ANZIAM J. 58(2016), 51–77
doi:10.1017/S1446181116000110

PERFORMANCE OF A REAL CODED GENETIC ALGORITHM FOR THE CALIBRATION OF SCALAR CONSERVATION LAWS

S. BERRES ✉ [1], A. CORONEL[2], R. LAGOS[3] and M. SEPÚLVEDA[4]

(Received 6 July, 2015; accepted 21 December, 2015; first published online 1 July 2016)

Abstract

This paper deals with the flux identification problem for scalar conservation laws. The problem is formulated as an optimization problem, where the objective function compares the solution of the direct problem with observed profiles at a fixed time. A finite volume scheme solves the direct problem and a continuous genetic algorithm solves the inverse problem. The numerical method is tested with synthetic experimental data. Simulation parameters are recovered approximately. The tested heuristic optimization technique turns out to be more robust than classical optimization techniques.

2010 *Mathematics subject classification*: primary 90C59; secondary 90C30.

Keywords and phrases: real coded genetic algorithm, parameter identification, scalar conservation laws, flux identification, sedimentation.

1. Introduction

The problem of flux identification with conservation laws has been extensively studied in the literature because of its relevance to many fields of science and engineering. For instance, flux identification arises in chromatography [24, 25], sedimentation problems [4, 5] with particular applications in wastewater treatment [12] or centrifugation [2, 42], flows through porous media [9, 33] and highway traffic models [19] (see also [6, 8] for analytical results).

[1]Departamento de Ciencias Matemáticas y Físicas, Facultad de Ingeniería, Universidad Católica de Temuco, Temuco, Chile; e-mail: sberres@uct.cl.
[2]GMA, Departamento de Ciencias Básicas, Facultad de Ciencias, Universidad del Bío-Bío, Campus Fernando May, Chillán, Chile; e-mail: acoronel@ubiobio.cl.
[3]Departamento de Matemática y Física, Facultad de Ciencias, Universidad de Magallanes, Punta Arenas, Chile; e-mail: richard.lagos@umag.cl.
[4]CI^{2}MA and DIM, Universidad de Concepción, Concepción, Chile; e-mail: mauricio@ing-mat.udec.cl.

This paper represents the first systematic application of a genetic algorithm to the flux identification problem and we find that, in the cases studied, this technique shows promising results for applications. In particular, the coded genetic algorithm proves to be more robust and reliable as a gradient method than the previously employed classical methods.

The common approach across all these application problems is set up by continuum mechanics, considering assumptions and simplifications appropriate to the particulars of the system under study. The governing equations typically comprise a set of conservation laws closed under state laws or constitutive conditions (algebraic relations for the flux function) and a set of initial and boundary conditions.

Typically, in applications, initial and boundary conditions are estimated by empirical measurements. The constitutive functions are determined qualitatively by phenomenological assumptions and quantitatively by available information from experimental data. Thereby, the constitutive equations depend on unknown parameters which are not accessible from experimental procedures. In cases where they are accessible, it would require very complicated, expensive laboratory tests to determine them experimentally. Therefore, there is a need to develop and test parameter identification methods.

In the framework of flux calibration, we consider an inverse problem where the observation profile is given at a fixed time. The goal is to find the flux function such that the entropy solution for an initial boundary value problem of a system of conservation laws is as close as possible to the observed data. Optimization techniques help to obtain the solution of this inverse problem, whereby a cost function quantifies the difference between the observed data and the simulated solutions.

In general, the solution of nonlinear conservation laws depends nonlinearly on the parameters. It develops discontinuities in finite time, independently of the regularity of the coefficients and the initial and boundary conditions [11]. In most of the cases, this kind of behaviour of the direct problem implies that the cost function turns out to be a nonconvex and nondifferentiable function (see, for instance, Figures 2 and 5 for illustration). Thus the inverse problem analysis and the choice of a suitable method needs to address the ability to minimize nonconvex and nondifferentiable functions.

The applications of numerical optimization tools fall into two kinds of optimization methods, namely, deterministic and stochastic, depending on the previous knowledge about the parameters. On the one hand, if some previous information on the range of the parameter values is known, then a local minimum of the cost function can be found by deterministic methods [3, 10]. Here the central and common point is that, close to a given estimation of the parameter values, local convexity of the cost function is assumed and a formal calculus for the discrete gradient is realized in order to improve the approximation. On the other hand, when little or no knowledge of the ranges of the parameter values is available, and if there is no intuitive estimation of the solution of the inverse problem to compensate this drawback, then the use of stochastic optimization methods is suggested [37, 38].

Recently, an exhaustive and complete review of stochastic optimization methods applied to the inverse scattering problem was done by Rocca et al. [37]. They reported a unified view on evolutionary algorithms constructed under competitive or cooperative paradigms [7, 16]. For example, genetic algorithms belong to the class following the competitive paradigm and swarm or ant colony algorithms belong to the class following the cooperative paradigm.

In particular, Rocca et al. [37] clearly describe the main features and disadvantages of the nature-inspired evolutionary algorithms. They elaborate several important historical milestones during the development. For instance, in the case of genetic algorithms the key properties are:

(a) to imitate the principles of biological evolution for the construction of an iterative algorithm;
(b) to belong to the class of global methods which have the main advantage of depending neither explicitly nor implicitly on gradients; and
(c) to require only the evaluation of the cost function in order to obtain a robust optimization.

Hence genetic algorithms usually work well in situations when gradient methods or gradient-like methods turn out to fail in the local convergence due to nondifferentiability of the objective function [35]. Furthermore, they conclude that although the existing evolutionary algorithms are reliable and efficient in the tested inversion problems, the implementation and adoption of these methodologies in new situations is of paramount importance for the theoretical development. Following this suggestion, we study genetic algorithms for the flux identification problem with nonlinear scalar conservation laws.

In the literature, there are a few results available concerning the specific application of genetic algorithms to inverse problems in conservation laws. For instance, Hou et al. [21] presents a review on algorithms used for the inverse problem arising in porous media flow, which is well known as the “history matching problem”. Ingham and Harris [22] recover the coefficients of a linear one-dimensional model for the flow through porous media. They extensively discuss the benefits of genetic algorithms in contrast to more traditional gradient-based techniques. Tang et al. [40] apply a genetic algorithm to identify the flux function of a system of conservation laws of Saint-Venant type modelling the flow of a river network. Akin and Demiral [1] have reconstructed the relative permeabilities that define the fractional flow in the Buckley–Leverett equation.

We note that Akin and Demiral [1], and Ingham and Harris [22] use a binary coded genetic algorithm. In addition, note that a new strategy of evolutionary computation was used by Jebalia et al. [26] to identify the flux in the chromatography system.

In this paper, we present four numerical examples in order to investigate the feasibility and applicability of the continuous genetic algorithm to scalar conservation laws. Although the results are preliminary in the perspective of solving the general inverse problem for a whole class of the considered equation type, we can discern some properties that serve as guidance to a further development in more complex situations. First, we note that the agreement of observed and recovered data is fine; this indicates that the parameters identified by the algorithm represent good approximations to their true values. Furthermore, we observe that the optimum of the numerical cost function depends on the order of the finite volume method used for the discretization of the direct problem.

In this perspective, we believe that the use of a high resolution numerical algorithm for solving the direct problem contributes naturally to the performance of the numerical flux identification in conservation laws. Thus the numerical examples given in this article clearly suggest that the use of genetic algorithms is a powerful numerical tool for parameter identification in conservation laws.

However, before the application of more complex scenarios of theoretical or engineering interest, some aspects have to be improved, for example hybrid algorithms can combine the continuous genetic algorithm with other evolutionary strategies and even with deterministic algorithms. Moreover, given the high computational cost of running finite volume schemes for the forward simulations, a parallelization of the genetic algorithm is strongly needed.

The paper is organized as follows. In Section 2, we introduce the continuous direct and inverse problems together with their respective discretization. In Section 3, we present the continuous genetic algorithm. In Section 4, we document the numerical results. Finally, in Section 5, we present some conclusions.

2. Continuous direct-inverse problem and discretization

In this section, after an outline on the direct and inverse problem, we define the parameter identification problem, the discretization of the direct problem and the discrete cost function.

2.1. Parameter identification problem The direct problem is given by the initial boundary value problem

$$u_t + (f(u))_x = 0, \quad (x,t) \in Q_T = \mathcal{I} \times \mathcal{T}, \tag{2.1}$$

$$u(x,0) = u_0(x), \quad x \in \mathcal{I} = (0,1), \tag{2.2}$$

$$u(\ell,t) = g_\ell(t), \quad \ell \in \{0,1\}, \quad t \in \mathcal{T} = [0,T], \tag{2.3}$$

where t denotes time, x the spatial variable, u the state variable, f the flux function, u_0 the initial condition and $g_\ell, \ell \in \{0,1\}$ the Dirichlet boundary conditions. Typically, we assume that u_0, f, g_0 and g_1 are given functions and we want to find $u(\cdot,T)$ for a finite time $T > 0$. However, the flux function f is often unknown and should be determined by the solution of an inverse problem. This inverse problem is the well-known calibration problem, where a set of experiments is considered in order to have an over-specified problem. For instance, if we assume that $\hat{u}(x)$ is a given experimental solution profile at a fixed time $t = T$, then the inverse problem of flux identification can be formulated as the optimization problem

$$\begin{aligned} &\text{minimize} && J(u) = \frac{1}{2}\int_{\mathcal{I}} |(u-\hat{u})(x)|^2\,dx \\ &\text{subject to} && E(u,p;f)=0, \quad f \in U_{\mathrm{ad}}, \end{aligned} \tag{2.4}$$

where the constraint $E(u, p; f)$ is the weak integral formulation of the direct problem (2.1)–(2.3)

$$\begin{aligned} E(u,p;f) = &-\iint_{Q_T} \{up_t + f(u)p_x\}\,dx\,dt \\ &+ \int_{\mathcal{T}} \{f(g_1(t))p(1,t) - f(g_0(t))p(0,t)\}\,dt - \int_{\mathcal{I}} u_0(x)p(x,0)\,dx \end{aligned}$$

for all $p \in C_0^1(Q_T)$, and the flux admissible set is defined as

$$\mathcal{U}_{ad} = \{f : \mathbb{R} \to \mathbb{R} \mid f \in C^2(\mathbb{R}), f(0) = f'(0) = 0, f''(u) > 0 \text{ for all } u \in I_{\max}\},$$

with the maximum interval

$$I_{\max} = [\min(A), \max(A)] \quad \text{with } A = \{\|u_0\|_{L^\infty(I)}, \|g_1\|_{L^\infty(\mathcal{T})}, \|g_2\|_{L^\infty(\mathcal{T})}\}.$$

Here, note that the optimization problem (2.4) allows us to determine the flux function from $\mathcal{U}_{ad}$ that best matches the observational data. The definition of $\mathcal{U}_{ad}$ is fundamental for the genetic algorithm, since it determines the trial region of the parameter space in which the initial population is sampled. Furthermore, note that the set $\mathcal{U}_{ad}$ is constrained to *convex* flux functions and should be redefined in the case of more general flux functions. For instance, in the case of sedimentation, the flux function f is a nonconvex flux function with one or two inflection points.

For the parameter identification problem, we assume that the flux function depends on a finite number of parameters, denoted by $\mathbf{e} = (e_1, \ldots, e_d) \in \mathbb{R}^d$: that is, $f(\cdot) = f(\cdot; \mathbf{e})$. Typically, the analytical parametric dependence of the flux function f on the parameter set $\mathbf{e}$ comes from a constitutive relationship between a velocity and a density. For instance, in the case of traffic flow modelling, it is assumed that the flux function is defined by $f(u) = uv(u)$, where v is the velocity. For example, in the Lighthill–Whitham model for traffic flow, the velocity is given by $v(u) = v_*(1 - u/\rho^*)$, such that v_* and ρ^* are the parameters for the calibration, (see [29, 30] and Example 4.2). Thus the general formulation of the optimization problem (2.4) is reduced to an optimization problem with respect to d parameters as

$$\begin{aligned} &\text{minimize} && \mathcal{J}(\mathbf{e}) = J(u(\mathbf{e})) \\ &\text{subject to} && E(u(\mathbf{e}), p; f(u(\mathbf{e}); \mathbf{e})) = 0, \\ &&& \mathbf{e} \in D = \{\mathbf{x} \in \mathbb{R}^d : f(u; \mathbf{x}) \in \mathcal{U}_{\mathrm{ad}} \text{ for all } u \in I_{\max}\}. \end{aligned} \tag{2.5}$$

The set $D \subset \mathbb{R}^d$ denotes the set such that the flux belongs to $\mathcal{U}_{\mathrm{ad}}$. This set D is the set of admissible parameters over which the optimization algorithm will search for the optimal parameters.

2.2. Numerical solution of direct problem For the description of the numerical method for the direct problem, let us first recall the standard notation of finite volume methods for conservation laws (see [14, 41] for more details). We subdivide the spatial domain $\mathcal{I}$ into M subintervals K_i of length, $\Delta x = 1/M$, centred at nodes $x_j = j\Delta x$ for $j = 0, \dots, M$. The intervals are defined by $K_j = (x_{j-1/2}, x_{j+1/2})$, where $x_{j\pm1/2} = x_j \pm \Delta x/2$ refers to positions between the nodes. Here, $x_{j+1/2} = (x_j + x_{j+1})/2$ for $j = 0, \dots, M-1$, $x_{-1/2} = x_0 - \Delta x/2$ and $x_{M+1/2} = x_M + \Delta x/2$. The sets K_j are called the cells or control volumes, and their boundaries are called interfaces. Similarly, the temporal domain $\mathcal{T}$ is partitioned into N subintervals of length $\Delta t = T/N$ defined by $R_n = [t_n, t_{n+1})$, where $t_n = n\Delta t$ for $n = 0, \dots, N$. For notational simplicity, we set $Q_j^n = K_j \times R_n$. The numerical solution of (2.1)–(2.3) over Q_j^n is denoted by u_j^n. With this notation, we discretize the equations (2.1)–(2.3).

Indeed, we start with the discretization of the initial condition (2.2) by setting

$$u_j^0 = \frac{1}{\Delta x}\int_{K_j} u_0(x)\,dx \quad j = 0, \dots, M.$$

Then, following the ideas of the finite volume technique, we integrate the equations (2.1)–(2.3) over Q_j^n, denote the mesh ratio by $\lambda = \Delta t/\Delta x$, and deduce the numerical scheme

$$u_j^{n+1} = u_j^n - \lambda\{f_{j+1/2}^n - f_{j-1/2}^n\}, \quad f_{j+1/2}^n \approx \frac{1}{\Delta t}\int_{R_n} f(u(x_{j+1/2}, t))\,dt, \tag{2.6}$$

for all $n = 0, \dots, N$ and $j = 0, \dots, M$, where $f_{j+1/2}^n$, for a stencil with $2p+1$ points, depends on the values at the nodes as $f_{j+1/2}^n = g(u_{j-p+1}^n, \dots, u_{j+p}^n)$. We assume that the numerical flux function $g \in \mathrm{Lip}(\mathbb{R}^{2p}, \mathbb{R})$ is Lipschitz continuous and ensures consistency of the finite volume scheme (2.6) with the discretized differential equation by satisfying $g(u, \dots, u) = f(u)$. Particular and interesting cases are the well-known monotone flux schemes, where $g : \mathbb{R}^2 \to \mathbb{R}$ satisfies the following assumptions [14] which hold for variables within lower and upper bounds, $-\infty < u_m < u_M < \infty$.

(G1) *Lipschitz-regularity:* $g(u, v)$ is locally Lipschitz continuous with respect to each of its variables bounded as $(u, v) \in [u_m, u_M]^2$: that is, both u and v belong to the interval $[u_m, u_M]$.

(G2) *Consistency:* $g(u, u) = f(u)$ for all $u \in [u_m, u_M]$.

(G3) *Monotonicity:* $g(u, v)$ is nondecreasing with respect to its first variable u and nonincreasing with respect to its second variable v, where both variables are bounded as $(u, v) \in [u_m, u_M]^2$.

In the development of numerical analysis for conservation laws, the explicit monotone flux schemes have played a very important role. This was mainly due to their favourable properties, namely, consistency in the finite volume sense, L^∞-stability, bounded variation stability and convergence of the numerical solution to the entropy solution under a Courant–Friedrichs–Lewy (CFL) condition (see [14] for further details). To be precise, in the numerical simulations given in this paper, we consider the numerical flux of Godunov [29], which is defined explicitly for $f \in \mathcal{U}_{\mathrm{ad}}$ by

$$g^{\mathrm{Godunov}}(u, v) = \begin{cases} f(u) & \text{for } u, v > 0 \\ f(v) & \text{for } u, v < 0 \\ \min\{f(u), f(v)\} & \text{for } u \le 0 \le v \\ \max\{f(u), f(v)\} & \text{for } v \le 0 \le u. \end{cases}$$

Naturally, the scheme (2.6) is written as

$$u_0^{n+1} = u_0^n - \lambda\{g^{\mathrm{Godunov}}(u_0^n, u_1^n) - f(u_0^n)\}, \tag{2.7}$$

$$u_j^{n+1} = u_j^n - \lambda\{g^{\mathrm{Godunov}}(u_j^n, u_{j+1}^n) - g^{\mathrm{Godunov}}(u_{j-1}^n, u_j^n)\}, \tag{2.8}$$

$$u_M^{n+1} = u_M^n - \lambda\{f(u_M^n) - g^{\mathrm{Godunov}}(u_{M-1}^n, u_M^n)\}. \tag{2.9}$$

For the stability and convergence of the Godunov scheme (2.7)–(2.9), we ensure that the CFL condition, $\lambda \cdot \max\{|f'(u)| \mid u \in I_{\max}\} \le 1/2$, is satisfied.

2.3. Discrete parameter identification problem Continuous observation data $\hat{u}$ are discretized by

$$\hat{u}_j = \frac{1}{\Delta x}\int_{K_j} \hat{u}(x)\,dx, \quad j = 0, \ldots, M.$$

The natural discretization of the parameter identification problem (2.5) is

$$\begin{aligned} &\text{minimize} \quad \mathcal{J}_\Delta(\mathbf{e}), \quad \mathcal{J}_\Delta(\mathbf{e}) = \frac{1}{2}\sum_{j=0}^{M} |u_j^N(\mathbf{e}) - \hat{u}_j|^2 \Delta x \\ &\text{subject to} \quad u_j^N(\mathbf{e}) \text{ is obtained by (2.7)–(2.9) for } \mathbf{e} \in D. \end{aligned} \tag{2.10}$$

Here we note that the continuous restriction

$$E(u(\mathbf{e}), p; f(u(\mathbf{e}); \mathbf{e})) = 0$$

is replaced by the discrete restriction that $u_j^N(\mathbf{e})$ is the numerical solution of (2.1)–(2.3) obtained by the finite volume scheme (2.7)–(2.9). The consistency between the discrete and continuous restrictions is given by the convergence of the monotone finite volume method.

3. Continuous genetic algorithm for flux estimation

Evolutionary computation techniques imitate the principles of natural selection and evolution. The basis of evolutionary computation form the following four paradigms:

(A) genetic algorithms [20];
(B) genetic programming [27];
(C) evolutionary strategies [34]; and
(D) evolutionary programming [15].

Among these techniques, genetic algorithms are the most popular ones, because they are easy to implement. Complementarily, under certain conditions, they provide global convergence (see [39]).

The first genetic algorithm was proposed by Holland in his pioneering work [20]. After this work, several researchers have been developing various improvements. The development of several versions of the original binary genetic algorithms given by Jong [13] was responsible for a wide and rapid diffusion of genetic algorithms.

Michalewicz [32] developed mainly the point concerning real coded genetic algorithms. In contrast to the classical binary representation of chromosomes, the coding of chromosomes with floating-point representation was introduced and was proved to give significant improvements in the implementation, computation speed and precision. A complete review on genetic algorithms before 2008 is given by Rocca et al. [37].

In the subsequent work, we use the standard terminology of genetic algorithms. For completeness we recall the main concepts, namely, chromosomes, genes, population and generation (for further details see [18, 39]).

Definition 3.1. A chromosome is an array of parameters to be identified: that is, where the cost function is evaluated.

Definition 3.2. A gene is a Cartesian coordinate of the parameter set that represents a chromosome (vector).

Definition 3.3. A population is a set of chromosomes.

Definition 3.4. A generation is the population at the end of one iteration of the genetic algorithm.

Genetic algorithms use several kinds of representation for the chromosomes. The most widely used variants are the binary representation and the floating-point representation, which characterize the so-called “real” and “continuous” genetic algorithms. In this paper, we opt for a floating-point representation, since the corresponding continuous genetic algorithm is inherently faster than the binary genetic algorithm. When the cost function is continuous, the chromosomes do not have to be decoded prior to the evaluation of the cost function. See the books of Haupt and Haupt [18] and Michalewicz [32] for further details on advantages and disadvantages of both representations.

After the selection of the initial population, the continuous genetic algorithm implemented in this paper iterates on the natural selection until the cost function of the best gene reaches an established tolerance, or until the maximum number of iterations is reached.

In their simplest form, standard genetic algorithms are unconstrained optimization methods, whereas the inverse problem formally has the form of minimizing a cost function subject to a constraint. The constraint is represented by the numerical solution of the direct problem $u_j^N(\mathbf{e})$ according to equations (2.7)–(2.9). Also, the constraint contributes to the cost function of the population, which, in turn, is handled in the selection and crossover operations. The algorithm basically consists of five major steps as outlined in Algorithm 1.

The hypercube Ω considered in (a) can be replaced by a convex set such that $\Omega \subset D$. The hypothesis of convexity is needed for the convex combination used in (d).

Algorithm 1 Major steps of implemented genetic algorithm

(a) **Initial population.** Define a full matrix that represents a random initial population $\mathbb{E} = [\mathbf{e}_1|\mathbf{e}_2|\cdots|\mathbf{e}_n]^{\mathrm{T}}$, where each row $\mathbf{e}_j^{\mathrm{T}} \in \Omega = \prod_{i=1}^{d}[l_i, u_i] \subset D \subset \mathbb{R}^d$ with $l_i < u_i$ for $i = 1, \ldots, d$ being a hypercube. Let G be the maximum number of iterations or the maximum number of generations. Initialize the counter $q \in \mathbb{Z}_0^+$ by $q = 0$ and start an iteration over the following steps.

(b) **Cost of the population.** Define the vector **cost** $\in \mathbb{R}^n$ by evaluation of the cost function for each chromosome of the population $\mathbb{E}$: that is, $\mathbf{J} = (\mathcal{J}_\Delta(\mathbf{e}_1), \ldots, \mathcal{J}_\Delta(\mathbf{e}_n))^{\mathrm{T}} \in \mathbb{R}^n$. For the calculation of the cost function, $u_j^N(\mathbf{e})$ is calculated according to equations (2.7)–(2.9). Define the matrix $\hat{\mathbb{E}} = [\mathbb{E}|\mathbf{J}]$. Given $J_{\min}$ as an established tolerance for the cost evaluation, if there is an $\ell \in \{1, \ldots, n\}$ such that $\mathbf{J}_\ell \leq J_{\min}$, then the solution of (2.10) is $\mathbf{e}_\ell$ and the iteration is stopped.

(c) **Select mates.** Select the parents in three stages. First, update $\hat{\mathbb{E}}$ by row permutation until the property $\hat{\mathbb{E}}^q_{1,d+1} \leq \hat{\mathbb{E}}^q_{2,d+1} \leq \ldots \leq \hat{\mathbb{E}}^q_{n,d+1}$ is satisfied. Second, if $s \in (0, 1]$ denotes the selection rate, select the first $[\![ns]\!]$ rows of $\hat{\mathbb{E}}$ and store the submatrix in the so-called mating pool matrix $\mathbb{F}$. Here, $[\![\cdot]\!]$ is the notation of the nearest integer. Third, by applying the roulette wheel weighting, choose the parents from the chromosomes on $\mathbb{F}$.

(d) **Mating.** Define the algebraic law for the crossover of the parents. In this paper, we obtain an offspring by applying a random convex combination of parents' genes at the random crossover point. The mating process stops when $n - [\![ns]\!]$ offspring are generated. Here, in this step, the population matrix $\mathbb{E}$ is updated by considering the parents in the first $[\![ns]\!]$ rows and the offspring in the next ones.

(e) **Mutation.** Given the mutation rate $\mu \in [0, 1]$, define the total number of mutations as $m = [\![\mu(n-1)d]\!]$. Repeat the following mutation process m times: the random gene $\mathbb{E}_{ij}$ is replaced by a random number which belongs to the interval $[l_j, u_j]$. We note that after the mutation has stopped, the population matrix $\mathbb{E}$ is naturally updated. Set $q = q + 1$. If $q \leq G$, then go to item (b); otherwise, calculate the vector $\mathbf{J}$ and the solution is the chromosome $\mathbf{e}_\ell$ such that $\mathbf{J}_\ell \leq \mathbf{J}_j$ for $j \in \{1, \ldots, n\}$.

4. Numerical results

In this section, we consider four examples of applications of the genetic algorithm used for the flux identification in conservation laws. In all the examples, the genetic algorithm is run with a population size $n = 20$, selection rate $s = 0.5$ and tolerance $J_{\min} = 1.0 \times 10^{-6}$ for the cost evaluation. The mutation rate is set to $\mu = 0.2$ for Example 4.1 and to $\mu = 0.37$ for Examples 4.2, 4.3 and 4.4. The maximum number of iterations for Examples 4.1 and 4.2 is $\mathsf{iter} = 100$, for Example 4.3 it is $\mathsf{iter} = 20$ and for Example 4.4 it is $\mathsf{iter} = 200$.

This selection of genetic parameter values is oriented by suggestions of Haupt and Haupt [18]. Here, several heuristic values are found by a set of performance assessments carried out with a range of numerical parameters. During extensive assessment runs, they identified that population size and mutation rate are the most crucial parameters, whereas the crossover rate, selection method and type of crossover are less relevant for the algorithm performance. Specifically, their version of the genetic algorithm "works best (optimizes quickest) with small population sizes and relatively high mutation rates. As the population size increases, the optimal mutation rate decreases". Our own validation of the suggested numerical parameters confirmed that the chosen values appear to offer a good rate of convergence.

Regarding the numerical method for the direct problem, the maximum number of time steps in the finite volume method for the three examples is selected via the relation

$$N = \Big[\!\!\Big[T(0.45\Delta x)^{-1} \max_{u\in I_{\max}} |f'(u)| \Big]\!\!\Big],$$

where the interval $I_{\max}$ is detailed in the description of each example.

4.1. Identification of a single parameter ($d = 1$)

Example 4.1. In this example, the flux function in (2.1) is specified to be of Burgers type, $f(u) = u^\alpha/\alpha$, and the initial boundary conditions are given by

$$u_0(x) = \begin{cases} 0, & (4x-1)(4x-3) > 0 \\ 1 & \text{elsewhere,} \end{cases} \qquad g_0(t) = g_1(t) = 0 \text{ for all } t > 0.$$

We note that $f''(u) = (\alpha-1)u^{\alpha-2}$, which implies that f belongs to the admissible set $\mathcal{U}_{\text{ad}}$ when $\alpha - 1 > 0$, since $I_{\max} = [0,1]$. Thus the restriction set in (2.5) is defined as $D = (1, \infty)$.

We take two analytical observations obtained with $\alpha = 2$ at $T_1 = 1/4$ and $T_2 = 1/2$, denoted by $\hat{u}_1$ and $\hat{u}_2$, respectively. By the method of characteristics (see Figure 1), we deduce that $\hat{u}_i : I \to \mathbb{R}$ for $i = 1, 2$ are defined as

$$\hat{u}_1(x) = u(x, 1/4) = \begin{cases} 0, & (4x-1)(8x-7) > 0 \\ 4x-1, & (4x-1)(2x-1) \le 0 \\ 1 & \text{elsewhere,} \end{cases}$$

$$\hat{u}_2(x) = u(x, 1/2) = \begin{cases} 0, & 4x-1 < 0 \\ 2x-1/2, & (4x-1)(4x-3) \le 0 \\ 1 & \text{elsewhere.} \end{cases}$$

The analytical cost functions are given by

$$\mathcal{J}_1(\alpha) = \frac{1}{8}\Big(\frac{\alpha-1}{\alpha+1} - \frac{2(\alpha-1)}{2\alpha-1} + \frac{1}{3}\Big) + \frac{\alpha-2}{16\alpha}\operatorname{sgn}(\alpha-2), \tag{4.1}$$

$$\mathcal{J}_2(\alpha) = \frac{1}{4}\Big(\frac{\alpha-1}{\alpha+1} - \frac{2(\alpha-1)}{2\alpha-1} + \frac{1}{3}\Big) + \frac{\alpha-2}{16\alpha}\{1 + \operatorname{sgn}(\alpha-2)\}, \tag{4.2}$$

where sgn is the "sign" function.

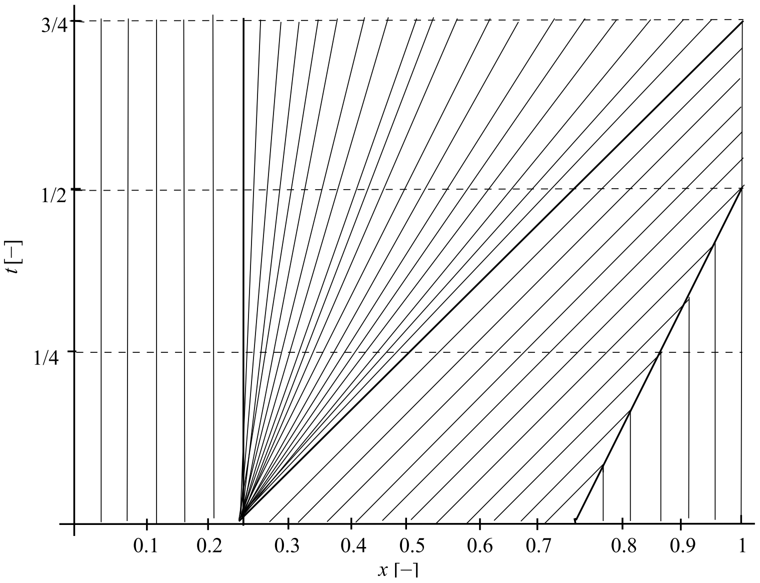


FIGURE 1. (Example 4.1) Configuration of characteristics with $\alpha = 2$.

For the genetic algorithm, we choose $\Omega = [1.1, 4] \subset D$. The graph of the cost function over Ω is shown in Figure 2. In both the plots in Figure 2(a) and Figure 2(b), we note that the optimum values of the numerical and analytical cost functions do not coincide. This behaviour is a natural consequence of the numerical method used for the simulation of the direct problem, since first-order finite volume methods have large numerical diffusion (see [14, 41]).

For the simulation of the direct problem, we use $M = 100$ space intervals and choose a variable number of time steps satisfying the CFL condition.

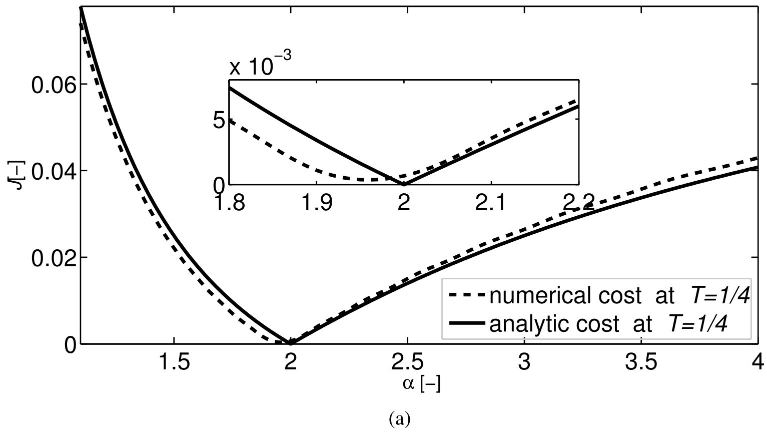
The results of the identification of α are given in Table 1, and the best chromosomes of each iteration are given in Table 2. The observed and identified profiles for $T = 1/2$ and $T = 1/4$ are compared in Figure 3(b) and Figure 3(c), respectively. The observed and identified flux functions are compared in Figure 3(d). From Table 2, we note a fast convergence of the implemented genetic Algorithm 1. The maximum number of generations and the cost tolerance are fixed to 1.0×10^2 and 1.0×10^{-6}, respectively. However, the algorithm found an acceptable optimum within four generations and stops at ten generations, when the stopping criteria of cost tolerance is reached.

In Table 3, we present a comparison of the parameter identification via the genetic algorithm and via the Nelder–Mead simplex method [28] using the `fminsearch` function of MATLAB. Here we take the observation function $\hat{u}_1$ and vary over the number of temporal steps M. The initial guess in the case of the simplex method is 1.1.

From Table 3, it can be seen that the cost function of the global optimum J_{ga} found by the genetic algorithm has the same dominant digits as the corresponding value of the

TABLE 1. (Example 4.1) Evolution of parameter α and cost function value "cost" for observation $\hat{u}_1$ (left) and $\hat{u}_2$ (right). In both cases, the column labelled "initial population" shows the 20 initial random chromosomes and their corresponding cost values before the first generation, and the column labelled "final population" shows the 20 chromosomes and their corresponding cost at the tenth generation. For each generation, the best individual is marked.

Observation $\hat{u}_1$				Observation $\hat{u}_2$			
Initial population		Final generation		Initial population		Final generation	
α	cost	α	cost	α	cost	α	cost
3.0978	0.02858023	1.9594	0.00036738	2.3049	0.01998901	1.9415	2.01182×10^{-4}
1.3222	0.03883032	1.9594	0.00036738	3.3342	0.06547159	1.9415	0.000201182
2.7524	0.02120940	1.9594	0.00036738	1.2853	0.02323202	1.9415	0.000201182
3.4640	0.03503519	1.9593	0.00036739	2.0188	0.00271066	1.9415	0.000201182
1.3758	0.03301540	1.9593	0.00036739	1.2099	0.03258585	1.9415	0.000201182
1.1587	0.06245224	1.9590	0.00036742	2.7299	0.04140269	1.9415	0.000201182
2.7273	0.02055735	1.9590	0.00036742	1.2730	0.02456631	1.9415	0.000201182
3.6778	0.03870158	1.9589	0.00036744	3.9831	0.08510473	1.9414	0.000201183
3.2516	0.03152623	1.9589	0.00036744	2.2301	0.01568804	1.9414	0.000201183
2.8147	0.02276175	1.9589	0.00036744	1.6769	0.00267327	1.9414	0.000201183
2.2748	0.00883104	1.9589	0.00036744	3.3244	0.06508412	1.9414	0.000201183
3.2396	0.03133313	1.9589	0.00036744	1.2958	0.02214488	1.9414	0.000201183
2.2518	0.00808391	1.9589	0.00036744	1.7865	0.00106196	1.9414	0.000201183
1.9375	0.00046090	1.9589	0.00036744	1.6033	0.00441310	1.9414	0.000201183
3.3569	0.03316766	1.9589	0.00036744	2.3370	0.02173338	1.9413	0.000201186
3.6595	0.03844159	1.9589	0.00036744	1.7015	0.00222037	1.9413	0.000201186
3.1396	0.02948630	1.9586	0.00036752	3.0458	0.05484254	1.9879	0.000899494
3.0755	0.02808072	2.7695	0.02165141	3.3520	0.06617539	2.7798	0.043662271
2.7699	0.02166166	3.0405	0.02729970	3.0941	0.05659175	2.8504	0.046921621
2.5272	0.01583049	3.4285	0.03438576	1.5932	0.00470261	3.0967	0.056695360



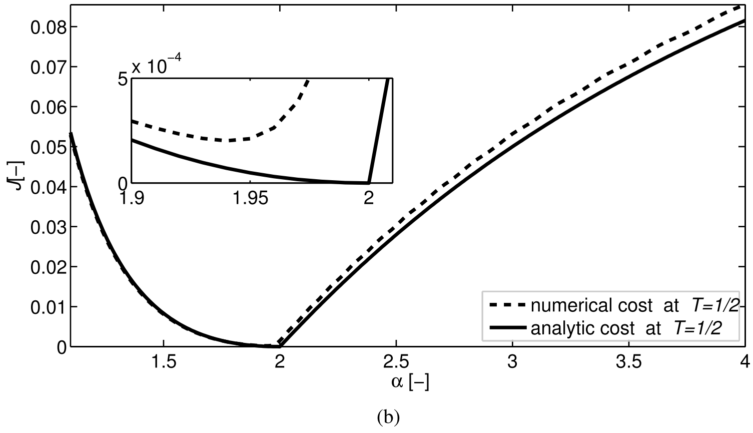


FIGURE 2. (Example 4.1) (a) Analytical cost function (4.1) (solid line) and the numerical cost function (dotted line); (b) analytical cost function (4.2) (solid line) and the numerical cost function (dotted line).

optimal cost J_{sim} calculated by the Nelder–Mead simplex method. However, looking at the further digits, the value of J_{ga} is lower than the value of J_{sim} in 70 per cent of the cases, which demonstrates that the genetic algorithm outperforms the Nelder–Mead simplex method even in a controlled situation, where the initial guess is chosen close to the optimum and the cost function is convex. Such a comparison would be even more in favour of the genetic algorithm in general situations, where the cost function loses convexity and the cost optimum is not known beforehand, so that it is not guaranteed that the initial guess is sufficiently close.

4.2. Identification of a flux modelling traffic flow ($d = 2$)

EXAMPLE 4.2. We note that the basic flux function used for traffic flow models is of the type $\hat{f}(u) = v_* u(1 - u/\rho^*)$, which is a concave function for v_* and where ρ^* belongs to $\mathbb{R}^+$: that is, $\hat{f} \notin U_{\text{ad}}$. It is well known that the analysis of concave flux functions is

TABLE 2. (Example 4.1) The best chromosomes of the first 11 generations and their corresponding cost function values.

q	Observation $\hat{u}_1$		Observation $\hat{u}_2$	
	α	cost	α	cost
0	1.9375	0.00046090	1.7865	0.00106196
1	1.9375	0.00046090	1.9605	0.00026737
2	1.9375	0.00046090	1.9110	0.00025926
3	1.9675	0.00037939	1.9383	0.00020224
4	1.9575	0.00036810	1.9415	0.00020118
5	1.9575	0.00036810	1.9415	0.00020118
6	1.9612	0.00036794	1.9415	0.00020118
7	1.9594	0.00036738	1.9415	0.00020118
8	1.9594	0.00036738	1.9415	0.00020118
9	1.9594	0.00036738	1.9415	0.00020118
10	1.9594	0.00036738	1.9415	0.00020118

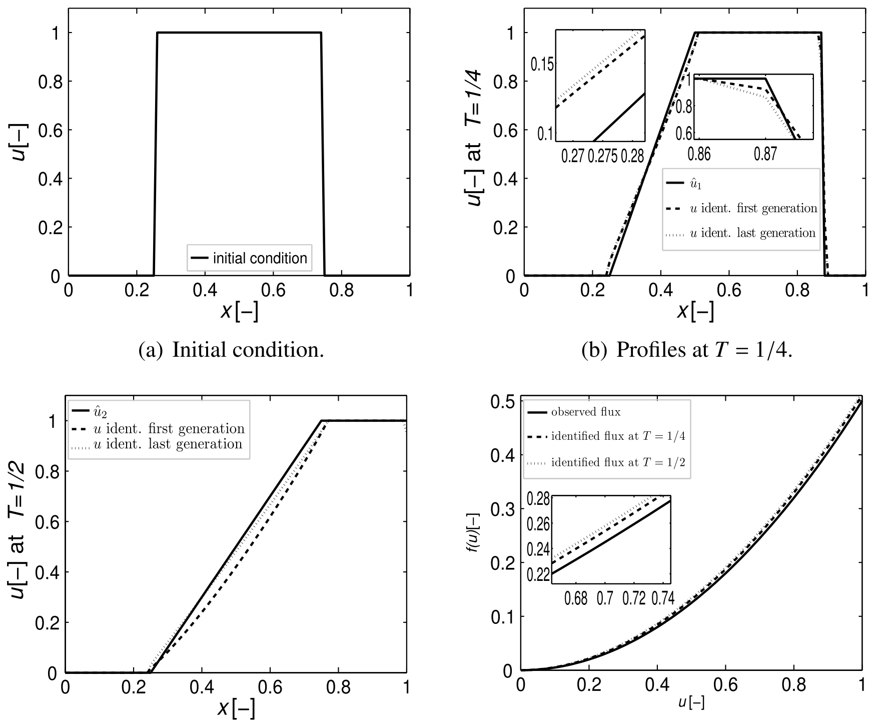
TABLE 3. (Example 4.1) Comparison of the identification via the genetic algorithm and the simplex method as a function of the spatial discretization steps (M). The values of $J_{\rm ga}$ and $J_{\rm sim}$ are the values of the cost function at the convergence point of the genetic algorithm and simplex method, respectively.

M	$J_{\rm sim}$	$J_{\rm ga}$	$J_{\rm sim} - J_{\rm ga}$
10	1.1421×10^{-2}	1.1421×10^{-2}	2.7347×10^{-10}
20	6.8933×10^{-3}	6.8933×10^{-3}	1.3056×10^{-08}
30	3.7897×10^{-3}	3.7897×10^{-3}	-5.6871×10^{-12}
40	4.0919×10^{-3}	4.0918×10^{-3}	1.3905×10^{-07}
50	2.4407×10^{-3}	2.4407×10^{-3}	8.4222×10^{-10}
60	1.9710×10^{-3}	1.9710×10^{-3}	-1.1750×10^{-10}
70	1.5684×10^{-3}	1.5684×10^{-3}	9.4597×10^{-10}
80	1.6682×10^{-3}	1.6682×10^{-3}	-1.5565×10^{-11}
90	1.2927×10^{-3}	1.2927×10^{-3}	4.4827×10^{-12}
100	1.1096×10^{-3}	1.1068×10^{-3}	2.7854×10^{-06}

completely analogous to the case of convex flux functions (see [11, 29] for details). In this specific case, if we redefine the flux function as $\hat{f}_1(u) = -\hat{f}(u) + \hat{f}'(0)u + \hat{f}(0)$, then we have $\hat{f}_1(u) = -v_* u^2/\rho^* \in U_{\rm ad}$ (see [31]). Thus the analysis of the traffic flow model can be carried out by following the theory for convex flux functions.

In this example, we consider a slightly more general flux than $\hat{f}_1(u) = -v_* u^2/\rho^*$, namely,

$$f(u) = \alpha u^\beta, \quad u_0(x) = \begin{cases} 2, & x \in [0, 1/3) \\ 3, & x \in [1/3, 2/3), \\ 1, & x \in [2/3, 1], \end{cases} \quad g_0(t) = 0 \quad \text{and} \quad g_1(t) = 1.$$



(a) Initial condition. (b) Profiles at $T = 1/4$.

(c) Profiles at $T = 1/2$. (d) Identified and analytical flux.

FIGURE 3. (Example 4.1) (a) Initial condition u_0; (b) random best initial profile at the initial evolution, the identified profile and the analytical profile $\hat{u}_1$ at $T = 1/4$; (c) random best initial profile at the initial evolution, the identified profile and the analytical profile $\hat{u}_2$ at $T = 1/2$; (d) comparison of the flux used for the observation profile and the identified flux. For (b)–(d), see Table 1 for numerical values of best initial and identified parameters.

Note that $f \in \mathcal{U}_{\mathrm{ad}}$ if $\alpha\beta(\beta - 1) > 0$, since $I_{\max} = [0, 3]$. Thus the restriction set, defined in (2.5), is given by

$$D = \{\mathbf{e} = (\alpha, \beta) \in \mathbb{R}^2 \mid \alpha\beta(\beta - 1) > 0, f(\cdot; \mathbf{e}) \in \mathcal{U}_{\mathrm{ad}}\}.$$

We consider an analytical observation obtained with $\alpha = 0.25$ and $\beta = 2$ at $T = 1/4$. By the method of characteristics (see Figure 4), we deduce that $\hat{u} : I \to \mathbb{R}$ is defined as

$$\hat{u}(x) = u(x, 1/4) = \begin{cases} 8x, & x(4x - 1) \le 0 \\ 2, & (4x - 1)(12x - 7) < 0 \\ 8x - 8/3, & (12x - 7)(24x - 17) \le 0 \\ 3, & (24x - 17)(11x - 12) < 0 \\ 1, & (11x - 12)(x - 1) \le 0. \end{cases}$$

For the genetic algorithm, we choose $\Omega = [0.1, 3] \times [1.1, 4] \subset D$. The graph of the cost function over Ω is shown in Figure 5. The shapes of the numerical cost functions

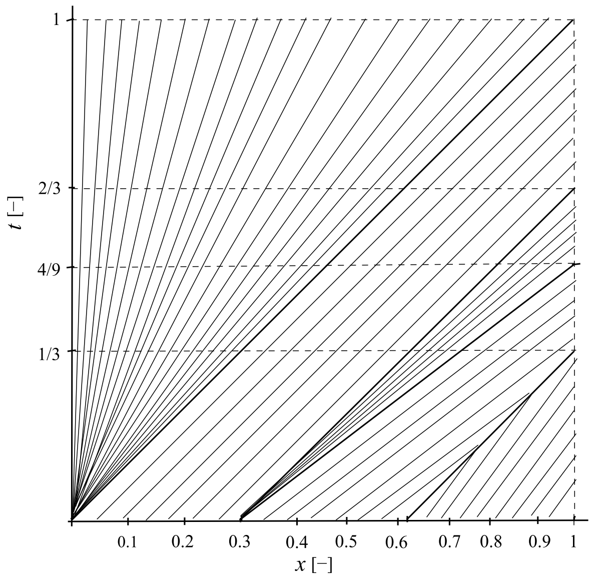


FIGURE 4. (Example 4.2) Characteristics configuration with $\alpha = 0.25$ and $\beta = 2$.

for $M = 100$ and $M = 200$ are similar. However, as in the case of Example 4.1, the numerical diffusion of the Godunov method implies that the optimum for the numerical cost function is slightly different from the optimum of the analytical cost function.

The results of the identification with $M = 100$ and $M = 200$ space steps are given in Table 4. Furthermore, the plot of the profiles is shown in Figure 6.

4.3. Identification of a flux in chromatography ($d = 3$)

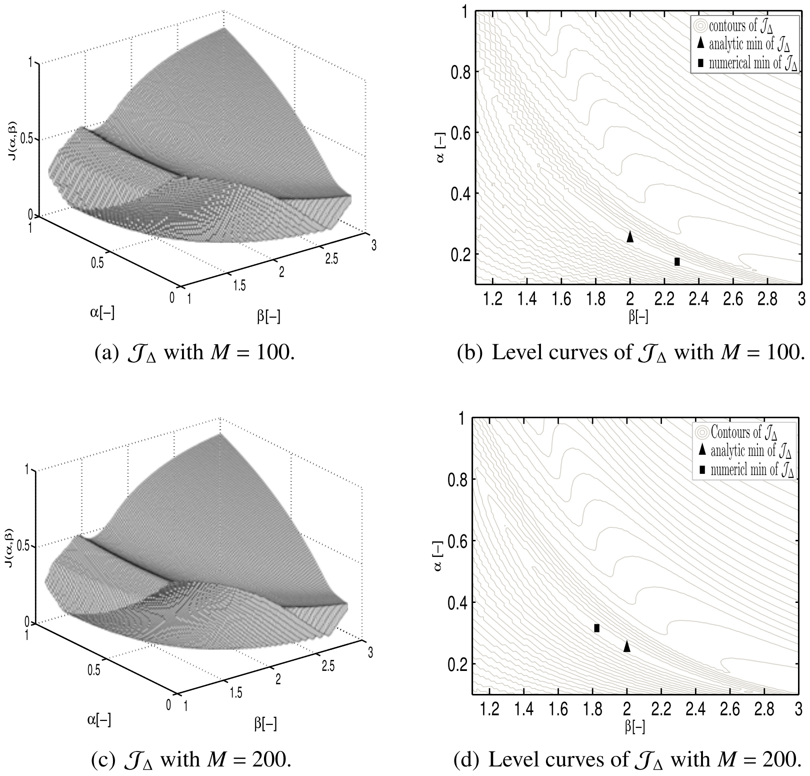
EXAMPLE 4.3. Chromatography is a laboratory technique for the separation of mixtures. It can be reasonably modelled by a first-order hyperbolic system of partial differential equations governed by nonlinear functions of the mixture concentrations, called isotherm functions, which appear as the flux of the mass-balance equations [36]. In this example, we consider a scalar conservation law with a nonlinear flux modelled by a modified Redlich–Peterson isotherm [35]

$$f(u) = \frac{\alpha u^2}{1 + \sigma u^\beta},$$

where α, σ and β are constant parameters. The identification of any of these three parameters α, σ or β from our Redlich–Peterson model is crucial from the theoretical point of view, as well as for the more practical consideration of accurately governing the experiment to improve separation. In this regard, there are several articles in the literature on the identification of other isotherms such as the Langmuir or the

TABLE 4. (Example 4.2) The column labelled $M = 100$ and $M = 200$ show the results for 100 and 200 space steps, respectively. In both cases, the column labelled "initial population" shows the 20 random chromosomes and their corresponding cost values before the first generation and the column labelled "final population" shows the 20 chromosomes and their corresponding cost values at the 100th generation. For each generation the best individuals are marked.

$M = 100$						$M = 200$					
Initial population			Final generation			Initial population			Final generation		
α	β	cost	α	β	cost	α	β	cost	α	β	cost
1.7283	3.4338	1.1239	0.1745	2.2739	0.0063	2.6301	3.1473	1.3036	0.3160	1.8248	0.003762
1.5308	3.1295	1.0845	0.1745	2.2739	0.0063	2.3292	3.2303	1.2522	0.3160	1.8248	0.003762
2.9825	1.1012	1.6864	0.1745	2.2739	0.0063	0.5599	3.4016	0.6811	0.3160	1.8248	0.003762
0.1745	1.9267	0.2250	0.1745	2.2739	0.0063	2.6800	3.2769	1.2974	0.3160	1.8248	0.003762
0.8526	1.5953	0.4348	0.1745	2.2739	0.0063	2.3853	2.3041	1.3524	0.3160	1.8248	0.003762
1.0488	1.6894	0.5162	0.1745	2.2739	0.0063	0.6968	2.4075	0.5552	0.3160	1.8248	0.003762
0.4563	1.3746	0.1175	0.1745	2.2739	0.0063	1.8596	3.1455	1.1722	0.3160	1.8248	0.003762
2.2632	1.2122	1.1530	0.1745	2.2603	0.0107	1.4463	3.3838	1.0674	0.3160	1.8248	0.003762
1.9251	2.2407	1.2175	0.1745	2.3092	0.0158	1.0724	3.8551	0.9622	0.3160	1.8248	0.003762
0.6701	1.7770	0.3540	0.1745	2.3670	0.0567	1.4626	2.6769	1.0689	0.3160	1.8248	0.003762
1.4449	1.4266	0.6714	0.1745	2.9074	0.2070	1.2137	1.8713	0.7703	0.3160	1.8676	0.034107
0.8713	2.3571	0.6876	0.1745	3.4616	0.3626	1.1586	2.7404	0.9467	0.3160	3.1048	0.371130
0.1105	1.9181	0.4183	0.1745	3.5301	0.2774	1.5768	3.0794	1.1062	0.3160	3.7089	0.508626
0.4357	3.5786	0.6029	0.1745	3.6887	0.3097	2.1864	3.1035	1.2385	2.1135	1.8248	1.300688
1.7209	1.2010	0.8032	0.1745	3.6890	0.3106	1.4225	2.5499	1.0477	2.2929	1.5230	1.366459
2.6876	3.9489	1.2326	0.1745	1.7517	0.3238	2.8055	2.0329	1.4883	2.4682	1.8248	1.432283
2.7135	3.8700	1.2413	0.1745	3.8098	0.3339	0.8936	3.9869	0.9050	3.3930	1.8248	1.655020
2.1297	3.9547	1.1666	1.1177	2.2739	0.8411	2.5917	3.1348	1.2996	3.6056	1.8248	1.690803
0.7780	3.4745	0.8196	1.9187	2.4159	1.2101	0.1996	1.4223	0.4374	3.6271	1.8248	1.694224
1.6356	1.2255	0.7686	3.9152	2.9713	1.4552	0.8860	2.3507	0.7004	3.7317	1.8248	1.710099



(a) $\mathcal{J}_\Delta$ with $M = 100$. (b) Level curves of $\mathcal{J}_\Delta$ with $M = 100$.
(c) $\mathcal{J}_\Delta$ with $M = 200$. (d) Level curves of $\mathcal{J}_\Delta$ with $M = 200$.

FIGURE 5. (Example 4.2) Plots and level curves of discrete cost function $\mathcal{J}_\Delta : \Omega \to \mathbb{R}$ with $M = 100$ (above) and $M = 200$ (below) space intervals.

bi-Langmuir model [23, 25]. We suppose that the initial and boundary conditions are given by

$$u_0(x) = \begin{cases} 0 & \text{if } (4x-1)(4x-3) > 0, \\ -x^2 + x - 3/16 & \text{elsewhere,} \end{cases}$$

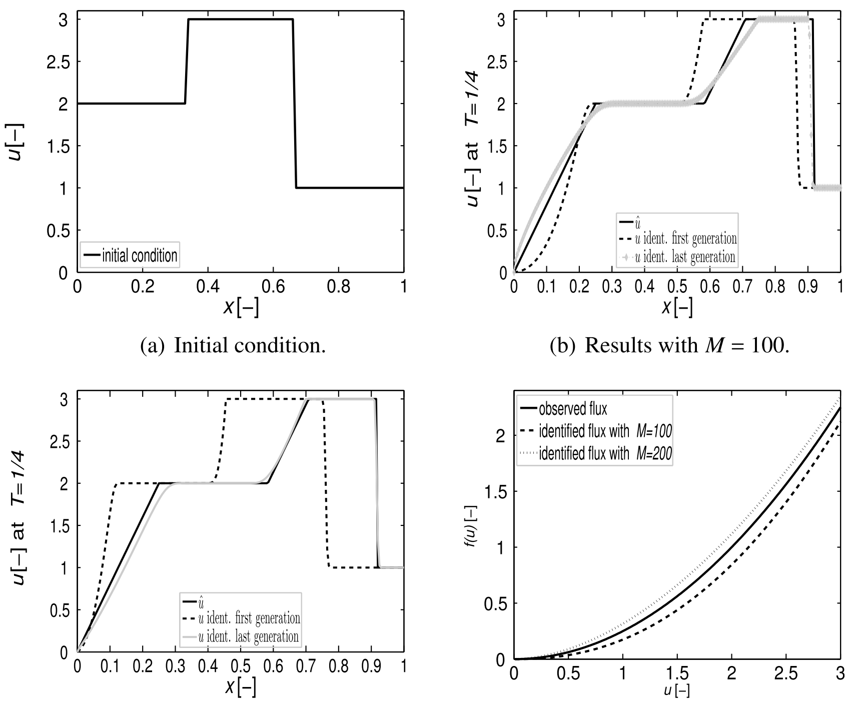
and $g_0(t) = g_1(t) = 0$, respectively. We note that

$$f'(u) = \frac{\alpha u[2 + \sigma(2-\sigma)u^\beta)]}{(1+\sigma u^\beta)^2},$$
$$f''(u) = \frac{\alpha[2 + \sigma(1-\sigma)(4+\beta)u^\beta + \sigma^2(2-\beta)(1-\beta)u^{2\beta}]}{(1+\sigma u^\beta)^3},$$

and $I_{\max} = [0, 1/16]$. Consequently, the parameter identification restriction set (2.5) is defined as

$$D = \{\mathbf{e} = (\alpha, \beta, \sigma) \in \mathbb{R}^3 \mid \alpha > 0,\ \beta \in [1, 2], \sigma > 0 \text{ and } f(\cdot; \mathbf{e}) \in U_{\text{ad}}\}.$$

The observational data are set as a piecewise linear fit to a simulation of the direct problem with parameters $\alpha = \sigma = 1, \beta = 1.5$, $M = 500$ and $T = 3$. More precisely, the



(a) Initial condition. (b) Results with $M = 100$.

(c) Results with $M = 200$. (d) Identified and analytical flux.

FIGURE 6. (Example 4.2) (a) Initial condition u_0; (b) random best initial profile at the initial evolution, the identified profile and the analytical profile $\hat{u}$ at $T = 1/4$ with $M = 100$; (c) random best initial profile at the initial evolution, the identified profile and the analytical profile $\hat{u}_1$ at $T = 1/4$ for $M = 100$; (d) comparison of the flux used for the observation profile and the identified flux. For (b)–(d), see Table 2 for numerical values of best initial and identified parameters.

observation $\hat{u} : I \to \mathbb{R}$ considered for this example is defined as

$$\hat{u}(x) = \begin{cases} 0, & x \in [0, 0.25[\cup]0.85, 1] \\ 0.1178x - 0.0288, & x \in [0.25, 0.598] \\ 0.0984x - 0.0174, & x \in]0.598, 0.75] \\ 0.0534x + 0.0168, & x \in]0.75, 0.85]. \end{cases}$$

For the genetic algorithm we choose $\Omega = [0.1, 2] \times [1, 2] \times [0.1, 2] \subset D$. The results of the identification with $M = 200$ space steps are given in Table 5 and the plots of the results are shown in Figure 7.

4.4. Performance, convergence and stability of the algorithm

EXAMPLE 4.4. In this example, we assume that the velocity is a polynomial function: that is, the flux is of the type

$$f(u) = uv(u) \quad \text{with } v(u) = \sum_{i=1}^{20} e_i u^{i-1},$$

TABLE 5. (Example 4.3) The initial population column shows the 20 random chromosomes and their corresponding cost before the first generation and the final population column shows the 20 chromosomes and their corresponding cost at the twentieth generation. The underlined chromosomes are the best individuals of their generation.

Initial population				Final generation			
α	β	σ	cost	α	β	σ	cost
0.9316	1.3251	0.5399	2.3191×10^{-5}	0.9912	1.7783	0.9433	1.6432×10^{-7}
0.5911	1.7578	0.3877	1.5353×10^{-4}	0.9912	1.8513	1.1029	1.6434×10^{-7}
1.6448	1.1740	0.8284	1.8691×10^{-4}	0.9912	1.8513	1.1029	1.6434×10^{-7}
1.2727	1.8093	1.4439	9.4545×10^{-5}	0.9912	1.8875	0.9433	2.0864×10^{-7}
0.2460	1.9109	1.4288	2.7872×10^{-4}	0.9912	1.8513	0.4929	2.7107×10^{-7}
1.3498	1.7926	1.7633	1.1675×10^{-4}	0.9912	1.8875	0.5755	2.9447×10^{-7}
1.8913	1.6329	0.2545	2.3345×10^{-4}	0.9912	2.1019	0.9433	2.9722×10^{-7}
0.2823	1.2721	1.1690	2.6780×10^{-4}	0.9912	1.8513	1.3111	6.5055×10^{-7}
1.2710	1.8265	0.6863	9.5575×10^{-5}	0.9912	1.8875	0.2158	1.0564×10^{-6}
0.7537	1.2150	1.8534	1.0555×10^{-4}	0.9912	1.7783	1.5167	1.5097×10^{-6}
1.7066	1.0165	1.4701	1.8586×10^{-4}	0.9912	1.4546	0.9433	2.6636×10^{-6}
0.7288	1.8147	1.1011	9.9594×10^{-5}	0.9751	1.8788	1.2793	3.5103×10^{-6}
0.2320	1.3725	0.8953	2.8537×10^{-4}	0.9912	1.2442	0.9433	8.1190×10^{-6}
0.6477	1.1466	0.3904	1.3488×10^{-4}	1.1148	0.7222	1.2465	9.7744×10^{-6}
0.1585	1.0861	1.3607	3.1185×10^{-4}	0.9341	0.6495	0.9207	6.5038×10^{-5}
1.8037	1.1925	0.9423	2.1949×10^{-4}	1.5637	1.7783	1.3346	1.7336×10^{-4}
1.8683	1.2471	0.9874	2.3347×10^{-4}	0.9751	0.1082	1.3553	1.9253×10^{-4}
1.6991	1.0650	1.7788	1.8445×10^{-4}	0.4825	1.7783	0.6419	1.9595×10^{-4}
1.3636	1.9357	0.4889	1.2287×10^{-4}	1.8277	0.9523	0.9020	2.1762×10^{-4}
1.9712	1.1818	1.7713	2.4154×10^{-4}	0.2163	1.8513	1.1029	2.8980×10^{-4}

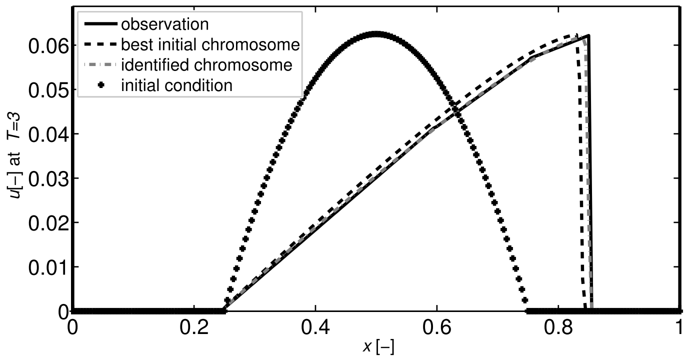
where $\mathbf{e} = (e_1, \ldots, e_{20})$ are the unknown parameters. The initial and boundary conditions are the functions u_0, g_0 and g_1 given for Example 4.1. The observation at $T = 0.5$ is given by

$$\hat{u}(x) = \begin{cases} 0 & \text{for } (x-0.58)(x-2.22) \geq 0, \\ 4.136 - 23.570x + 53.622x^2 - 64.085x^3 & \\ \quad + 44.742x^4 - 18.394x^5 + 4.135x^6 - 0.393x^7 & \text{elsewhere,} \end{cases}$$

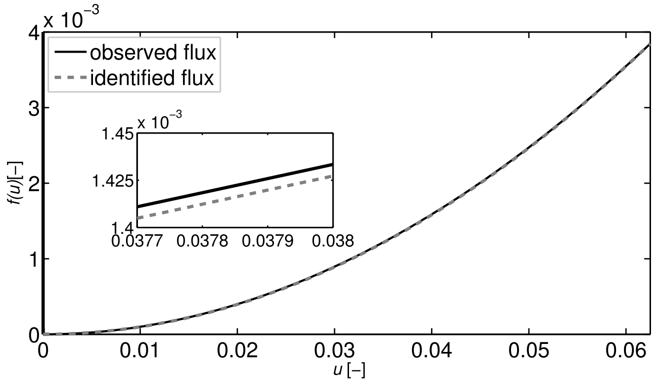
which is a fitted function to the solution of a numerical simulation of the direct problem with $\mathbf{e} = \mathbf{e}_\infty = (1, \ldots, 1)$ and $M = 200$.

In this test example, we consider three categories of experiments.

(a) *Comparison of optimization methods.* We compare the genetic algorithm with the Nelder–Mead simplex search method according to Lagarias et al. [28], which is implemented in the `fminsearch` routine of MATLAB. For the genetic algorithm, we have selected $I_{\max} = [0, 1]$. The restriction set is $\Omega = [0, 2]^{20} \subset D$, where the



(a) Initial condition.



(b) Results with $M = 100$.

FIGURE 7. (Example 4.3) (a) Initial condition u_0, random best initial profile at initial evolution, identified profile and analytical profile $\hat{u}$ at $T = 3$; (b) comparison of the flux used for the observational profile and the identified flux. See Table 5 for numerical values of best initial and identified parameters.

restitution set D, given by (2.5), is defined as

$$D = \{\mathbf{e} = (e_1, \ldots, e_{20}) \in \mathbb{R}^{20} \mid e_i > 0 \text{ and } f(\cdot; \mathbf{e}) \in U_{\mathrm{ad}}\}.$$

For the Nelder–Mead simplex method, the initial guess was $0.9\,\mathbf{e}_\infty$, since, in other cases, the convergence turned out to be too slow, and the execution was aborted when the cpu time was higher than 12 hours. The values of the cost function obtained by the genetic algorithm and the Nelder–Mead simplex method after 200 iterations are 9.454×10^{-4} and 8.164×10^{-4}, respectively. The corresponding cpu time in the case of the genetic algorithm was 10.38 hours and for the Nelder–Mead simplex method it was 11.21 hours. The selection of the Nelder–Mead simplex method follows the suggestions of Hansen et al. [17]. Although the performances of both compared methods are on the same level by concept, the genetic algorithm is more robust and feasible when the cost function is nonconvex. In fact, in test runs, the Nelder–Mead simplex method evolved extremely slowly, and it almost got stuck whenever the initial data were not chosen close to the optimum.

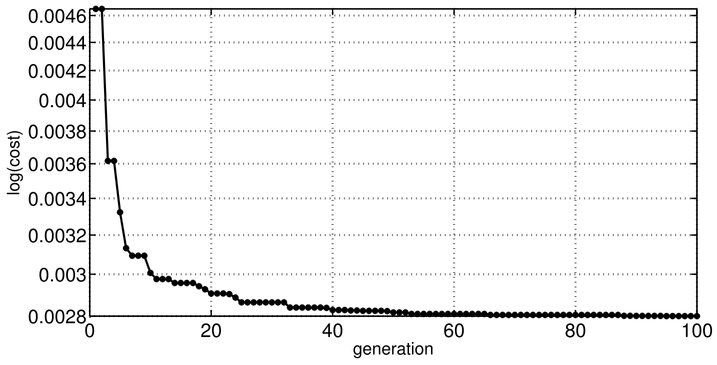
(b) *Convergence of the algorithm.* We study two types of convergence. First, we study the convergence in terms of the mesh refinement. In this case, we consider that the total number of iterations is fixed at 200 and develop the identification with mesh size $M_i = 10\,i$ for $i \in \{1, \dots, 20\}$. We find the best chromosomes in terms of the mesh size and we plot the space steps M_i versus the evaluation of the cost function of these chromosomes. The results are given in Figure 8(a). Second, we study the convergence at fixed space steps. We fix the temporal steps as $M = 50$, find the best chromosomes in the first 100 iterations and plot the generation versus the cost of the best chromosome (see Figure 8(b)).

(c) *Stability with respect to noisy data on the observation.* In this experiment, we consider the identification of a random perturbation of the observational data. The considered random perturbations are uniformly distributed numbers in the interval $(-0.05\,\overline{u}, 0.05\,\overline{u})$, where $\overline{u}$ is the average of the observed data. The identification with the genetic algorithm with $M = 200$ spatial nodes appears to be stable. The results are shown in Figure 9.

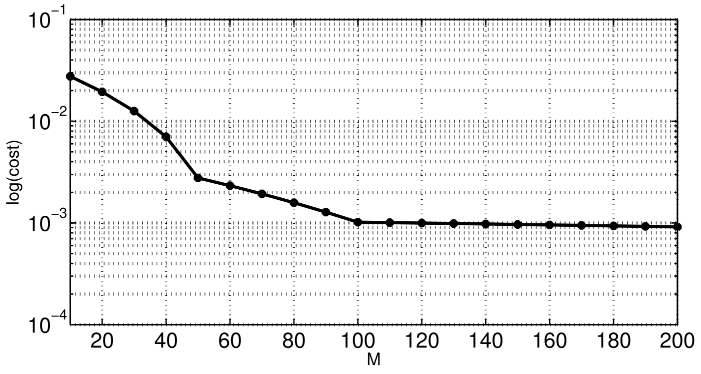
5. Conclusion and discussion

The parameter identification problem for scalar conservation laws is formulated as an optimization problem and solved by a continuous genetic algorithm. The continuous genetic algorithm has been applied to four numerical examples. The agreement between the observed parameters and the parameters identified by the continuous genetic algorithm are of high quality. Furthermore, we observe that the optimum of the numerical cost function depends on the discretization of the direct problem. Therefore, we believe that the application of the continuous genetic algorithm with a high resolution numerical method for the direct problem solution contributes naturally to several improvements to the numerical flux identification in scalar conservation laws.

The considered equations, namely, "scalar conservation laws" typically lead to low-dimensional optimization problems, since the number of parameter dimensions coincides with the number of parameters of the flux function. For performance demonstration, three examples with low dimensions $d \in \{1, 2, 3\}$ have been considered. For the types of equation considered, the dimension of the optimization problems in terms of free parameters is typically up to around five parameters. The limitation of parameter dimensions is caused by the fact that the shape of the flux function and its functional form is frequently restricted by constraints that are imposed by a specific application context. Typical constraints include the sign of the flux function, its derivatives and eventually even functional values as $f(0) = 0$. These constraints lead to specific parametric forms that are well established in the corresponding literature. An example is the S-shaped flux function in the Buckley–Leverett equation [1].



(a)



(b)

FIGURE 8. (Example 4.4) Measurements of two types of convergence: (a) the convergence in terms of the number of space steps at 200 iterations; (b) the generation versus the better evaluation of the cost function with 50 space steps.

Heuristic optimization techniques can be fruitfully applied to very high-dimensional optimization problems, where classical optimization techniques may not improve the results at all. On the other hand, in this study, the reason that justifies and actually motivates the use of heuristic optimization techniques is their ability to capture situations where the cost function is highly nonconvex, as is the case when the solutions of the direct problem count with jumps.

One can increase the dimension within the considered type of equations through an amplified modelling approach by considering systems of equations, that is, systems of conservation laws. (A typical application is for multiphase flow-like traffic. Systems of equations, however, need refined numerical techniques for the solution of the direct problem.) Another possibility for dimensional incrementation is to stay with scalar equations, but increase the order of the interpolating polynomial or spline to arbitrary height in order to obtain a best fit to the observational data. This situation has been considered in the last example for dimension $d = 20$.

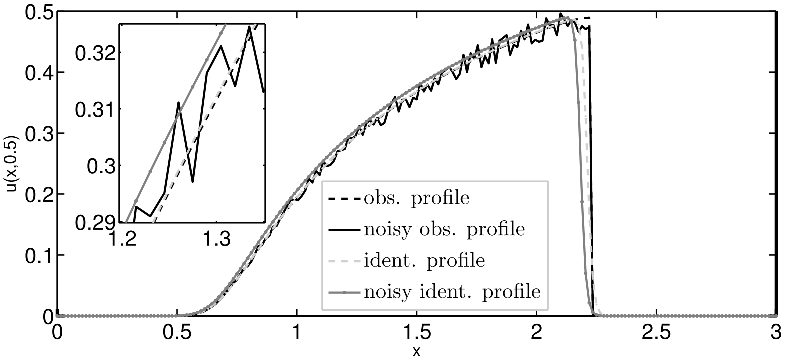


Figure 9. (Example 4.4) The observational data $\hat{u}$ (obs. profile) is randomly perpetuated and we obtain the noisy data (noisy obs. profile). The identification from $\hat{u}$ is given by the identification profile (ident. profile) and the identification from the noisy data is given by the noisy ident. profile.

The flux identification problem matters, since it helps to calibrate conservation law models describing various convective flow and transport problems occurring in chromatography, porous media flow or sedimentation. Corresponding mathematical models give a better understanding of the underlying physical processes and enable predictions. Model parameters are fitted to experimental observational data by parameter identification.

Therefore, robust optimization methods are needed. Classical methods have a low convergence rate, or they even fail to converge. Heuristic methods represent a promising alternative. In this contribution, we provide a series of test cases that vary over the number of parameters and application backgrounds. We assess the performance of the continuous genetic algorithm to solve the inverse problem. The main feature is its ability to handle nonconvex cost functions, which contributes to the robustness of the parameter identification procedure. In a future work, we plan to move towards a more ambitious use of the proposed solution technique in applications on the basis of the work presented here. In particular, the application of hybrid genetic algorithms to the calibration of conservation laws and their application to higher-dimensional optimization problems is a highly relevant subject for further investigation.

Acknowledgements

The first author S. Berres acknowledges the support of Conicyt (Chile) through Fondecyt project 1120587. The second author A. Coronel acknowledges the support of Conicyt (Chile) through Fondecyt project 11060400 and also thanks research projects DIUBB GI 153209/C and DIUBB GI 152920/EF at Universidad del Bío-Bío, Chile for their support. The author M. Sepúlveda thanks Fondecyt project 1140676, CONICYT project Anillo ACT1118 (ANANUM), Red Doctoral REDOC.CTA, project UCO1202 at Universidad de Concepción, BASAL project CMM, Universidad de Chile and CI^{2}MA, Universidad de Concepción for their support.

References

[1] S. Akin and B. Demiral, “Genetic algorithm for estimating multiphase flow functions from unsteady-state displacement experiments”, *Comput. Geosci.* **24** (1998) 251–258; doi:10.1016/S0098-3004(97)00129-5.

[2] S. Berres, R. Bürger, A. Coronel and M. Sepúlveda, “Numerical identification of parameters for a strongly degenerate convection-diffusion problem modelling centrifugation of flocculated suspensions”, *Appl. Numer. Math.* **52** (2005) 311–337; doi:10.1016/j.apnum.2004.08.002.

[3] R. Bürger, A. Coronel and M. Sepúlveda, “A numerical descent method for an inverse problem of a scalar conservation law modelling sedimentation”, in: *Numerical mathematics and advanced applications: numerical mathematics and advanced applications* (Springer, 2008) 225–232; doi:10.1007/978-3-540-69777-0_26.

[4] R. Bürger, A. Coronel and M. Sepúlveda, “Numerical solution of an inverse problem for a scalar conservation law modelling sedimentation”, in: *Hyperbolic problems: theory, numerics and applications*, Volume 67 of *Proceedings of Symposia in Applied Mathematics* (American Mathematical Society, Providence, RI, 2009) 445–454; doi:10.1090/psapm/067.2/2605240.

[5] R. Bürger and S. Diehl, “Convexity-preserving flux identification for scalar conservation laws modelling sedimentation”, *Inverse Problems* **29** (2013) 045008; doi:10.1088/0266-5611/29/4/045008.

[6] C. Castro and E. Zuazua, “Flux identification for 1-d scalar conservation laws in the presence of shocks”, *Math. Comput.* **80** (2011) 2025–2070; doi:10.1007/978-0-387-36797-2.

[7] C. A. Coello, G. L. Lamont and D. A. van Veldhuizen, *Evolutionary algorithms for solving multi-objective problems*, 2nd edn, *Genetic and Evolutionary Computation* (Springer, Berlin–Heidelberg, 2007); doi:10.1007/978-0-387-36797-2.

[8] T. J. Connolly and D. J. N. Wall, “On some inverse problems for a nonlinear transport equation”, *Inverse Problems* **13** (1997) 283–295; doi:10.1088/0266-5611/13/2/006.

[9] D. Constales, J. Kacur and R. Van Keer, “Parameter identification by a single injectionextraction well”, *Inverse Problems* **18** (2002) 1605–1620; doi:10.1088/0266-5611/18/6/312.

[10] A. Coronel, F. James and M. Sepúlveda, “Numerical identification of parameters for a model of sedimentation processes”, *Inverse Problems* **19** (2003) 951–972; doi:10.1088/0266-5611/19/4/311.

[11] C. M. Dafermos, *Hyperbolic conservation laws in continuum physics*, Volume 325 of *Grundlehren der Mathematischen Wissenschaften (Fundamental Principles of Mathematical Sciences)* (Springer, Berlin, 2010); doi:10.1007/978-3-642-04048-1.

[12] J. De Clerq, I. Nopens, J. Defrancq and Pa. Vanrolleghem, “Extending and calibrating a mechanistic hindered and compression settling model for activated sludge using in-depth batch experiments”, *Water Research* **42** (2008) 781–791; doi:10.1016/j.watres.2007.08.040.

[13] K. A. De Jong, “An analysis of the behaviour of a class of genetic adaptive systems”, Ph. D. Thesis, University of Michigan, 1975.

[14] R. Eymard, T. Gallouët and R. Herbin, *Finite volume methods*, Volume VII *Handbook of numerical analysis* (North-Holland, Amsterdam, 2000) 713–1020; doi:10.4249/scholarpedia.9835.

[15] L. J. Fogel, A. J. Owens and M. J. Walsh, *Artificial intelligence through simulated evolution* (Wiley, Chichester, 1966).

[16] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*, 1st edn (Addison-Wesley Longman Publishing, Boston, MA, 1989).

[17] N. Hansen, A. Auger, R. Ros, S. Finck and P. Pošík, “Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009”, in: *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation, GECCO ’10* (New York, 2010) 1689–1696; doi:10.1145/1830761.1830790.

[18] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms*, 2nd edn (Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, 2004); doi:10.1002/0471671746.

[19] H. Holden, F. S. Priuli and N. H. Risebro, “On an inverse problem for scalar conservation laws”, *Inverse Problems* **30** (2014) 035015; doi:10.1088/0266-5611/30/3/035015.

[20] J. H. Holland, "Adaptation in Natural and Artificial Systems", (University of Michigan Press, Ann Arbor, MI, 1975); doi:10.1137/1018105.

[21] J. Hou, D.-G. Wang, F.-Q. Luo and Y.-H. Zhang, "A review on the numerical inversion methods of relative permeability curves", *Procedia Engineering (International Workshop on Information and Electronics Engineering)* **29** (2012) 375–380; doi:10.1016/j.proeng.2011.12.726.

[22] D. B. Ingham and S. D. Harris, *Parameter identification within a porous medium using genetic algorithms*, Volume VII *Handbook of Porous Media, Geothermal, Manufacturing, Combustion, and Bioconvection Applications in Porous Media* (CRC Press, 2005) 678–742; doi:10.1201/9780415876384.ch17.

[23] F. James and M. Sepúlveda, "Parameter identification for a model of chromatographic column", *Inverse Problems* **10** (1994) 1299–1314; doi:10.1088/0266-5611/10/6/008.

[24] F. James and M. Sepúlveda, "Convergence results for the flux identification in a scalar conservation law", *SIAM J. Control Optim.* **37** (1999) 869–891; doi:10.1137/S0363012996272722.

[25] F. James, M. Sepúlveda, F. Charton, I. Quiñones and G. Guiochon, "Determination of binary competitive equilibrium isotherms from the individual chromatographic band profiles", *Chem. Eng. Sci.* **54** (1999) 1677–1696; doi:10.1016/S0009-2509(98)00539-9.

[26] M. Jebalia, A. Auger, M. Schoenauer, F. James and M. Postel, "Identification of the isotherm function in chromatography using cma-es", in: *IEEE congress on evolutionary computation* (2007) 4289–4296; doi:10.1109/CEC.2007.4425031.

[27] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection* (MIT Press, Cambridge, MA, 1992); doi:10.1007/BF00175355.

[28] J. C. Lagarias, J. A. Reeds, M. H. Wright and P. E. Wright, "Convergence properties of the Nelder–Mead simplex method in low dimensions", *SIAM J. Optim.* **9** (1998) 112–147; doi:10.1137/S1052623496303470.

[29] R. J. LeVeque, *Numerical methods for conservation laws*, 2nd edn *Lectures in Mathematics ETH Zürich* (Birkhäuser, Basel, 1992); doi:10.1007/978-3-0348-8629-1.

[30] M. J. Lighthill and G. B. Whitham, "On kinematic waves. II. A theory of traffic flow on long crowded roads", *Proc. R. Soc. Lond. Ser.* A **229** (1955) 317–345; doi:10.1098/rspa.1955.0089.

[31] H. Liu and T. Pan, "Interaction of elementary waves for scalar conservation laws on a bounded domain", *Math. Methods Appl. Sci.* **26** (2003) 619–632; doi:10.1002/mma.370.

[32] Z. Michalewicz, *Genetic algorithms + data structures = evolution programs*, *Artificial Intelligence* (Springer, Berlin, 1992); doi:10.1007/978-3-662-03315-9.

[33] A. Mikelic and Z. Tutek., "Identification of mobilities for the Buckley–Leverett equation", *Inverse Problems* **6** (1990) 767; doi:10.1088/0266-5611/6/5/007.

[34] I. Rechenberg, *Evolutionstrategie: Optimierung Technischer Systeme nach Prinzipien des Biologischen Evolution* (Frommann-Holzboog, Stuttgart, 1973); doi:10.1002/fedr.19750860506.

[35] O. Redlich and D. L. Peterson, "A useful adsorption isotherm", *J. Phys. Chem.* **63** (1959) 1024; doi:10.1021/j150576a611.

[36] H. K. Rhee, R. Aris and N. R. Amundson, "First-order partial differential equations", in: *Theory and application of hyperbolic systems of quasilinear equations*, Volume II *Prentice Hall Int. Ser. in Physical and Chemical Engineering Sciences* (Prentice Hall, Englewood Cliffs, NJ, 1989).

[37] P. Rocca, M. Benedetti, M. Donelli, D. Franceschini and A. Massa, "Evolutionary optimization as applied to inverse scattering problems", *Inverse Problems* **25** (2009) 123003; doi:10.1088/0266-5611/25/12/123003.

[38] J. Schneider and S. Kirkpatrick, *Stochastic optimization*, *Scientific Computation* (Springer, 2007); doi:10.1007/978-3-540-34560-2.

[39] S. N. Sivanandam and S. N. Deepa, *Introduction to genetic algorithms* (Springer, Berlin, 2008); doi:10.1007/978-3-540-73190-0.

[40] H.-W. Tang, X.-K. Xin, W.-H Dai and Y. Xiao, "Parameter identification for modeling river network using a genetic algorithm", *J. Hydrodynamics, Ser.* B **22** (2010) 246–253; doi:10.1016/S1001-6058(09)60051-2.

[41] E. F. Toro, *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*, 3rd edn (Springer, Berlin, 2009); doi:10.1007/b79761.

[42] S. P. Usher, L. J. Studer, R. C. Wall and P. J. Scales, “Characterisation of dewaterability from equilibrium and transient centrifugation test data”, *Chem. Eng. Sci.* **93** (2013) 277–291; doi:10.1016/j.ces.2013.02.026.