Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 103 (2017) 597 – 604

Procedia Computer Science

XIIth International Symposium «Intelligent Systems», INTELS'16, 5-7 October 2016, Moscow, Russia

# Method of binary analytic programming to look for optimal mathematical expression

A.I. Diveev[a,b]*, N.B. Konyrbaev[b], E.A. Sofronova[b]

*[a]Federal Research Center "Computer Science and Control" Russian Academy of Sciences, 44, Vavilova str., Moscow 119333, Russia*
*[b]RUDN University, 6, Miklukho-Maklaya str., Moscow 117198, Russia*

---

## Abstract

In the known methods of symbolical regression by search of the solution with the help of a genetic algorithm, there is a problem of crossover. Genetic programming performs a crossover only in certain points. Grammatical evolution often corrects a code after a crossover. Other methods of symbolical regression use excess elements in a code for elimination of this shortcoming. The work presents a new method of symbolic regression on base of binary computing trees. The method has no problems with a crossover. Method use a coding in the form of a set of integer numbers like analytic programming. The work describes the new method and some examples of codding for mathematical expressions.

*Keywords:* symbolic regression; genetic programming; analytic programming; genetic algorithm

---

## 1. Introduction

Methods of symbolic regressions can find structure of mathematical expression. Many important tasks require finding of the optimal mathematical expression, including a task of handling of experimental data for forecasting and search of common factors, identification of mathematical model, and synthesis of control.

---

* Corresponding author.
*E-mail address:* aidiveev@mail.ru, diveev_ai@pfur.ru

doi:10.1016/j.procs.2017.01.073

All methods of symbolic regression code mathematical expressions and carry out search of the optimal solution on space of codes by means of an evolutionary algorithm. The most suitable for search of the solution in methods of symbolical regression is a genetic algorithm. It does not use arithmetic operations for modification of possible solutions. The main operations of a genetic algorithm are a crossover and a mutation. Crossover of genetic algorithm depends on kind of code. The genetic programming[1] makes crossover in certain points for correct codes in order that the new code after crossing was correct and corresponded to some possible solution. Grammatical evolution[2] and analytic programming[3] carry out crossing in any points like a genetic algorithm for numerical tasks. These methods after crossover sometimes receive an incorrect code. Correction of a code demands additional expenses of time. Variational methods of symbolical regression check conditions of performance of variations as in the network operator method[4] or also correct a code after a crossover, as in variational methods of genetic[5] and analytic[6] programming. Cartesian genetic programming[7] and the method of parse matrix[8] always have correct a crossover. These methods of symbolical regression use excess elements in a code. Both methods consider coding of functions with the maximum number of arguments. If it is necessary to code function with smaller number of arguments, then excess codes for arguments at calculation aren't considered. All codes have to have the same length independent on the coded mathematical expression that why some codes of functions aren't considered at calculation.

The work presents the new method of symbolical regression, a method of binary analytical programming. The method codes only mathematical expressions in the form of superposition of functions with one or two arguments. The method for performance of correct crossover also includes excess elements of a code, the identity function with one argument and unit elements for functions with two arguments. These elements like zeros in positional notation of numbers also allow construct a regular code of mathematical expression in which codes of functions with different number of arguments alternate.

## 2. Binary analytic programming

To build a code of binary analytic programming we use the following base sets:

- a set of arguments of mathematical expression

$$\mathrm{F}_0 = (q_1, \cdots, q_P, x_1, \cdots, x_N) \tag{1}$$

- a set of functions with one argument

$$\mathrm{F}_1 = (f_{1,1}(z) = z, f_{1,2}(z), \cdots, f_{1,R}(z)) \tag{2}$$

- a set of functions with two arguments

$$\mathrm{F}_2 = (f_{2,1}(z_1, z_2), \cdots, f_{2,S}(z_1, z_2)) \tag{3}$$

- a set of unit elements for functions with two arguments

$$\mathrm{E}_2 = (e_1, \cdots, e_M) \tag{4}$$

A set of function with one argument must include the identity

$$f_{1,1}(z) = z \tag{5}$$

Every function with two arguments of a set (3), $\forall f_{2,i}(z_1, z_2) \in \mathrm{F}_2$, has unit element of a set (4), $\exists e_j \in \mathrm{E}_2$

$$f_{2,i}(e_j, z_2) = z_2,\ f_{2,i}(z_1, e_j) = z_1,\ i \in \{1, \cdots, S\},\ j \in \{1, \cdots, M\}. \tag{6}$$