UDC 519.6 *doi: 10.30970/vam.2023.31.11848*

GENETIC PROGRAMMING FOR THE TWO-DIMENSIONAL BOUNDARY RECONSTRUCTION PROBLEM

I. Borachok, A. Marchenko

Ivan Franko National University of Lviv,
1, Universytetska str., 79000, Lviv, Ukraine
e-mails: ihor.borachok@lnu.edu.ua, arkadii.marchenko@lnu.edu.ua

We consider the application of the genetic programming for solving a non-linear ill-posed problem, namely for the reconstruction of the inner boundary of the annular planar domain from the known measurement data of the harmonic function. The unknown boundary is represented in a tree form, for which the fitness function is defined, as a selection criteria for transition to the next generation. The fitness function is computed by solving a mixed boundary value problem. In turn, the boundary-value problem is solved numerically by the method of boundary integral equations using single-layer potentials and the Nyström method for full discretization. To generate new individuals we use various tree-specific genetic operators such as mutation, crossover and tree generation algorithms. Different algorithm configurations and values of the genetic parameters are included. The proposed approach is numerically validated and the results of numerical experiments for exact and noisy input data are given.

Key words: genetic programming, boundary reconstruction problem, Laplace equation, boundary integral equations method.

1. Introduction

The genetic algorithm was primary introduced by Holland [12] as a search method for the complex problems when there is no suitable numerical method. It is a stochastic optimization algorithm, based on biological evolution principals. The algorithm manages a population of chromosomes (each chromosome represents a possible solution to the problem) and by applying the genetic operators to the chromosomes and using an appropriate selection technique, the next population is built. In general, in the genetic algorithm, each chromosome is represented by a bit string. If some data structure is used for chromosome representation, the method is called an evolutionary algorithm. When this data structure is a tree, the method is called genetic programming (meaning that the chromosome can change or program its own structure). For more details in this field, we refer to fundamental works [11, 18].

The real-coded genetic algorithm [18] has been successfully applied to the inverse ill-posed problems. The selection of the regularization parameter [20] or the initial approximation [7, 8] are the main areas of the genetic algorithm application. Alternatively, the algorithm can be used as a standalone iterative procedure in addition to some regularization technique, see for example [17, 19].

However, the application of genetic programming to the ill-posed problems is less studied. Let us consider the application of this technique for the reconstruction of the inner boundary of the annular planar domain. The problem widely used in impedance tomography, non-destructive testing and electrostatics [8, 14]. Due to its importance, there are already several works based, for example, on boundary integral equations method

and iterative regularization, see, [9,10,13,14], which makes it possible to obtain a sufficiently good reconstruction of the boundary. Usually, during iterative procedures, there is a need to solve additional boundary value problems, for example, calculation of the Fréchet derivative, but in the approach of genetic algorithms it is not necessary. Also, the real-coded genetic algorithm is used as a procedure to construct an initial guess of the boundary, see [7,8], followed by application of classical methods. In this work we use the genetic programming as a standalone iterative regularization procedure. In order to get a more accurate solution at the end, the classic algorithm can be applied, when the solution according to our method selected as the initial guess.

We focus only on the two-dimensional double connected domains and consider the Laplace equation as a governing equation, but the proposed method can be extended to the other domains, higher dimensions or equations, as it has been successfully done for the classical numerical methods [3,4,6].

Let us formulate the studied problem more precisely. We consider the doubly connected domain D, bounded with the two simple (without self intersections) closed curves Γ_1 and Γ_2, that belongs to C^2 class. Moreover, the curve Γ_1 lies in the interior of Γ_2. An example of the domain configuration is given in the fig. 1.

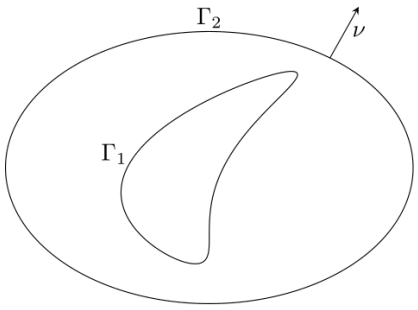

Fig. 1. Example of the domain D

Firstly, let's consider the direct problem. Let $u \in C^2(D) \cap C(\bar{D})$ be a solution of the Laplace equation

$$\Delta u = 0 \ \text{ in } D, \tag{1}$$

and function u satisfies the corresponding boundary Dirichlet and Neumann conditions

$$u = 0 \ \text{ on } \Gamma_1 \quad \text{and} \quad \frac{\partial u}{\partial \nu} = g \ \text{ on } \Gamma_2, \tag{2}$$

where $g \in C^1(\Gamma_2)$ known function. Here by the ν we denote the outward unit normal to Γ_ℓ, $\ell = 1, 2$. Existence and uniqueness of the solution of the mixed boundary value problem is well established, see, for example, [13]. Note that we can formulate the problem in term of weak solution, when function g belongs to Sobolev trace space, for more details see [16].

Let's consider the inverse problem that consists on reconstruction of the shape of the interior curve Γ_1 from the measurement (2) with $g \neq 0$ and the additional Dirichlet data on the boundary Γ_2

$$u = f \ \text{ on } \Gamma_2, \tag{3}$$

where $f \in C(\Gamma_2)$ known function and $f \neq 0$. Well known that there exists only one representation of Γ_1, for more details we refer to [3, 13]. The solution of the formulated problem (1)-(3) doesn't depend continuously on the input data, thus we have the case of non-linear ill-posed problem.

The unknown boundary Γ_1 is represented in a tree form (for simplicity we assume that it belongs to 2π periodic starlike curves). For each tree-chromosome we define the fitness or cost function that is used as a selection criteria for transition to the next population. For the computing of the fitness function we are going to numerically solve the well-posed boundary value problems. To do this, we represent the solution in a form of a single-layer potentials with unknown densities. Matching the given data, a system of boundary integral equations is obtained, which, in turn solved by the Nyström method. The method of boundary integral equations is widely used for solution of well-posed or ill-posed boundary value problems, see [5, 13].

For the outline of the work, in the section 2 we introduce the genetic programming as the main iterative algorithm to obtaining a solution of the reconstruction problem. In the section 3 we use the boundary integral equations approach and discretization algorithm to formulate the error function that is used as a fitness function of the chromosome. The configuration of the algorithm and results of the some numerical experiments are given in the section 4.

2. Genetic programming for the minimization problem

Genetic programming is a class of the genetic algorithms when the chromosome is represented in the tree form. In other words, it's a stochastic optimization technique that models the process of the evolution and manages the population of the tree-chromosomes. Following [11, 18], the algorithm starts with a randomly generated population of tree-chromosomes. Next tree-specific genetic operators are applied to candidate solutions to create a new chromosome. Moreover, for the each chromosome, fitness (sometimes called cost) function is defined. Based on fitness function, for each population the fittest solution transits to the next population, and bad candidates die.

Let's apply the genetic programming to the problem (1)-(3). In order, let's reformulate the problem as a minimization problem. Assume, that the unknown boundary Γ_1 belongs to the starlike curves. Thus, Γ_1 has following representation

$$\Gamma_1 = \{x_1(t) = (x_{11}(t), x_{12}(t)) = r(t)(\cos t, \sin t),\ t \in [0, 2\pi]\}, \tag{4}$$

where $r : \mathbb{R} \to (0, \infty)$ is a 2π periodic unknown function, representing the radial distance from the origin.

The problem (1)-(3) consists of the finding of the unknown function r, that minimize:

$$J(r) = \|u - f\|_{L_2(\Gamma_2)} \to \min, \tag{5}$$

where u is a solution of the well-posed mixed boundary value problem (1)-(2), for Γ_1 given by (4). The function J is a fitness function of the tree-chromosome. In order, to calculate the J we need to solve a separate problem. This could be done by the genetic algorithm as well, but to obtain a more precise fitness, we apply the classical method of the boundary integral equations to find the function u and compute J. The algorithm for the numerical solution of the (1)-(2) is shown in the next section.

Let's describe the genetic programming in more detail. Let F be a set of predefined functions, T – set of the predefined terminals with $T_i \subset T$ be a set of the values and

$T_v \subset T$ – set of the variables. Let's note by the v a tree-chromosome which represents a possible candidate $\tilde{r}$ for the function r in the tree form. The fitness function of the $v = v(\tilde{r})$ is obtained from the (5), where for the direct problem (1)-(2) the function $\tilde{r}$ is used.

There are several restriction to curves $\tilde{r}$, such as no self intersection, being inside outer curve Γ_2 and no sharp corners. To detect sharp corners we use curvature that calculated as follow

$$\kappa = \frac{|x'_{11}x''_{12} - x'_{12}x''_{11}|}{(x'^{\,2}_{11} + x'^{\,2}_{12})^{\frac{3}{2}}}\|\tilde{r}\|.$$

There $\|\tilde{r}\|$ is used for normalizing curvature coefficient for different curve scales.

To create new chromosomes we use genetic operators: mutation and crossover, see [18].

Mutation operators

Subtree mutation chooses random node (except root) and replace it with randomly generated sub-tree, an example is given in the fig. 2.

Size-fair sub-tree mutation same as previous one, except it generates sub-tree to be replaced in such a way that tree does not exceed maximum (minimum) predefined height.

Terminal mutation selects random leaf and replace it by the new terminal.

Shrink mutation replaces a randomly chosen sub-tree with a randomly created terminal.

Node replace mutation selects one node randomly and replace it with the node of the same type.

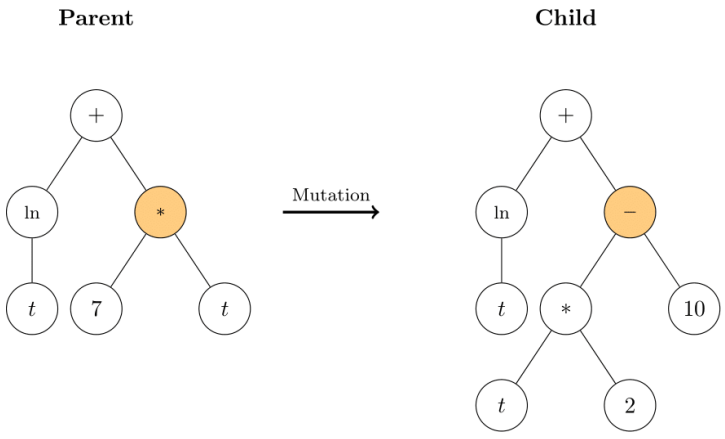

Fig. 2. An example of the mutation operator

Crossover operators

Crossover is a classic genetic operator that takes random sub-tree (can be just a leaf) in each of the parent and swap them, see an example in the fig. 3.

Size-fair crossover same as previous, except it restricts choosing nodes that can overflow maximum tree height.

Common point crossover finds the common points of two trees (the example is shown in the fig. 4) and apply crossover for randomly selected pair of nodes.

Uniform crossover similar to the previous method, finds the common nodes, but swaps each one with coin toss probability. If a node to be swapped belongs to the common region and is a function, then the root sub-tree is also inherited.

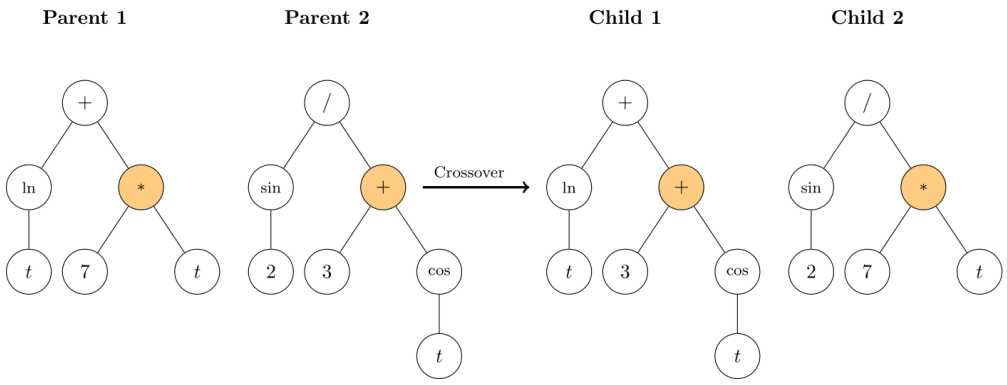

Fig. 3. An example of the crossover operator

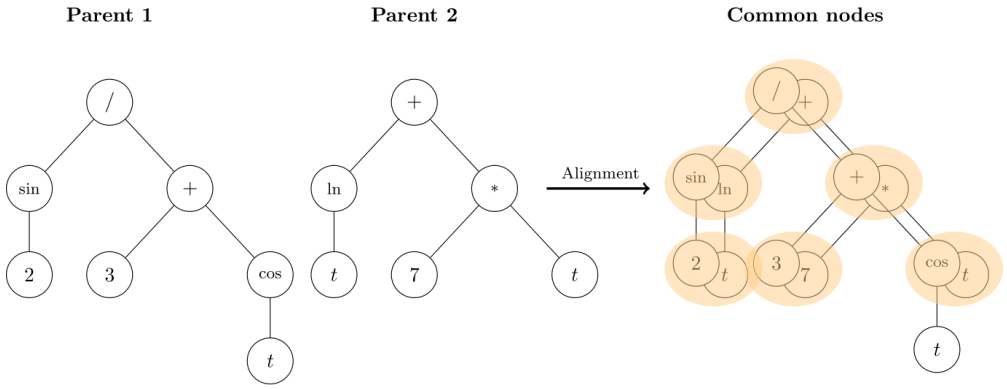

Fig. 4. An example of the uniform and one point crossover point choosing

To generate the initial population we need to have a random tree generation algorithm. Let's consider **tree generation algorithms**, for more details about it, we refer to [15].

Full-Grow is a classic algorithm that combines Grow and Full.

Full-Grow with minimum height limit is like the Full-Grow algorithm, only that at least one terminal node is on predefined minimum height level.

PTC-1 alternately choose new child nodes from either the nonterminals or terminals by user defined q_n and q_t probabilities from non-terminal F and terminal T set.

PTC-2 is a modification of the PTC-1 that additionally provides user to define tree size distribution $\omega_1, \omega_2, ..., \omega_s$.

On the whole, the initial population of pop_{size} random chromosomes is generated. For each of the chromosome fitness is computed by the (5). Using crossover and mutation operators with selected probabilities (p_{cross}, p_{mut}) new chromosomes are created, that make the new population. The process continues until the stopping rules are met. For the stopping rules we use maximum not decreasing fitness iterations count $t_{max} > 0$ and chosen accuracy $\delta_{max} > 0$.

At the end we select the best chromosome and use it as the numerical approximation to the unknown function r. The main algorithm structure is given in the fig. 5.

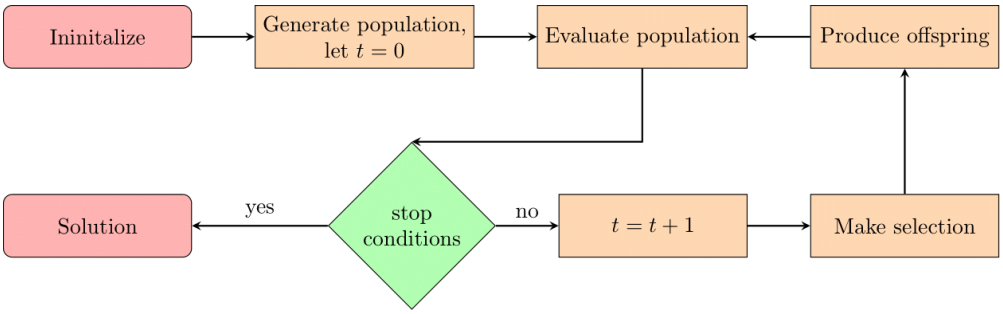

Fig. 5. The main steps of the genetic programming

In the next section, we focus on computing chromosome fitness. Thus, we have the representation of the $\tilde{r}$ for which the boundary $\tilde{\Gamma}_1$ corresponds. For simplicity we still use notation r and Γ_1.

3. Boundary integral equations method for computing of the fitness function

Let's consider the numerical solution of the well-posed mixed boundary value problem (1)-(2).

Following [1,2,5,13] the solution to the mixed problem is represented as a sum of two single layer potentials

$$u(x) = \sum_{\ell=1}^{2} \int_{\Gamma_\ell} \varphi_\ell(y)\Phi(x,y)ds(y), \quad x \in D, \tag{6}$$

where $\Phi(x,y) = \dfrac{1}{2\pi} \ln \dfrac{1}{|x-y|}$, $x \neq y$ being a fundamental solution of the two-dimensional Laplace equation and $\varphi_\ell \in C(\Gamma_\ell)$, $\ell = 1, 2$ unknown densities. Let's consider single-layer

operator and its normal derivative

$$(S_{k,\ell}\varphi)(x) = \int_{\Gamma_\ell} \varphi(y)\Phi(x,y)ds(y), \quad x \in \Gamma_k \tag{7}$$

and

$$(T_{k,\ell}\varphi)(x) = \int_{\Gamma_\ell} \varphi(y)\frac{\partial\Phi(x,y)}{\partial\nu(x)}ds(y), \quad x \in \Gamma_k. \tag{8}$$

Using properties of the single-layer potentials, we reduce the mixed boundary value problem (1)-(2) to the system of well-posed boundary integral equations

$$\begin{cases} S_{1,1}\varphi_1 + S_{1,2}\varphi_2 = 0 \quad \text{on } \Gamma_1, \\ T_{2,1}\varphi_1 + \dfrac{1}{2}\varphi_2 + T_{2,2}\varphi_2 = g \quad \text{on } \Gamma_2. \end{cases} \tag{9}$$

We assume that the known boundary Γ_2 has following parametrization

$$\Gamma_2 = \{x_2(t) = (x_{21}(t), x_{22}(t)),\ t \in [0, 2\pi]\}, \tag{10}$$

where $x_{2\ell}$, $\ell = 1, 2$ are given functions. Using (4) and (10) we can transform the system (9) to the following parameterised system

$$\begin{cases} \tilde{S}_{1,1}\psi_1 + \tilde{S}_{1,2}\psi_2 = 0 \quad \text{on } [0, 2\pi], \\ \tilde{T}_{2,1}\psi_1 + \dfrac{1}{2}\psi_2 + \tilde{T}_{2,2}\psi_2 = \tilde{g} \quad \text{on } [0, 2\pi], \end{cases} \tag{11}$$

where the parameterised integral operators are given as

$$\left(\tilde{S}_{k,\ell}\varphi\right)(t) = \frac{1}{2\pi}\int_0^{2\pi} \varphi(x_\ell(\tau))H(x_k(t), x_\ell(\tau))|x_\ell'(\tau)|d\tau, \quad t \in [0, 2\pi] \tag{12}$$

and

$$\left(\tilde{T}_{k,\ell}\varphi\right)(t) = \frac{1}{2\pi}\int_0^{2\pi} \varphi(x_\ell(\tau))Q(x_k(t), x_\ell(\tau))|x_\ell'(\tau)|d\tau, \quad t \in [0, 2\pi], \tag{13}$$

with kernels defined as

$$H(x,y) = 2\pi\Phi(x,y) \quad \text{and} \quad Q(x,y) = \frac{\partial H(x,y)}{\partial\nu(x)} = -\frac{(x-y, \nu(x))}{|x-y|^2},$$

here $(\cdot,\cdot)$ denotes the inner product. We also introduced new functions $\psi_1(t) = \varphi_1(x_1(t))$, $\psi_2(t) = \varphi_2(x_2(t))$, $\tilde{g}(t) = g(x_2(t))$, for $t \in [0, 2\pi]$.

The integral operators $\tilde{S}_{l,l}$ and $\tilde{T}_{l,l}$, for $\ell = 1, 2$ are weekly singular. Straightforward calculations lead to following:

$$\left(\tilde{S}_{l,l}\varphi\right)(t) = \frac{1}{2\pi}\int_0^{2\pi} \varphi(x_l(\tau))\left[K_l(t,\tau) - \frac{1}{2}\ln\left(\frac{4}{e}\sin^2\left(\frac{t-\tau}{2}\right)\right)\right]|x_l'(\tau)|d\tau, \tag{14}$$

with

$$K_l(t,\tau) = \begin{cases} \dfrac{1}{2}\ln\dfrac{4\sin^2\left(\dfrac{t-\tau}{2}\right)}{e|x_l(t) - x_l(\tau)|^2},\ t \neq \tau, \\ \\ \dfrac{1}{2}\ln\dfrac{1}{e|x_l'(t)|^2},\ t = \tau. \end{cases}$$

and

$$\left(\tilde{T}_{l,l}\varphi\right)(t) = \frac{1}{2\pi}\int_0^{2\pi} \varphi(x_l(\tau))M_l(t,\tau)|x_l^{'}(\tau)|d\tau, \tag{15}$$

with

$$M_l(t,\tau) = \begin{cases} -\dfrac{(x_l(t)-x_l(\tau),\nu(x_l(t)))}{|x_l(t)-x_l(\tau)|^2}, \ t \neq \tau, \\ \\ \dfrac{1}{2}\dfrac{(x_l^{''}(t),\nu(x_l(t)))}{|x_l'(t)|^2}, \ t = \tau. \end{cases}$$

By the Riesz theory [13] we receive that the system of integral equations (11) for given $\tilde{g} \in C^{m+1,\alpha}[0,2\pi]$ has a unique solution $\psi_\ell \in C^{m,\alpha}[0,2\pi]$, $\ell = 1,2$, where for $m \in \mathbb{N}\cup\{0\}$ and $0 < \alpha < 1$, by the $C^{m,\alpha}[0,2\pi]$ we denote the Hölder space.

Let's describe how to fully discretise the parameterised system (11). Following quadrature rules are used for integrals with continuous integrand

$$\frac{1}{2\pi}\int_0^{2\pi} f(\tau)d\tau \approx \frac{1}{2n}\sum_{j=0}^{2n-1} f(t_j) \tag{16}$$

and for integrals with weak singular integrand

$$\frac{1}{2\pi}\int_0^{2\pi} \ln\left(\frac{4}{e}\sin^2\left(\frac{t-\tau}{2}\right)\right) f(\tau)d\tau \approx \sum_{j=0}^{2n-1} R_j(t)f(t_j), \tag{17}$$

where

$$R_j(t) = -\frac{1}{2n}\left\{1 + 2\sum_{m=1}^{n-1}\frac{1}{m}\cos(m(t-t_j)) + \frac{\cos(n(t-t_j))}{n}\right\}, \ t \in [0,2\pi].$$

For the discretization of the (11) we use the Nyström method, see [13]. By applying quadrature rules (16), (17) and collocating in points $t_i = \dfrac{\pi}{n}i$, $i = 0,\ldots,2n-1$ from the (11) we obtain following linear system

$$\begin{cases} \displaystyle\sum_{j=0}^{2n-1}\psi_{1,j}A_{i,j}^{1,1} + \sum_{j=0}^{2n-1}\psi_{2,j}A_{i,j}^{1,2} = 0, \quad i = 0,\ldots,2n-1, \\ \displaystyle\sum_{j=0}^{2n-1}\psi_{1,j}A_{i,j}^{2,1} + \sum_{j=0}^{2n-1}\psi_{2,j}A_{i,j}^{2,2} = \tilde{g}(t_i), \quad i = 0,\ldots,2n-1, \end{cases} \tag{18}$$

where $\psi_{\ell,j} \approx \psi_\ell(t_j)$, $\ell = 1,2$, $j = 0,\ldots,2n-1$ and coefficients given as

$$A_{i,j}^{1,1} = \left[\frac{1}{2n}K_1(t_i,t_j) - \frac{1}{2}R_j(t_i)\right]|x_1^{'}(t_j)|, \qquad A_{i,j}^{1,2} = \frac{1}{2n}H(x_1(t_i),x_2(t_j))|x_2^{'}(t_j)|,$$
$$A_{i,j}^{2,1} = \frac{1}{2n}Q(x_2(t_i),x_1(t_j))|x_1^{'}(t_j)|, \qquad A_{i,j}^{2,2} = \frac{1}{2n}M_2(t_i,t_j)|x_2^{'}(t_j)| + \frac{1}{2}\delta_{ij}.$$

Solving the (18) we can build the numerical approximation to the function u. Finally, using quadrature (16) we can compute the fitness (5)

$$J^2(r) \approx \sum_{i=0}^{2n-1} \left[\sum_{j=0}^{2n-1} \psi_{1j} C_j^1(t_i) + \sum_{j=0}^{2n-1} \psi_{2j} C_j^2(t_i) - f(x_2(t_i)) \right]^2, \tag{19}$$

where $\psi_{\ell,j}$, $\ell = 1, 2$ is a solution of the (18) and

$$C_j^1(t) = \frac{1}{2n} H(x_2(t), x_1(t_j)) |x_1'(t_j)|, \qquad C_j^2(t) = \left[\frac{1}{2n} K_2(t, t_j) - \frac{1}{2} R_j(t) \right] |x_2'(t_j)|.$$

The error estimate of this method can be covered via the theory of collective compact operators [13].

4. Algorithm configuration and numerical examples

In this section, we describe the configuration of the genetic programming algorithm and present the results of two numerical experiments for exact and noisy data.

Table 1

Configuration of genetic programming parameters

Genetic operator	Probability
Mutation operators	
Size-fair	80%
Shrink	80%
Terminal	60%
Node replace	60%
Crossover operators	
Size-fair	50%
Uniform	80%
Common point	90%

(a) genetic operators probabilities

Functions	Probability
Unary operators	
abs	40%
sin	100%
cos	100%
ln	30%
sqrt	100%
exp	30%
Binary operators	
+	100%
−	100%
*	100%
/	25%
power	30%
Terminal	
$[-100; 100]$	100%
t	100%

(b) tree nodes type probabilities

4.1. Algorithm configuration

The following parameters of the genetic programming are used:

- number of chromosomes in the tournament selection method is 3;
- in the r-model selection, r is 40% of pop_{size} (see [18]);
- minimum tree height is 1;
- maximum tree height is 20;

- expected tree size in PTC-1 is 15;
- Full-Grow algorithm iterates in tree height ranges $[1;5], [5;10], [10;20]$. The idea of bracing tree height is to make it easier to analyze the results (and probably adjust parameters) for smaller trees and not to generate the same small (probably worse) trees at large height;
- curvature threshold $\tilde{\kappa}$ is 500 (the parameter was chosen based on several numerical tests for $n = 64$);
- maximum accuracy $\delta_{max} = 10^{-3}$;
- nondecreasing-fitness number of iterations $t_{max} = 100$;
- other parameters are presented in table 1 (a).

Table 2

Errors in case of different algorithm configurations for exact and noisy data for the example 1

	ε	pop_{size}	p_{mut}	p_{cross}	iteration	$\|\tilde{\delta}\|$
Implementation 1	0%	1000	25%	75%	599	$3.20e-3$
	2%				577	$9.74e-2$
	5%				922	$1.99e-1$
Implementation 2	0%	200	75%	30%	3058	$1.64e-2$
	2%				936	$7.21e-2$
	5%				1415	$2.86e-1$
Implementation 3	0%	1000	25%	75%	1123	$3.90e-3$
	2%				1653	$9.05e-2$
	5%				324	$2.94e-1$

For the fitness function computing in (19) $n = 64$ is used. In the method, we combine different algorithms of the generation of the tree-chromosome and different genetic operators. Let's consider several implementations.

Implementation 1. This implementation uses Full-Grow with minimum heigh limit algorithm as a tree generator, all crossover operators and mutations such as Size-fair and terminal. The goal of this implementation is to work with the fixed tree height range to prevent using big trees, which usually have high curvature, for a small number of collocation points.

Implementation 2. A classic implementation that uses PTC-1 as a tree generator and all genetics operators with exception of Size-fair. The probability of choosing each terminal/non-terminal is defined in the table 1 (b).

Implementation 3. The same configuration as in the previous one, assumes the use of PTC-2. The tree size distribution is chosen as normal cumulative density function with mean $\nu = 14.5$ and variance $\sigma = 5$.

Having described the configuration of the algorithm, let's consider two examples.

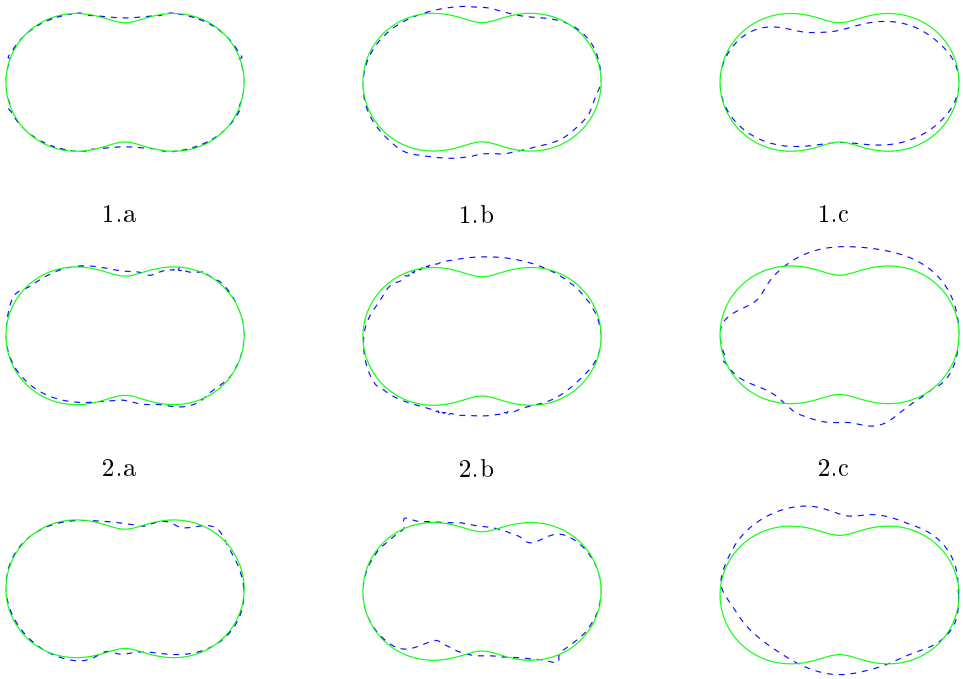

Fig. 6. Numerical approximations of the inner boundary Γ_1 for three implementations for the example 1

4.2. Numerical examples

Let us describe the input data for numerical examples. The noised data g^ε are generated from the exact data g by the following rule

$$g^\varepsilon = g + \varepsilon(2\eta - 1)\|g\|_{L_2},$$

where ε is a noise level and η is a random value in a range $(0, 1)$. For both examples we use the same outer boundary Γ_2, defined by

$$\Gamma_2 = \{x_2(t) = 2(\cos(t), \sin(t)),\ t \in [0, 2\pi]\}.$$

The function f is numerically generated by the solving the mixed boundary value problem (1)-(2) with $g(x) = 2,\ x \in \Gamma_2$ and exact boundary Γ_1.

For the first example, as the exact representation we choose the following function r

$$r(t) = \sqrt{\cos^2 t + \frac{\sin^2 t}{4}}, \quad t \in [0, 2\pi].$$

The results and errors of the reconstruction of the inner boundary Γ_1 for different implementations in case of exact and noised data are provided in the table 2 and in the

fig. 6 ((1.a) – (1.c) for the implementation 1, (2.a) – (2.c) for the implementation 2, (3.a) – (3.c) for the implementation 3).

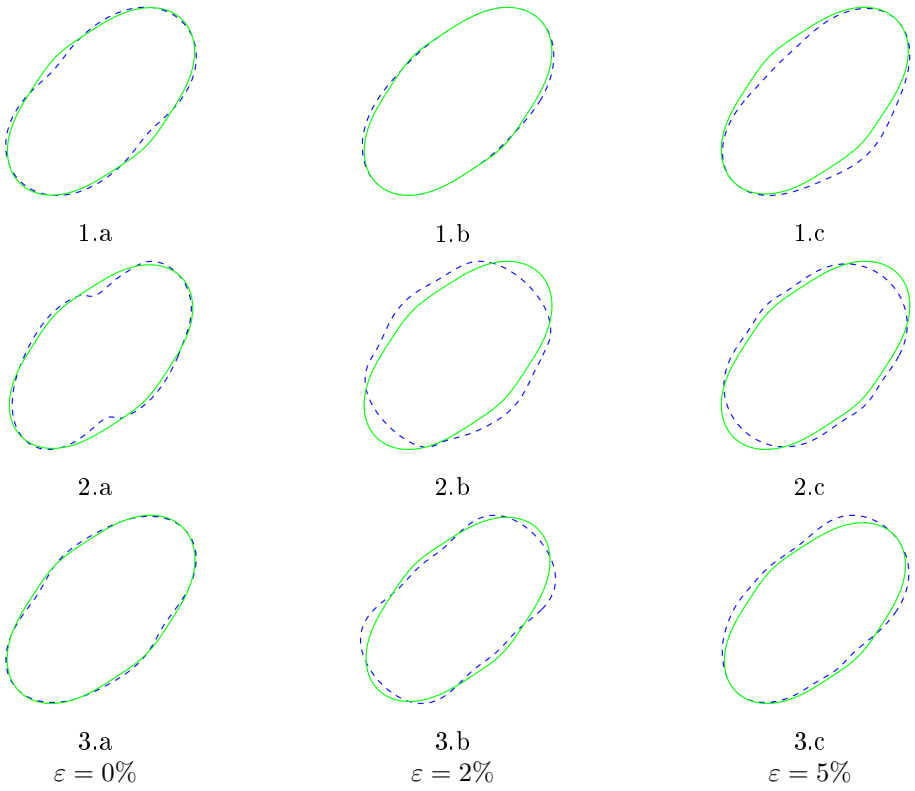

Fig. 7. Numerical approximations of the inner boundary Γ_1 for three implementations for the example 2

And for the second example, as the exact representation we choose the following function r

$$r(t) = \cos(\sin(\cos t - \sin t)), \quad t \in [0, 2\pi].$$

The results and errors of the reconstruction of the inner boundary Γ_1 for different implementations in case of exact and noised data are provided in the table 3 and in the fig. 7 ((1.a)–(1.c) for the implementation 1, (2.a)–(2.c) for the implementation 2, (3.a)–(3.c) for the implementation 3).

In additional we note that the fitness calculation, which usually takes most of program execution time, can be easily accelerated with parallelization. In that approach the CUDA technology is used, see [21]. The comparison of the average execution time per iteration with and without parallelization is shown in table 4.

As it can be seen from the result of numerical examples, the genetic programming can be used as an algorithm for solving ill posed problem of reconstruction of the inner boundary Γ_1. For exact input data and noised data (up to 5%) the inner boundary is

Table 3

Errors in case of different algorithm configurations for exact and noisy data for the example 2

	ε	pop_{size}	p_{mut}	p_{cross}	iteration	$\|\tilde{\delta}\|$
Implementation 1	0%	800	25%	75%	353	$1.43e-2$
	2%				80	$1.75e-1$
	5%				223	$3.45e-1$
Implementation 2	0%	300	75%	30%	972	$4.09e-2$
	2%				957	$3.24e-1$
	5%				60	$3.47e-1$
Implementation 3	0%	700	30%	80%	450	$6.50e-3$
	2%				145	$2.28e-1$
	5%				210	$2.83e-1$

Table 4

Comparison of execution time per iteration before and after parallelization in seconds

n	population size = 50			population size = 200		
	before	after	rate	before	after	rate
8	0.73	0.03	21	2.53	0.15	17
16	4.83	0.03	142	17.34	0.21	81
32	36.31	0.04	760	120.13	0.23	512
64	274.65	0.09	2937	600.58	0.41	1459

reconstructed, for a higher noise level the results are becoming corrupted. Provided errors confirm the application of proposed method. A combination of the genetic programming with the classical regularization algorithm can be further explored to obtain a better accuracy.

5. Conclusions

The genetic programming method has been applied for reconstruction of the inner boundary of the annular planar domain from the known data of the measurement of the harmonic function. Candidate solutions (chromosomes) are represented in a tree form for which genetic operators are described. For each chromosome, the fitness function is obtained by solving the well-posed mixed boundary value problem. The boundary value problem is numerically solved by the method of boundary integral equations. Various algorithm configurations and values of the genetic algorithm parameters are also included. At the end, results of numerical experiments for both exact and noisy data are presented, which confirms the applicability of the proposed approach.

ACKNOWLEDGMENTS

The authors are grateful to professor Roman Chapko for discussions on the topic of the paper.

REFERENCES

1. Baravdish G. An iterative method for the Cauchy problem for second-order elliptic equations / G. Baravdish, I. Borachok, R. Chapko, B.T. Johansson, M. Slodička // International Journal of Mechanical Sciences. – 2018. – Vol. 142. – P. 216–223.
2. Cakoni F. Integral equations for inverse problems in corrosion detection from partial Cauchy data / F. Cakoni, R. Kress // Inverse Probl. Imaging. – 2007. – Vol. 1. – P. 229–245.
3. Chapko R. On the non-linear integral equation approaches for the boundary reconstruction in double-connected planar domains / R. Chapko, O. Ivanyshyn Yaman, T. Kanafotskyi // Journal of Numerical and Applied Mathematics. – 2016. – Vol. 2, (122). – P. 7–20.
4. Chapko R. On a nonlinear integral equation approach for the surface reconstruction in semi-infinite-layered domains / R. Chapko, O. Ivanyshyn, O. Protsyuk // Inverse Problems in Science and Engineering. – 2013. – Vol. 21. – P. 547–561.
5. Chapko R. On the use of an integral equation approach for the numerical solution of a Cauchy problem for Laplace equation in a doubly connected planar domain / R. Chapko, B.T. Johansson, Y. Savka // Inverse Problems in Science and Engineering. – 2013. – Vol. 22. – P. 130–149.
6. Chapko R. On the Non-Linear Integral Equation Approach for an Inverse Boundary Value Problem for the Heat Equation / R. Chapko, L. Mindrinos // Journal of Engineering Mathematics. – 2019. – Vol. 119. – P. 255–268.
7. Eckel H. Numerical study of an evolutionary algorithm for electrical impedance tomography, Dissertation / H. Eckel. – Göttingen, 2008.
8. Eckel H. Non-linear integral equations for the complete electrode model in inverse impedance tomography / H. Eckel, R. Kress // Applicable Analysis. – 2008. – Vol. 87. – P. 1267–1288.
9. Ivanyshyn O. Nonlinear integral equation methods for the reconstruction of an acoustically sound-soft obstacle / O. Ivanyshyn, B.T. Johansson // J. Integral Equations Appl. – 2007. – Vol. 19. – P. 289–308.
10. Ivanyshyn O. Nonlinear integral equations for solving inverse boundary value problems for inclusions and cracks / O. Ivanyshyn, R. Kress // J. Integral Equations Appl. – 2006. – Vol. 18. – P. 13–38.
11. Goldberg D.E. Genetic Algorithm in Search, Optimisation and Machine Learning / D.E. Goldberg. – Addison-Wesley, Reading, MA, 1989.
12. Holland J.H. Adaptation in Natural and Artificial System / J.H. Holland. – University of Michigan Press, Ann Arbor, 1975.
13. Kress R. Linear Integral Equations, 2nd. ed. / R. Kress. – Springer-Verlag, Berlin Heidelberg New York, 1999.
14. Kress R. Nonlinear integral equations and the iterative solution for an inverse boundary value problem / R. Kress, W. Rundell // Inv. Probl. – 2005. – Vol. 21. – P. 1207–1223.
15. Luke S. Issues in Scaling Genetic Programming: Breeding Strategies, Tree Generation, and Code Bloat / S. Luke. – [Dissertation] University of Maryland, 2000.
16. McLean W. Strongly Elliptic Systems and Boundary Integral Operators, Cambridge University Press / W. McLean. – Cambridge, 2000.
17. Mera N.S. On the use of Genetic Algorithms for Solving Ill-Posed Problems / N.S. Mera, L. Elliott, D.B. Ingham // Inverse Problems in Engineering. – 2010. – Vol. 11. – P. 105–121.
18. Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs, 3rd ed. / Z. Michalewicz. – Springer-Verlag, Berlin, 1996.

19. Pourgholi R. Solving an inverse heat conduction problem using genetic algorithm: Sequential and multi-core parallelization approach / R. Pourgholi, H.L. Dana, S.H. Tabasi // Applied Mathematical Modelling. – 2014. – Vol. 38. – P. 1948–1958.
20. ShahnazariM.R. Solving inverse heat conduction problems by using Tikhonov regularization in combination with the genetic algorithm / M.R. Shahnazari, A. RoohiShali, A. Saberi // Inv. Probl. – 2021. – Vol. 3. – P. 60–66.
21. Trujillo L. GSGP-CUDA – a CUDA framework for geometric semantic genetic programming / L. Trujillo, J.M. Muñoz Contreras, D.H. Hernandez, M. Castelli J.J. Tapia // SoftwareX. – 2022. – Vol. 18. – Article 101085.

Article: received 17.07.2023
revised 06.09.2023
printing adoption 20.09.2023

ЗАСТОСУВАННЯ ГЕНЕТИЧНОГО ПРОГРАМУВАННЯ ДЛЯ ЧИСЕЛЬНОГО РОЗВ'ЯЗУВАННЯ ДВОВИМІРНОЇ ЗАДАЧІ РЕКОНСТРУКЦІЇ ГРАНИЦІ

І. Борачок, А. Марченко

Львівський національний університет імені Івана Франка,
вул. Університетська 1, Львів, 79000
e-mails: ihor.borachok@lnu.edu.ua, arkadii.marchenko@lnu.edu.ua

Розглянуто застосування генетичного програмування для чисельного розв'язування нелінійної некоректної задачі реконструкції внутрішньої границі двовимірної двозв'язної області за відомими даними вимірювань гармонічної функції. Потенційний розв'язок (індивід) подається у вигляді дерева, для якого функція пристосованості обчислюється шляхом розв'язування мішаної крайової задачі за методом граничних інтегральних рівнянь. Наведено різні конфігурації алгоритму та продемонстровано результати чисельних експериментів для точних і збурених вхідних даних.

Ключові слова: генетичне програмування, задача реконструкції границі, рівняння Лапласа, метод граничних інтегральних рівнянь.