of the previous equation we use the following formula (see [3]):

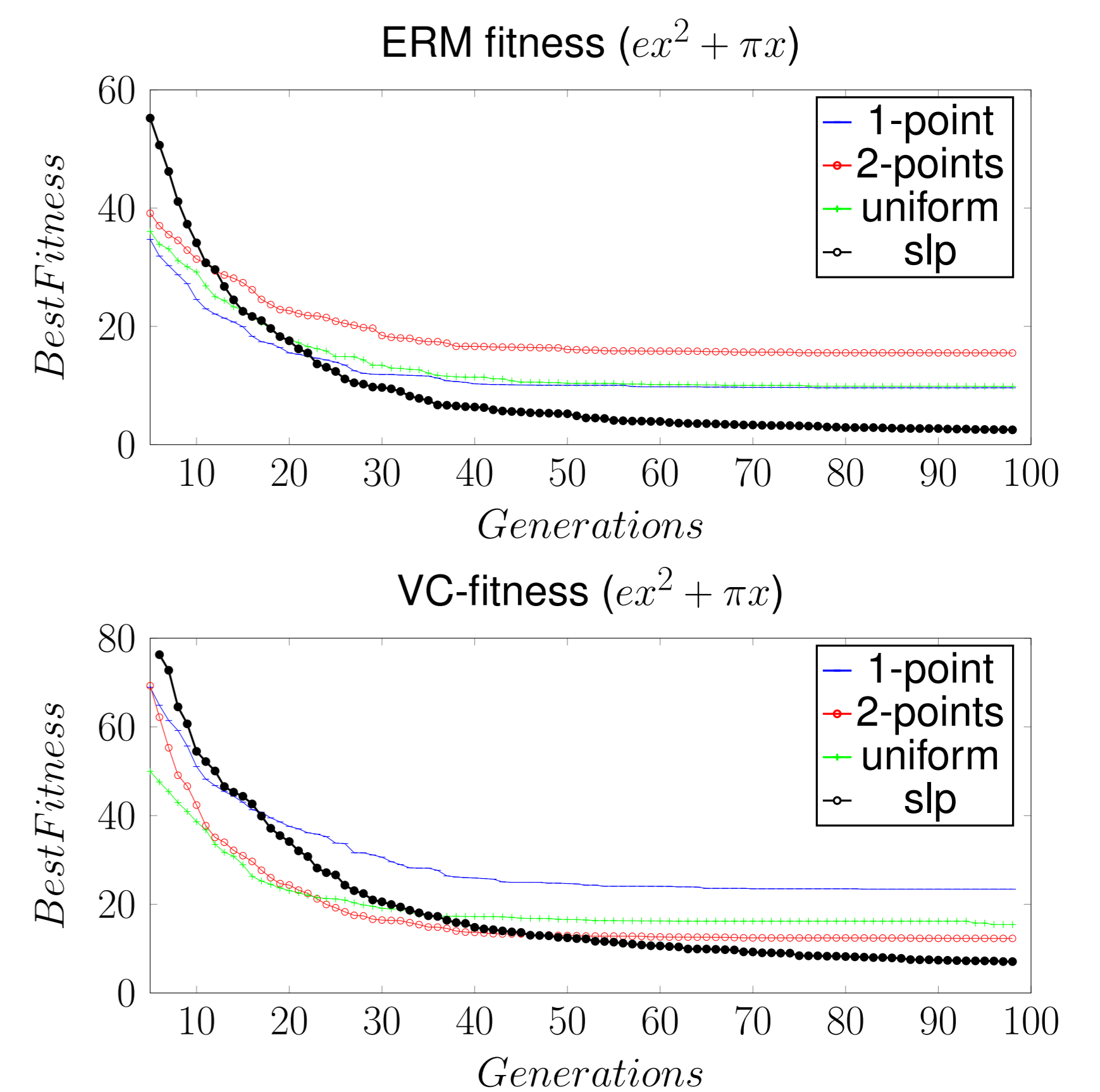
$$\varepsilon_m(h) \left(1 - \sqrt{p(h) - p(h) \ln(p(h)) + \frac{\ln(m)}{2m}} \right)^{-1},$$

where $\varepsilon_m(h)$ means empirical risk, $p(h) = \frac{h}{m}$ and h is the VC-dimension of the class of models using as many non-scalar operations as Γ .

4. Experimental Results

- The symbolic regression problem has been approached by Genetic Programming using a population of tree-like structures encoding expressions that evolved within the process. In contrast, we adopt slp's as the structures that evolve.
- The following experiments were done:

Experiment 1: Comparison between the classical crossover methods 1-point, 2points and uniform with the slp crossover previously defined.



Experiment 2: Comparison between the fitness based on Structural Risk Minimization using the formulas previously defined and the classical approach based on Empirical Risk Minimization.

Figure 5: Prediction Risk: VC vs. ERM

Problem	VC	ERM	Comparative
f1	5.40E-28	6.90E-28	1.28E+00
f2	1.62E+00	7.27E-01	4.48E-01
f3	2.73E-02	2.26E-02	8.28E-01
f4	1.11E-32	9.80E-06	8.86E+26
f5	3.39E-02	3.26E-02	9.62E-01
g1	3.31E-02	2.46E-02	7.43E-01
g2	1.27E-01	1.97E+01	1.56E+02
g3	1.74E+01	6.47E+01	3.71E+00
F	1.23E-05	1.86E-03	1.51E+02
G	3.22E+00	4.46E+00	1.38E+00
K	7.46E+01	3.74E+01	5.01E-01

5. Future Research

- Compare the VC-regularization of slp with other regularization method based on asymptotical analysis like AIC or BIC.
- Study the experimental behaviour of slp computation model under VC-regularization but without assuming previous knowledge of the length of the structure.

6. Acknowledgements

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

References

- [1] M. Aldaz, J. Heintz, G. Matera, J. L. Montaña, and L.M. Pardo. Time-space tradeoffs in algebraic complexity theory. *Journal of Complexity*, 16:2–49, 1998.
- [2] C. L. Alonso, J. L. Montaña, and J. Puente. Straight line programs: a new Linear Genetic Programming Approach. In *Proc. 20th IEEE International Conference on Tools with Artificial Intelligence*, pages 517–524. ICTAI, 2008.
- [3] V. Cherkassky and M. Yunkian. Comparison of Model Selection for Regression. *Neural Computation*, 15:1691–1714, 2003.
- [4] M. Karpinski and A. Macintyre. Polynomial bounds for VC dimension of sigmoidal and general Pffafian neural networks. *J. Comp. Sys. Sci.*, 54:169–176, 1997.
- [5] G. Lugosi. *Principles of Nonparametric Learning*, chapter Pattern classification and learning theory, pages 5–62. Springer, 2002.
- [6] O. Teytaud, S. Gelly, N. Bredeche, and M. A. Schoenauer. Statistical Learning Theory Approach of Bloat. In *Proceedings of the 2005 conference on Genetic and Evolutionary Computation*, pages 1784–1785, 2005.
- [7] V. Vapnik. *Statistical learning theory*. John Wiley & Sons, 1998.