Ivorra et al. *Journal of Mathematics in Industry* (2018) 8:4
https://doi.org/10.1186/s13362-018-0046-3

Journal of Mathematics in Industry
a SpringerOpen Journal

RESEARCH Open Access

Modelling and optimization applied to the design of fast hydrodynamic focusing microfluidic mixer for protein folding

Benjamin Ivorra[1*], Miriam R. Ferrández[2], María Crespo[3], Juana L. Redondo[2], Pilar M. Ortigosa[2], Juan G. Santiago[4] and Ángel M. Ramos[1]

*Correspondence:
ivorra@mat.ucm.es
[1] Instituto de Matemática Interdisciplinar (IMI), Grupo MOMAT, Dept. de Análisis Matemático – Matemática Aplicada, Universidad Complutense de Madrid, Madrid, Spain
Full list of author information is available at the end of the article

Abstract

In this work, we consider a microfluidic mixer that uses hydrodynamic diffusion stream to induce the beginning of the folding process of a certain protein. To perform these molecular changes, the concentration of the denaturant, which is introduced into the mixer together with the protein, has to be diminished until a given value in a short period of time, known as mixing time. In this context, this article is devoted to optimize the design of the mixer, focusing on its shape and its flow parameters with the aim of minimizing its mixing time. First, we describe the involved physical phenomena through a mathematical model that allows us to obtain the mixing time for a considered device. Then, we formulate an optimization problem considering the mixing time as the objective function and detailing the design parameters related to the shape and the flow of the mixer. For dealing with this problem, we propose an enhanced optimization algorithm based on the hybridization of two techniques: a genetic algorithm as a core method and a multi-layer line search methodology based on the secant, which aims to improve the initialization of the core method. More precisely, in our hybrid approach, the core optimization is implemented as a sub-problem to be solved at each iteration of the multi-layer algorithm starting from the initial conditions that it provides. Before applying it to the mixer design problem, we validate this methodology by considering a set of benchmark problems and, then, compare its results to those obtained with other classical global optimization methods. As shown in the comparison, for the majority of those problems, our methodology needs fewer evaluations of the objective function, has higher success rates and is more accurate than the other considered algorithms. For those reasons, it has been selected for solving the computationally expensive problem of optimizing the mixer design. The obtained optimized device shows a great reduction in its mixing time with respect to the state-of-the-art mixers.

Keywords: Global optimization method; Meta-heuristic algorithms; Mathematical modelling; Microfluidic mixer design

1 Introduction

Proteins are bio-molecules composed of one or more long chains of amino acids. Protein folding refers to the processes by which these amino acids interact with each other and produce a well-defined three-dimensional structure, called folded protein, able to perform

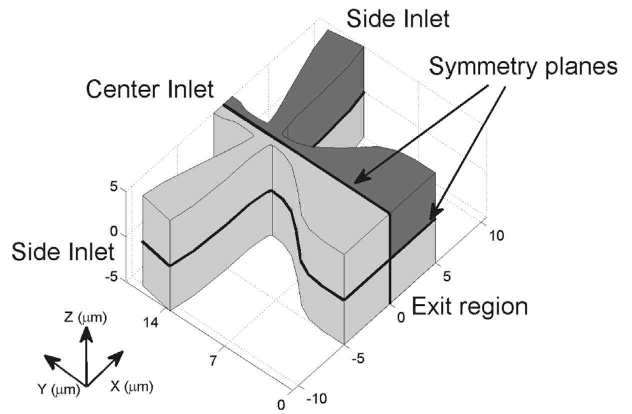

Figure 1 Schematic representation of the microfluidic mixer geometry. The region depicted in dark gray corresponds to the domain $\Omega_{3D,q}$. The wide solid black lines highlight the geometry's symmetry planes.

a wide range of biological functions [1]. The fundamental principles of protein folding have practical applications in genome research, drug discovery, molecular diagnostics or food engineering. Protein folding can be initiated, for instance, by inducing changes in chemical potential (e.g. changes in the concentration of a chemical specie).

The primary idea of a micromixer based on molecular diffusion across the fluid streamlines was proposed for the first time by Brody et al. in Ref. [2]. This kind of micromixer enables a fast and effective laminar mixing of unfolding proteins and a chemical denaturant, favoring the folding process. As illustrated in Fig. 1, the considered mixer consists of three inlet channels and a common outlet channel, being symmetric about its center channel. Unfolded proteins and chemical denaturant are injected through the inlet channel, while a background buffer is introduced through the two side inlet channels. The objective is to rapidly decrease the denaturant concentration to initiate protein folding in the outlet channel. Since the publication of Brody et al., various researchers have aimed to improve the micromixer performances [3–5], either by reducing the consumption rate of reactants or by minimizing the so called *mixing time*, i.e. the time required to attain a desired denaturant concentration (see Sect. 3 for a more detailed definition). For instance, the primary mixer of Brody et al. [2] exhibited mixing times larger than 10 μs, while Hertzog et al. [3] reported mixing times of 1.2 μs.

The aim of this work is to optimize the main design parameters of a particular hydrodynamic focused microfluidic mixer (mixer shape and flow injection velocities) for minimizing the mixing time of this device, taking into account that, till the date, the best mixer designs perform mixing times of approximately 1.0 μs [3].

To do so, we consider a general optimization problem of the form:

$$\min_{\phi\in\Phi} T(\phi) \tag{1}$$

where $T:\Omega\to\mathbb{R}$ is the cost function, ϕ is the optimization parameter and $\Phi\subset\mathbb{R}^N$, with $N\in\mathbb{N}$, is the search space. In order to choose a suitable methodology to solve Problem (1), we outline that in any iterative procedure, the determination of the initial condition is decisive, especially when T has various local minima. For example, gradient methods such as the Steepest Descent algorithm (SD) [6], may converge to different local minima of T depending on their initialization. Nevertheless, these algorithms can still end up with the global minimum if the initial choice is contained in its attraction basin. Initialization is also of vital importance for meta-heuristic methods such as Genetics Algorithms (**GA**) [7, 8], for which a lack of heterogeneity in the individuals of the initial population may yield to a early convergence to a local minimum of T [9]. From a general point of view, choosing suitable initial conditions can improve the efficiency of existing optimization algorithms by reducing the number of evaluations of the cost function, which is particularly worthy when dealing with expensive functional calculations, as it is the case in many industrial design problems [10–18].

There already exists several optimization algorithms which have been improved by choosing appropriate initialization techniques. To cite an instance, the Direct Tabu Search algorithm (DTS) [19, 20] relies on a modification of the cost function which adds penalty terms and prevent the algorithm to revisit previously investigated neighborhoods. Other methodologies, such as the Greedy Randomized Adaptive Search Procedure (GRASP), combine greedy solutions with a local search. Line search methods [6, 21] have been likewise modified by coupling them with other optimization algorithms. To give an example, in [22] the authors combined the Enhanced Unidirectional Search method [23] with the 3-2-3 line search scheme [24], and the resulting optimization method was proven to be satisfactory for solving high-dimensional continuous non-linear optimization problems. In the context of Multi-Objective optimization, we underline the approach presented in [25], a method composed of two line search algorithms: one establishes an initial choice in the Pareto front and the other one sets a suitable initial condition for exploring the front.

In this paper, we propose a meta-heuristic technique based on a specific **GA** combined with a line search method to dynamically upgrade its population. Our approach is validated through multiple test cases [26]. The results are then compared with those given by the following optimization algorithms: **SD**, **GA**, **DTS**, Continuous GRASP (**CGR**), a Controlled Random Search algorithm (**CRS**) [27], and a Differential Evolution algorithm (**DE**) [28].

This work is organized as follows. Section 2 introduces a mathematical model which computes the mixing time for a given mixer design (i.e., mixer geometry and flow injection velocities). In Sect. 3, we state the optimization problem which aims to minimize the mixing time by choosing a suitable mixer design. In Sect. 4, we describe the methodology used to solve the considered optimization problem. Finally, in Sect. 5, we present and discuss the results obtained during this work. First, we validate the optimization algorithm on various benchmark problems. Then, we detail and analyze the optimized microfluidic mixer.

2 Mathematical modelling

Let Ω_{3D} be the three-dimensional microfluididic hydrodynamic focusing mixer introduced in Sect. 1, and depicted in Fig. 1. With a view to reduce the simulation domain, we point out that the mixer geometry has two symmetry planes. Therefore, one only needs to study a quarter of the mixer, denoted by $\Omega_{3D,q}$ and represented in dark gray in Fig. 1. Furthermore, $\Omega_{3D,q}$ can be approximated considering a two-dimensional projection, as suggested in previous works [3, 16, 29]. A representation of this projection, denoted by $\Omega_{2D,q}$, is shown in Fig. 2.

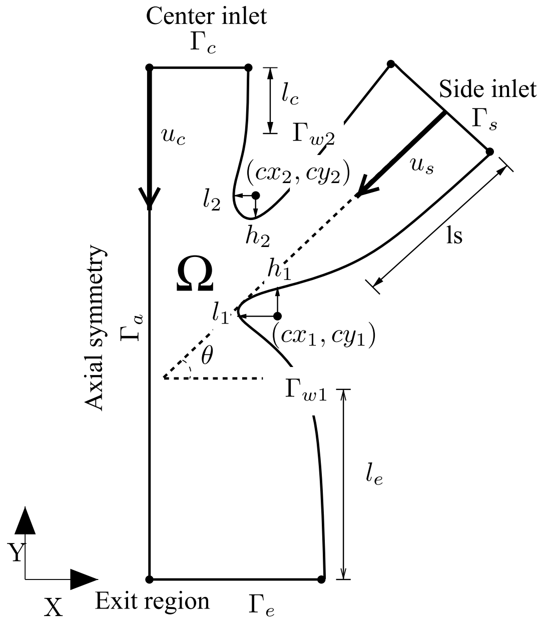

Figure 2 Domain $\Omega_{2D,q}$ and parametrization of the mixer considered when solving the optimization problem

In order to simplify the notations, we introduce $\Omega = \Omega_{2D,q}$. In the boundary of Ω, denoted by Γ, we define: Γ_c the boundary representing the center inlet; Γ_s the boundary representing the side inlet; Γ_e the boundary representing the outlet; Γ_{w1} the boundary representing the wall defining the lower corner; Γ_{w2} the boundary representing the wall defining the upper corner; Γ_a the boundary representing the Y-axis symmetry. A schematic representation of these boundaries is given in Fig. 2.

We assume that the liquid flow in the mixer is incompressible [4] and describe the concentration distribution of the denaturant by using the incompressible Navier–Stokes equations coupled with a convection-diffusion equation. Transient behavior of the device is not required, and, so, we only consider its configuration at stationary state. More specifically, we describe the flow velocity and the denaturant concentration distribution with the following system of equations [3, 4]:

$$\begin{cases} -\nabla \cdot (\eta(\nabla \mathbf{u} + (\nabla \mathbf{u})^\top) - \mathbf{I}p) + \rho(\mathbf{u} \cdot \nabla)\mathbf{u} = 0 & \text{in } \Omega, \\ \nabla \cdot \mathbf{u} = 0 & \text{in } \Omega, \\ \nabla \cdot (-D\nabla c) + \mathbf{u} \cdot \nabla c = 0 & \text{in } \Omega, \end{cases} \tag{2}$$

where c is the denaturant normalized concentration distribution, $\mathbf{u}$ is the flow velocity vector (m s^{-1}), p is the pressure field (Pa), D is the diffusion coefficient of the denaturant in the mixer (m^2 s^{-1}), η is the denaturant dynamic viscosity (kg m^{-1} s^{-1}), ρ is the denaturant density (kg m^{-3}) and $\mathbf{I}$ is the identity matrix.

System (2) is completed with the following boundary conditions:

For the flow velocity $\mathbf{u}$:

$$\begin{cases} \mathbf{u} = -u_c \mathrm{P}_c \mathbf{n} & \text{on } \Gamma_c, \\ \mathbf{u} = -u_s \mathrm{P}_s \mathbf{n} & \text{on } \Gamma_s, \\ (\eta(\nabla \mathbf{u} + (\nabla \mathbf{u})^\top) - \mathbf{I}p)\mathbf{n} = 0 & \text{on } \Gamma_e, \\ \mathbf{u} = 0 & \text{on } \Gamma_{w1} \cup \Gamma_{w2}, \\ \mathbf{n} \cdot \mathbf{u} = 0 \quad \text{and} \quad \mathbf{t} \cdot (\eta(\nabla \mathbf{u} + (\nabla \mathbf{u})^\top) - \mathbf{I}p)\mathbf{n} = 0 & \text{on } \Gamma_a, \end{cases} \tag{3}$$

where u_s and u_c are the maximum side and center channel injection velocities (m s^{-1}), respectively, with corresponding laminar flow profiles P_s and P_c (parabolas equal to 0 in the inlet border and unity in the inlet center); and $(\mathbf{t}, \mathbf{n})$ is the local orthonormal reference frame on the boundary.

For the concentration c:

$$\begin{cases} \mathbf{n} \cdot (-D\nabla c + c\mathbf{u}) = -c_0 \mathbf{u} & \text{on } \Gamma_c, \\ c = 0 & \text{on } \Gamma_s, \\ \mathbf{n} \cdot (-D\nabla c) = 0 & \text{on } \Gamma_e, \\ \mathbf{n} \cdot (-D\nabla c + c\mathbf{u}) = 0 & \text{on } \Gamma_{w1} \cup \Gamma_{w2} \cup \Gamma_a, \end{cases} \tag{4}$$

where $c_0 = 1$ is the initial denaturant normalized concentration in the center inlet. Notice that the first equation in (4) corresponds to the inward denaturant flux at the center inlet channel, while the third equation describes the convective flux leaving the outlet channel. This kind of boundary conditions, typically used for continuous flow systems [30], preserve the continuity of the denaturant concentration at the inlet and outlet boundaries.

3 Optimization problem

Here, we optimize the main design parameters (mixer shape and flow injection velocities) for minimizing the time required to attain a desired conversion of the denaturant concentration. This denaturant conversion, in turn, induces the folding process of the protein.

First, we specify the set of parameters defining a particular mixer design. Then, we introduce the optimization problem to be solved in Sect. 5.

We consider microfluidic mixers whose geometry can be described by rational Bézier curves and two ellipsoids (denoted as ellipsoids 1 and 2), similar to the one depicted in Fig. 2. Part of the ellipsoid 1 joins, in Γ_{w1}, the outlet and side channels while part of the ellipsoid 2 joins, in Γ_{w2}, the center and side channels. These curves are determined by the following parameters, suitably bounded to avoid non-admissible shapes (i.e., shape with intersected curves): the angle $\theta \in [0, \pi/3]$ between Γ_c and the direction normal to Γ_s; the length of the center inlet channel $l_c \in [2.5\ \mu\mathrm{m}, 5\ \mu\mathrm{m}]$; the length of the side inlet channel $l_s \in [1\ \mu\mathrm{m}, 9\ \mu\mathrm{m}]$; the length of the outlet channel $l_e \in [0.1\ \mu\mathrm{m}, 20\ \mu\mathrm{m}]$; the coordinates of the center of the ellipsoid i, with $i = 1, 2$, (cx_i, cy_i), where $cx_1 \in [0.8\ \mu\mathrm{m}, 3\ \mu\mathrm{m}]$, $cy_1 \in [l_e\ \mu\mathrm{m}, (l_e + 2)\ \mu\mathrm{m}]$, $cx_2 \in [0.8\ \mu\mathrm{m}, 0.9\ \mu\mathrm{m}]$ and $cy_2 \in [(cy_1 + 1)\ \mu\mathrm{m}, (cy_1 + 3)\ \mu\mathrm{m}]$; the radius l_i in the X-axis of the ellipsoid i, with $i = 1, 2$, satisfies $l_i \in [0\ \mu\mathrm{m}, (cx_i - 0.5)\ \mu\mathrm{m}]$; the radius h_i in the Y-axis of the ellipsoid i, with $i = 1, 2$, h_i satisfies $h_1 \in [0\ \mu\mathrm{m}, (cy_2 - cy_1 - 1)\ \mu\mathrm{m}]$ and $h_2 \in [0\ \mu\mathrm{m}, (cy_2 - cy_1 - 1 - h_1)\ \mu\mathrm{m}]$. Besides those parameters, we also consider the maximum injection velocities u_s and u_c as design variables. Furthermore, in order to preserve the laminar regime and to avoid additional flows at the outlet channel (such as Dean vortices), we constrained the typical flow Reynolds number Re to be less than 15. It is defined as $Re = \rho u_s L/\eta$, where $L = 3\ \mu\text{m}$ is the side channel nozzle width. This imposed bound implies that $u_s \leq \eta Re/\rho L$ m s^{-1}. Moreover, in real application, u_s should be at least 10 times faster than u_c to ensure a good mixing between fluids [3]. Therefore, we impose that $u_s \in [0, \eta Re/\rho L]$ m s^{-1} and $u_c = \sigma \times u_s$, where $\sigma \in [0.001, 0.1]$. Thus, the set of parameters defining a particular mixer design is denoted by

$$\phi = \{u_s, \sigma, \theta, l_c, l_s, l_e, cx_1, cy_1, l_1, h_1, cx_2, cy_2, l_2, h_2\} \in \Phi,$$

where $\Phi = \prod_{i=1}^{14} [\underline{\Phi}(i), \overline{\Phi}(i)] \subset \mathbb{R}^{14}$ is the admissible space; $\underline{\Phi}(i) \in \mathbb{R}$ and $\overline{\Phi}(i) \in \mathbb{R}$ are the upper and lower constraint values of the ith parameter in ϕ described previously.

We state the following optimization problem

$$\min_{\phi \in \Phi} T(\phi), \tag{5}$$

where $T(\phi)$ (μs), usually referred as *mixing time* [3, 4, 14], is defined as the time required to change the denaturant normalized concentration of a typical Lagrangian stream fluid particle located at the symmetry streamline at depth $z = 0$ (midway between the bottom and the top walls) from $\alpha \in [0, 1]$ to $\omega \in [0, 1]$, when considering the particular mixer described by $\phi \in \Phi$. It is computed as

$$T(\phi) = \int_{c_\omega^\phi}^{c_\alpha^\phi} \frac{dy}{\mathbf{u}^\phi(y)}, \tag{6}$$

where $\mathbf{u}^\phi$ and c^ϕ denote the solution of System (2)–(4), when the mixer defined by ϕ is considered; c_α^ϕ and c_ω^ϕ denote the points located along the symmetry streamline where the denaturant normalized concentration is α and ω, respectively.

In order to solve numerically Problem (5), we have considered the numerical implementation detailed below.

The solution of System (2)–(4) was computed numerically by using the software Matlab (www.mathworks.com) and COMSOL Multiphysics 5.2a (www.comsol.com), the latter based on the Finite Element Method (FEM). More specifically, we considered Lagrange P2-P1 elements to stabilize the pressure and to accomplish the Ladyzhenskaya, Babouska and Brezzi stability condition. The 2nd-order Lagrange elements model the flow field and concentration components, while linear elements stand for the pressure. The Navier–Stokes equations were solved using Galerkin Least Square streamline and crosswind diffusion approaches with a view to avoid numerical oscillations. The convection-diffusion equation was solved by using an upwind scheme. A Direct Dampled Newton method was applied to solve the associated linear systems. Finally, when computing the mixing time (defined by Equation (6)), we made use of the solutions of the previous FEM model and a trapezoidal approximation of the integral. We refer to the book [31] for a complete description of those techniques.

The model parameters considered during this work are associated to the denaturant guanidine hydrochloride (GdCl), a Chaotropic agent which is commonly used for protein folding. Its thermo-physical coefficients can be approximated as those of water, that is, its density is $\rho = 1010$ kg m^{-3} and its dynamic viscosity is $\eta = 9.8 \times 10^{-4}$ kg m^{-1} s^{-1}. In addition, the diffusion coefficient of GdCl in the background buffer (presumed to be similar to water) is $D = 2 \times 10^{-9}$ m^2 s^{-1}. Taking into account those coefficients and the constraint $Re = \rho vL/\eta \leq 15$ previously set, the maximum side injection velocity is $u_s \leq 7$ m s^{-1}. The values of α and ω in Equation (6) have been adapted to GdCl and taken here as 0.9 and 0.3, respectively. It has been empirically observed that a 3 times decrease of the GdCl concentration infers the folding process of several proteins (see for instance Ref. [3, 14]).

4 Optimization method

In this section, we describe in detail the optimization algorithm and the parameters used to solve Problem (5).

The considered optimization algorithm, called *Genetic Multi-Layer Algorithm* (**GMA**), consists on the combination of two methods: a genetic algorithm (**GA**) and a multi-layer secant algorithm (**MSA**). On the one hand, the **GA** [7] approximates the solution of (5) and, on the other hand, the **MSA** [32, 33] provides suitable initial populations for the **GA**. Both algorithms have been validated on several industrial problems in Refs. [13–15, 18, 33–35].

The **GA**s are meta-heuristics global optimization methods based on a set of points, known as the population of individuals, which evolve using some stochastic processes inspired from the Darwinian theory of species [7]. They are widely applied to solving many complex optimization problems due to the fact that they are intrinsically parallel and they can deal with large-scale problems. Moreover, regarding their search capacity, they exhibit good performances with functions presenting several local minima. Another advantage is that they do not require a-priori knowledge about the objective function neither sensitivity computation. However, it is important to point some of their weaker characteristics. Indeed, they are less accurate and have slower convergence than other algorithms as, for instance, those based on gradients. Before explaining the methodology used to enhance these inconveniences, we detail the **GA** used in this work.

For the considered **GA**, the user must set four parameters: the number of individuals in the population denoted by $N_p \in \mathbb{N}$, the number of generations $N_g \in \mathbb{N}$, the mutation probability $p_m \in [0,1]$ and the crossover probability $p_c \in [0,1]$. Additionally, the **GA** method needs an initial population X^0 of N_p individuals belonging to the search space Φ, i.e., $X^0 = \{x_j^0 \in \Phi, j = 1,\dots,N_p\}$.

The **GA** is mainly an iterative procedure for which each step $i = 0,\dots,N_{g-1}$ consists in generating a new population X^{i+1} from the previous population X^i. Each population $X^i = \{x_j^i \in \Phi, j = 1,\dots,N_p\}$ can be expressed as a $N_p \times N$ real-valued matrix, whose rows are occupied by an individual $x_j^i = (x_j^i(1),\dots,x_j^i(N)) \in \Phi$, given by:

$$X^i = \begin{bmatrix} x_1^i \\ \vdots \\ x_{N_p}^i \end{bmatrix} = \begin{bmatrix} x_1^i(1) & \dots & x_1^i(N) \\ \vdots & \ddots & \vdots \\ x_{N_p}^i(1) & \dots & x_{N_p}^i(N) \end{bmatrix}.$$

Then, the new population X^{i+1} is obtained by applying different stochastic mechanisms as selection, crossover, mutation and elitism as follows:

$$X^{i+1} = \left(I_N - \mathcal{E}^i\right)\left(\mathcal{C}^i \mathcal{S}^i X^i + \mathcal{M}^i\right) + \mathcal{E}^i X^i,$$

where these four mechanisms are represented by the matrices $\mathcal{S}^i$, $\mathcal{C}^i$, $\mathcal{M}^i$ and $\mathcal{E}^i$, respectively, and I_N denotes the N-dimensional identity matrix. Next, those processes are explained in detail, following their order of application.

1. *Selection*: The first one being implemented is the selection procedure consisting in randomly choosing N_p individuals among the N_p individuals belonging to the previous population X^i and considering that they can be repeated. Then, the intermediate population obtained after applying this first operator, denoted by $X^{i+1,1}$, is calculated as:

   $$X^{i+1,1} = \mathcal{S}^i X^i,$$

   where $\mathcal{S}^i$ is a (N_p, N_p)-matrix whose entries $\mathcal{S}^i_{j,k}$ are defined as:

   $$\mathcal{S}^i_{j,k} = \begin{cases} 1 & \text{if the } k\text{th individual of } X^i \text{ is the } j\text{th selected individual,} \\ 0 & \text{otherwise.} \end{cases}$$

   It is important to mention that each individual $x^i_j \in X^i$, with $j = 1, \ldots, N_p$, has a probability to be selected during this process which is given by $T^{-1}(x^i_j)/\sum_{k=1}^{N_p} T^{-1}(x^i_k)$.
2. *Crossover*: Once the selected population $X^{i+1,1}$ is obtained, its individuals are considered in consecutive pairs for the crossover, which carries out an exchange of data between them with a probability p_c. Analogously to the selection procedure, this operation can be expressed mathematically as:

   $$X^{i+1,2} = \mathcal{C}^i X^{i+1,1},$$

   where $\mathcal{C}^i$ is also a real-valued (N_p, N_p)-matrix. As said previously, the crossover process has to perform over each pair of consecutive individuals, i.e., the ones placed at the $2j - 1$ and $2j$ rows of $X^{i+1,1}$, with $j = 1, \ldots, \text{floor}(N_p/2)$. (Notice that floor$(y)$ is the function providing the nearest integer which is lower or equal than y). Therefore, the coefficients corresponding to those rows in the matrix $\mathcal{C}^i$ are given by:

   $$\mathcal{C}^i_{2j-1,2j-1} = \lambda_1, \qquad \mathcal{C}^i_{2j-1,2j} = 1 - \lambda_1, \qquad \mathcal{C}^i_{2j,2j} = \lambda_2, \qquad \mathcal{C}^i_{2j,2j-1} = 1 - \lambda_2,$$

   where
   - with a probability p_c, the considered rows exchange data, thus, λ_1 and λ_2 are randomly chosen in $]0, 1[$, considering a uniform distribution;
   - in the other case, with a probability $1 - p_c$, the considered individuals are maintained in the next intermediate population $X^{i+1,2}$ without any changes, i.e., $\lambda_1 = \lambda_2 = 1$.

   Then, the remaining coefficients of $\mathcal{C}^i$ are set to 0. Finally, if N_p is odd then the last element of the population $X^{i+1,1}$ does not have a partner, so it is directly copied in $X^{i+1,2}$ by setting $\mathcal{C}^i_{N_p,N_p} = 1$.
3. *Mutation*: Unlike the previous processes, this mechanism operates individually on each row of $X^{i+1,2}$. More specifically, the mutation procedure consists in randomly

perturbing, with a probability p_m, the individual corresponding to a row of $X^{i+1,2}$. It can be written as

$$X^{i+1,3} = X^{i+1,2} + \mathcal{M}^i,$$

where the real-valued (N_p,N)-matrix $\mathcal{M}^i$ has zeros at the jth row, with $j = 1,\ldots,N_p$, with a probability $1 - p_m$, or it has a vector $m_j \in \mathbb{R}^N$, randomly generated considering a uniform distribution in the subset of $\mathbb{R}^N$ such that $x_j^{i+1,2} + m_j \in \Phi$, otherwise.

4. *Elitism*: After applying all the four previous mechanisms, the majority of the individuals in the new population $X^{i+1,3}$ may be different from the ones in the previous population X^i. Then, it could happen that the individual x_b^i, where $b \in \{1,\ldots,N_p\}$, having the best value of the objective function T in X^i is no longer included in $X^{i+1,3}$. Furthermore, it may occur that none of the individuals in $X^{i+1,3}$ enhance the value $T(x_b^i)$. The purpose of the elitism procedure is copying the individual x_b^i into the population X^{i+1}, more specifically into its bth row. If there are several individuals with the best value for T, one of them is randomly selected for being maintained. Using matrix operations, the elitism process can be formulated as follows:

$$X^{i+1} = \left(I_N - \mathcal{E}^i\right)\left(X^{i+1,3}\right) + \mathcal{E}^i X^i,$$

where I_N denotes the N-dimensional identity matrix and the real-valued matrix $\mathcal{E}^i$ of size (N_p,N_p) has all the coefficients equal to zero except for the entry $\mathcal{E}^i_{b,b} = 1$, when the individual x_b has better value for T than all the individuals in $X^{i+1,3}$.

Those four mechanisms are carried out at each generation $i = 0,\ldots,N_{g-1}$ for creating the population X^{i+1} from the population X^i. When the N_g iterations are completed, the **GA** stops and returns as output solution the individual who has the lowest value for the objective function T among all the individuals in all the populations considered during the whole evolving process, i.e.,

$$\begin{aligned} &GAO\left(X^0,N_p,N_g,p_m,p_c\right) \\ &\quad = \operatorname{argmin}\left\{T\left(x_j^i\right) | x_j^i \text{ is the } j\text{th row of } X^i, i = 1,\ldots,N_g, j = 1,\ldots,N_p\right). \end{aligned}$$

As said at the beginning of this section, in order to accelerate the convergence and improve the accuracy of the above-described **GA**, we combine it with the **MSA** described below to build a hybrid algorithm, called **GMA**. A general scheme of the **GMA** is shown in Algorithm 1. Notice that, in addition to the **GA** input parameters, the number $l_{\max} \in \mathbb{N}$ of iterations for the **MSA** must be provided by the user. At the beginning of the **GMA**, a first initial population, X_1^0, of N_p individuals, $x_{1,j}^0 \in \Phi$, $j = 1,\ldots,N_p$, is randomly generated using a uniform distribution. At each **GMA** iteration l, with $l = 1,\ldots,l_{\max}$, the **GA** is executed starting from the initial population X_l^0, during N_g generations and with crossover and mutation probabilities of p_c and p_m, respectively. At the end of the **GMA** iteration a new initial population for the **GA**, X_{l+1}^0, is calculated by considering a secant method between each element in X_l^0 and the optimal individual returned by the **GA**, denoted by o_l. For controlling that the new individuals fit into the search space Φ, the projection function

Algorithm 1 GMA ($l_{\max}, T, N_p, N_g, p_c, p_m$)

```
Set $l = 1$, $X_1^0$ = initial population of $N_p$ randomly generated individuals
while $l < l_{\max}$ do
    Calculate $\mathbf{o}_l = \text{GA}(X_l^0, N_p, N_g, p_c, p_m)$        ▷ Running Genetic Algorithm
    for $j = 1, \dots, N_p$ do
            ▷ Construction of the next initial population $X_{l+1}^0 = \{x_{l+1,1}^0, \dots, x_{l+1,N_p}^0\}$
        if $T(o_l) = T(x_{l,j}^0)$ then
            $x_{l+1,j}^0 = x_{l,j}^0$
        else
            $x_{l+1,j}^0 = \text{proj}_\Phi(x_{l,j}^0 - T(o_l)\frac{o_l - x_{l,j}^0}{T(o_l) - T(x_{l,j}^0)})$
    $l = l + 1$
return $\text{argmin}\{T(o_l) | l = 1, \dots, l_{\max}\}$
```

$\text{proj}_\Phi : \mathbb{R}^N \to \Phi$ defined as $\text{proj}_\Phi(x)(i) = \min(\max(x(i), \underline{\Phi}(i)), \overline{\Phi}(i))$, with $i = 1, \dots, N$, is also applied. We note that o_l is also introduced in X_{l+1}^0 by randomly replacing one individual of this population. After $l_{\max}$ iterations, the **GMA** algorithm returns a solution

$$GMAO(l_{\max}, N_p, N_g, p_m, p_c) = \text{argmin}\left\{T(o_l) | l = 1, \dots, l_{\max}\right\}.$$

Algorithm 1 intends to improve, individual by individual, the initial population of the **GA**. More precisely, for each individual in the initial population:

- If there is a significant evolution of the cost function between this individual and o_l, the secant method generates a new individual close to o_l that performs a refined search near the actual solution.
- Otherwise, the secant method creates a new individual far from o_l, to expand the exploration of the admissible space.

The hybrid algorithm **GMA** has been already tested for solving different optimization problems [13, 15, 35] and, according to several numerical experiments, it seems that it achieves good results consuming lower computational time than the **GA** used on its own. The **GMA** is included by the authors in the software *Global Optimization Platform* which is freely available for its download at http://www.mat.ucm.es/momat/software.htm.

5 Results and discussion

In Sect. 5.1, we first validate the **GMA** presented previously on benchmark problems. Then, in Sect. 5.2, we present and discuss the numerical results obtained when searching the optimal design of the microfluidic mixer.

5.1 Validation of the GMA

This subsection aims to validate the above-proposed optimization methodology by applying it to solve a set of benchmark problems. In particular, we consider the following set of box constrained optimization problems detailed in [26]: Branin (denoted by **Bra**), Eason (**Eas**), Goldstein–Price (**G-P**), Shubert (**Shu**), Hartmann with 3 (**Hm3**) and 6 (**Hm6**) variables, Rosenbrock with 2 (**Rb2**), 5 (**Rb5**) and 10 (**Rb10**) variables, Shekel with 4 variables and a number of objectives of 5 (**Sk5**), 7 (**Sk7**) and 10 (**Sk10**), and Zakharov with 5 (**Za5**) and 10 (**Za10**) variables. That particular list of benchmark problems is considered as a good enough representation of low dimensional optimization problems (i.e., ≈10 variables) because it illustrates a wide and diverse set of difficulties that can be found in real problems [19, 36].

The **GMA** is applied for solving numerically those selected benchmark problems using the following parameters: $l_{\max} = 1000$, $N_p = 10$, $N_g = 10$, $p_c = 0.55$, $p_m = 0.5$. Those parameters have been demonstrated to be suitable for solving similar optimization problems in [13, 16, 29, 37]. Moreover, when the **GMA** ends, its solution is improved by performing 10 iterations of the Steepest Descent (**SD**) algorithm, in which the descent step size ρ is determined using 10 iterations of a dichotomy method starting from $\rho_0 = 1$. This last layer of **SD** is carried out in order to enhance the accuracy of the final solution.

In order to validate the **GMA**, it is compared with the following meta-heuristic methods from the literature:

- **DTS**: The Direct Tabu Search algorithm is an improved algorithm which exploits the idea of not revisiting regions of the search space that have been already explored. For achieving that, some penalty terms are added to the objective function. In this work, we use the implementation detailed in [19] with the parameters recommended in there.
- **CGR**: The Continuous GRASP (CGR) algorithm is an enhanced version of the Greedy Randomized Adaptive Search Procedure (**GRASP**). The latter combines the construction of a greedy solution with a local search method. Their implementation, the employed parameters, and the considered results can be found in [36].
- **SD**: We use the well-known Steepest Descent algorithm based on the gradient starting from a random point in the search space Φ, with 3000 iterations and with a descent step size ρ determined by the mean of 10 iterations of a dichotomy method with initial condition $\rho_0 = 1$.
- **GA**: The Genetic Algorithm explained at Sect. 4 is run with the following settings: $N_g = 1000$, $N_p = 180$, $p_c = 0.45$ and $p_m = 0.15$, which are suggested in [14, 17].
- **CRS**: The Controlled Random Search algorithm is also considered for solving the selected benchmark problems. Their parameters have been set according to the values recommended in [38]. In particular, we consider 200 individuals in the population, as many trial points as the size of the problem, 3000 iterations as the maximum threshold and a rate of 0.55 for the success test.
- **DE**: Finally, the Differential Evolution method is run with the parameters prescribed for low dimensional optimization problems in [39]. More precisely, the crossover operator is set to rand/1/exp with 0.9 for the crossover probability and 0.5 for the mutation probability. Additionally, we consider a maximum number of 5000 iterations and the population size is set to 5 times the dimension of the benchmark problem.

As done for improving the **GMA** accuracy, at the end of the **GA**, **CRS** and **DE**, 10 iterations of the **SD** configured with the same above-detailed settings are performed.

Since the global minimum of those benchmark problems is known, we implement the following stopping criterion [36], which is based on the distance between this minimum point, denoted by h_0^*, and the current solution of the algorithm, denoted by $\tilde{h}_0$:

$$\left|h_0^* - \tilde{h}_0\right| \le \epsilon_1 \left|h_0^*\right| + \epsilon_2, \tag{7}$$

where $\epsilon_1 = 10^{-2}$ and $\epsilon_2 = 10^{-3}$ for the algorithms **DE**, **CRS**, **GA** and **GMA**, but $\epsilon_1 = 10^{-4}$ and $\epsilon_2 = 10^{-6}$ for the **SD**.

Moreover, we consider another complementary stopping criterion consisting in a maximum number of evaluations of the objective function. In this work, we set this maximum value to 50,000 for each performance because, according to the literature [12, 13, 16, 17], it is a high enough value. If such an algorithm consumes all the 50,000 evaluations and its solution does not satisfy the first stopping criterion (7), we consider that the algorithm has failed solving numerically the problem at hand.

In order to establish a fair comparison among the different algorithms, since they are all heuristic, we perform 100 runs of each of them for solving each benchmark problem. Then, we calculate the success rate of an optimization algorithm as the percentage of runs satisfying the stopping criterion (7) (see Table 1). Furthermore, taking only into account those successful performances, the number of evaluations of the objective function that consumes each algorithm for each considered optimization problem is averaged and reported in Table 2.

In view of those results, **GMA** shows a good performance in comparison to the other analyzed algorithms. In fact, according to Table 1, it has success rates similar to the ones of **GA** and better than **DTS**, **CRS** and **DE**. Moreover, as we can observe in Table 2, **GMA**

Table 1 Success rate (%) of the optimization algorithms when solving the considered benchmark problems (**Func**)

Func.	DTS	CGR	SD	GA	CRS	DE	GMA
Bra	100	100	100	100	100	100	100
Eas	82	100	0	100	100	100	100
G-P	100	100	53	100	100	100	100
Shu	92	100	25	100	100	100	100
Hm3	100	100	51	100	100	100	100
Hm6	83	100	48	100	90	51	100
Rb2	100	100	80	100	100	100	100
Rb5	85	100	74	96	87	97	92
Rb10	85	100	71	95	68	93	81
Sk5	57	100	16	97	17	74	96
Sk7	65	100	7	96	23	88	98
Sk10	52	100	0	96	10	93	97
Za5	100	100	100	100	100	100	100
Za10	100	100	100	100	100	100	100

Table 2 Average number (considering only the runs satisfying the stopping criterion (7)) of evaluations needed by the optimization algorithms to solve the considered benchmark problems (**Func**)

Func.	DTS	CGR	SD	GA	CRS	DE	GMA
Bra	212	10,090	251	1304	2953	2347	252
Eas	223	5093	–	40,125	2877	3851	3488
G-P	230	53	295	465	2429	1937	439
Shu	274	18,608	120	7748	9947	3049	1270
Hm3	438	1719	466	1119	1493	447	425
Hm6	1787	29,894	217	4418	2907	8456	1054
Rb2	254	23,544	2275	3918	6177	7952	1675
Rb5	1684	182,520	3465	43,604	7927	41,939	43,972
Rb10	9037	725,281	5096	44,557	43,822	44,156	44,828
Sk5	819	9274	229	37,328	5702	40,032	6991
Sk7	812	11,766	208	36,046	3618	3479	4619
Sk10	6828	17,612	–	40,217	3540	2386	1637
Za5	1003	12,467	268	24,988	5384	40,026	2674
Za10	4032	2,297,937	540	40,489	9004	40,031	20,719

needs a number of evaluations which is, in the majority of the cases, lower than the one required by **GA**, **CRS** and **DE**. Therefore, for those problems where the gradient of the objective function is not available or it is difficult to compute, the **GMA** can be a good alternative to classical evolutionary algorithms.

Finally, we analyze the level of improvement that achieves the **GMA** with respect to the **GA** used without the multi-layer method. For this aim, we define the following improvement threshold (in %) as:

$$\mathrm{Imp}(\mathbf{GMA}) = 100 \times \frac{\mathrm{Tev}(\mathbf{GA}) - \mathrm{Tev}(\mathbf{GMA})}{\mathrm{Tev}(\mathbf{GA})}, \tag{8}$$

where $\mathrm{Tev}(\mathbf{A})$ is the total number of evaluations needed by the algorithm **A** for solving all the benchmark problems. For its calculation, we also include those runs that are considered unsuccessful regarding the stopping criterion (7). Then, we obtain a value of $\mathrm{Imp}(\mathbf{GMA}) = 58\%$, which give us an idea of the computational effort reduction reached when using the **GMA** instead of the **GA**.

From those results, the **GMA** seems to enhance the convergence and to achieve a reduction of the computational effort associated with the number of functional evaluations. In addition, the **GMA** has been also applied for solving some industrial design problems, as the one concerning us, here, about the optimization of microfluidic mixers. In this kind of real-world problems, where the objective functions are frequently computationally expensive and exhibit several local minima, the improvements in convergence and the savings in evaluations are of vital importance.

5.2 Microfluidic mixer optimization results

In this subsection, we present the optimization results obtained when solving Problem (5) with the Genetic Multi-Layer Algorithm (**GMA**), and its parameters, presented in Sect. 4. The number of evaluations of T used by **GMA** was approximately 6000 and the optimization process took about 40 h (on a 3.6 Ghz I7 Intel Computer with 32 Gb of RAM).

We denote by ϕ^{opt} the result reported at the end of the optimization process. Its values are presented on Table 3. The geometry of the optimized mixer is plotted in Fig. 3. Its denaturant concentration distribution and the transient concentration of a particle in the central streamline is depicted in Fig. 4. The mixing time corresponding to the optimized mixer is $T(\phi^{\mathrm{opt}}) \approx 0.10\ \mu\mathrm{s}$, this time being 10 times lower than those achieved by prior mixer designs with the same model [3, 14] (in those works the mixing times were larger than 1 μs). This improvement could be attributed to three main factors:

- The width of the mixing region (i.e., the area where the center and side mixer channels intersect and both fluids are mainly mixed) reaches a minimum value of about 1.1 μm near $y = 16.5\ \mu$m. At that location, the maximum velocity comes up to 26 m s^{-1}, helping to accelerate the mixing time.
- The value of angle θ (about $\pi/5$ radians) between the center and side mixer channels, which was set to 0 in Refs. [3, 4, 14]).

Table 3 Value of the optimal parameters in ϕ^{opt}, solution of Problem (5)

Parameter	u_s	σ	θ	l_c	l_s	l_e	cx_1	cy_1	l_1	h_1	cx_2	cy_2	l_2	h_2
Value	5.2	7.3×10^{-3}	0.6	2.5	9.1	16.3	1.1	16.6	0.5	0.3	0.9	18.9	0.1	1.1

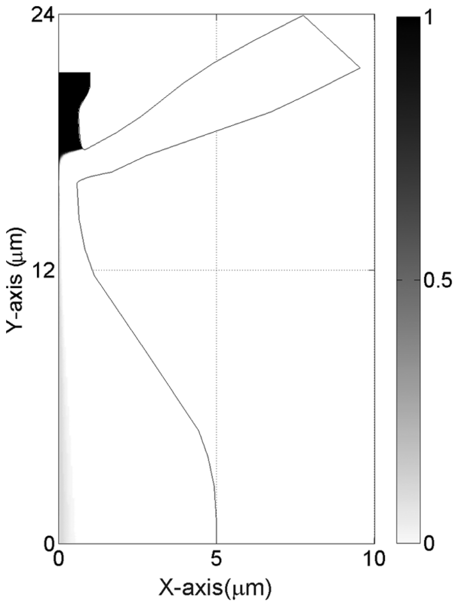

Figure 3 Shape of the optimized mixer and denaturant concentration distribution

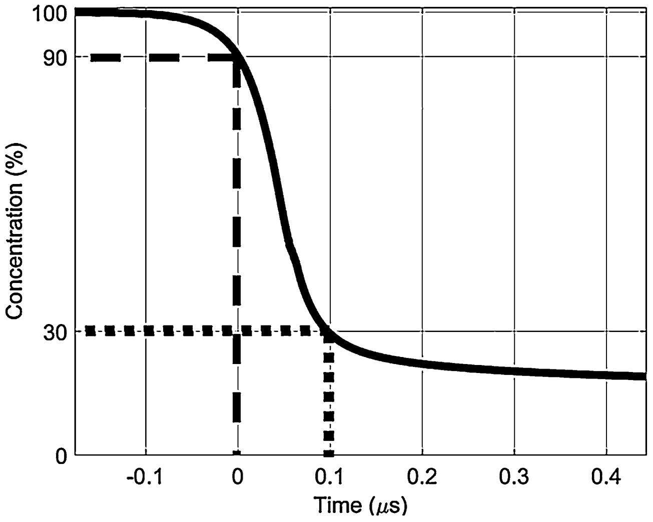

Figure 4 Transient behavior of the denaturant concentration of a particle in the mixer symmetry streamline

- The choice of suitable maximum injection velocities, set to $u_s = 5.2$ m s^{-1} and $u_c = 0.038$ m s^{-1}. In that case, the Reynolds number Re (defined in Sect. 3) is around 9, satisfying the constraint $Re < 15$ imposed to avoid secondary flows.

We recall that the admissible intervals for variables θ, u_s and u_c on the space Φ were chosen to be $[0, \pi/3]$, $[0, 7]$ m s^{-1} and $[5.2 \times 10^{-3}, 0.52]$ m s^{-1}, respectively. Thus, one can easily see that the optimized values of θ, u_c and u_s are included in the interior of these admissible intervals (and not on their boundary) which tends to show that the optimization process is not conditioned by the design constraints. The existence of such optimal values was also observed empirically in previous studies. More precisely, in Ref. [4] the authors showed that, for these mixers, there exists an optimal ratio between the side and center flows rates. For instance, if the flow is tuned too vigorously and the value of the injection velocities is constrained, then the area with significant diffusion ascends to the slow moving center stream, and, thus, the diffusive mixing takes place in a relatively low velocity region resulting in longer mixing times. Additionally, the analysis developed in Ref. [5] underlines the benefits of having inclined side channels (i.e., $\theta > 0$). Nevertheless, strong inclinations may cause centripetal accelerations of the fluid which bring secondary flows that worsen the mixing performance. On the contrary, slight inclinations may reduce these centripetal accelerations but make decrease the rate of stretching of material lines in the mixing region.

6 Conclusions

In this work, we dealt with the design of a microfluidic mixer based on hydrodynamic focusing for protein folding. More precisely, we aimed to find the shape parameters and the injection velocities that minimize the mixing time, which is the time needed for reducing the denaturant concentration under a desired level.

In order to compute the mixing time of a device attending to given design variables, we have presented a mathematical model consisting in the incompressible Navier–Stokes equations coupled with a convection-diffusion equation. This model was solved numerically using the Finite Element Method approximation over a simplified two-dimensional domain.

Then, we have formulated the mixer design optimization problem and we have proposed a methodology for solving it efficiently. This methodology, called **GMA**, is composed by a genetic algorithm (**GA**) and a multi-layer line search method for enhancing the **GA** initialization. Regarding the result, we have observed that applying it to some benchmark problems, the **GMA** improves the convergence of the **GA**. Moreover, the **GMA** consumes less number of evaluations of the objective function than the **GA**, achieving a reduction in the computational effort. Therefore, the **GMA** is strongly recommended for dealing with optimization problems whose evaluations of the objective function are computationally expensive, as our mixer design problem.

Finally, solving our mixer design problem with the **GMA**, we have found an optimized device which exhibits a mixing time of 0.1 μs. This new design implies a great reduction with respect to the previously developed mixers. In fact, according to the literature [3–5, 14, 40], the lowest mixing time among the previous similar mixers was 1 μs and it was achieved by the devices detailed in Refs [5, 14]. Therefore, the mixer designed with the parameters determined by our optimization methodology presents a mixing time of only 10% of those previous works. Analyzing the factors that can be responsible for this improvement, we have noticed two important novelties in the obtained design parameters: the angle of the side inlet channels and the inlet velocities. The angle parameter, with a value of $\pi/5$ radians, helps avoiding strong centripetal accelerations in the inlet side channel streams, an experimentally-observed phenomenon, explained in Ref. [5]. On the other hand, the inlet velocities have a great influence on the mixing time, but they were not optimized numerically in previous works. For instance, the values for those velocities were $u_s = 3.25$ m s^{-1} and $u_c = 0.032$ m s^{-1} in Ref. [3]. Here, we have obtained $u_s = 5.2$ m s^{-1} and $u_c = 0.038$ m s^{-1}.

For a deep sensitivity analysis of the mixing times regarding the optimized mixer parameters, we refer the interested reader to the Refs [16, 29].

Acknowledgements
Not applicable.

Funding
This work was carried out thanks to the financial support of the "Spanish Ministry of Economy and Competitiveness" under projects MTM2011-22658 and MTM2015-64865-P; the "Junta de Andalucía" and the European Regional Development Fund through project P12-TIC301; and the research group MOMAT (Ref.910480) supported by "Banco Santander" and "Universidad Complutense de Madrid".

Nomenclature
Bra, Branin benchmark optimization problem; **CGR**, Continuous Grasp optimization method; **CRS**, Controlled Random Search optimization method; **DE**, Differential Evolution optimization method; **DTS**, Direct Tabu Search optimization method; **Eas**, Eason benchmark optimization problem; **G-P**, Goldstein–Price benchmark optimization problem; **GA**, Genetic Algorithm optimization method; **GMA**, Genetic Multi-Layer Algorithm optimization method; **Hm3**, Hartman (with 3 variables) benchmark optimization problem; **Hm6**, Hartman (with 6 variables) benchmark optimization problem; **MSA**, Multi-Layer Secant Algorithm optimization method; **Rb10**, Rosenbrock (with 10 variables) benchmark optimization problem; **Rb2**, Rosenbrock (with 2 variables) benchmark optimization problem; **Rb5**, Rosenbrock (with 5 variables) benchmark optimization problem; **SD**, Steepest Descent optimization method; **Shu**, Shubert benchmark optimization problem; **Sk10**, Shekel (with 4 variables and 10 objectives) benchmark optimization problem; **Sk5**, Shekel (with 4 variables and 5 objectives) benchmark optimization problem; **Sk7**, Shekel (with 4 variables and 7 objectives) benchmark optimization problem; **Za5**, Zakharov (with 5 variables) benchmark optimization problem; **Za**, Zakharov (with 5 variables) benchmark optimization problem.

Availability of data and materials
The **GMA** can be freely downloaded at: http://www.mat.ucm.es/momat/software.htm. No other material is available.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All authors have contributed to this work. All authors read and approved the final manuscript.

Authors' information
Ivorra Benjamin, Miriam R. Ferrández, Maria Crespo and Angel M. Ramos are specialist in mathematical modelling and optimization methods. Juana L. Redondo and Pilar M. Ortigosa are specialist in global optimization algorithms. Juan G. Santiago is a specialist in chemical engineering processes and, in particular, microfluidic devices.

Author details
[1]Instituto de Matemática Interdisciplinar (IMI), Grupo MOMAT, Dept. de Análisis Matemático – Matemática Aplicada, Universidad Complutense de Madrid, Madrid, Spain. [2]Dept. de Arquitectura de Computadores y Electrónica, Universidad de Almeria, ceiA3, La Cañada de San Urbano, Spain. [3]Departamento de Matemática Aplicada, Ciencia e Ingeniería de los Materiales y Tecnología Electrónica, Universidad Rey Juan Carlos, Madrid, Spain. [4]Mechanical Engineering Dept., Stanford University, Stanford, USA.

Publisher's Note

Received: 4 December 2017 Accepted: 8 June 2018 Published online: 15 June 2018

References

1. Berg J, Tymoczko J, Stryer L. Biochemistry. 5th ed. New York: Freeman; 2002.
2. Brody J, Yager B, Goldstein R, Austin R. Biotechnology at low Reynolds numbers. Biophys J. 1996;71(6):3430–41.
3. Hertzog D, Ivorra B, Mohammadi B, Bakajin O, Santiago J. Optimization of a microfluidic mixer for studying protein folding kinetics. Anal Chem. 2006;78(13):4299–306.
4. Hertzog D, Michalet X, Jäger M, Kong X, Santiago J, Weiss S, et al. Femtomole mixer for microsecond kinetic studies of protein folding. Anal Chem. 2004;76(24):7169–78.
5. Yao S, Bakajin O. Improvements in mixing time and mixing uniformity in devices designed for studies of proteins folding kinetics. Anal Chem. 2007;79(1):5753–9.
6. Luenberger D, Ye Y. Linear and nonlinear programming. International series in operations research & management science. Berlin: Springer; 2008.
7. Goldberg DE. Genetic algorithms in search, optimization and machine learning. 1st ed. Boston: Addison-Wesley; 1989.
8. Gonçalves JF, de Magalhães Mendes JJ, Resende MGC. A hybrid genetic algorithm for the job shop scheduling problem. Eur J Oper Res. 2005;167(1):77–95.
9. Rocha M, Neves J. Preventing premature convergence to local optima in genetic algorithms via random offspring generation. In: Imam I, Kodratoff Y, El-Dessouki A, Ali M, editors. International conference on industrial, engineering and other applications of applied intelligent systems. Lecture notes in computer science. vol. 1611. Berlin: Springer; 1999. p. 127–36.
10. Carrasco M, Ivorra B, Ramos AM. A variance-expected compliance model for structural optimization. J Optim Theory Appl. 2012;152(1):136–51.

11. Carrasco M, Ivorra B, Ramos AM. Stochastic topology design optimization for continuous elastic materials. Comput Methods Appl Mech Eng. 2015;289:131–54.
12. Muyl F, Dumas L, Herbert V. Hybrid method for aerodynamic shape optimization in automotive industry. Comput Fluids. 2004;33(5):849–58.
13. Gomez S, Ivorra B, Ramos AM. Optimization of a pumping ship trajectory to clean oil contamination in the open sea. Math Comput Model. 2011;54(1):477–89.
14. Ivorra B, Mohammadi B, Santiago J, Hertzog D. Semi-deterministic and genetic algorithms for global optimization of microfluidic protein folding devices. Int J Numer Methods Eng. 2006;66(2):319–33.
15. Ivorra B, Mohammadi B, Ramos AM. Optimization strategies in credit portfolio management. J Glob Optim. 2009;43(2–3):415–27.
16. Ivorra B, Redondo JL, Santiago JG, Ortigosa PM, Ramos AM. Two- and three-dimensional modeling and optimization applied to the design of a fast hydrodynamic focusing microfluidic mixer for protein folding. Phys Fluids. 2013;25(3):032001.
17. Ivorra B, Mohammadi B, Ramos AM. Design of code division multiple access filters based on sampled fiber bragg grating by using global optimization algorithms. Optim Eng. 2014;15(3):677–95.
18. Ivorra B. Application of the laminar Navier–Stokes equations for solving 2D and 3D pathfinding problems with static and dynamic spatial constraints: implementation and validation in comsol multiphysics. J Sci Comput. 2018;74(2):1163–87.
19. Hedar AR, Fukushima M. Tabu search directed by direct search methods for nonlinear global optimization. Eur J Oper Res. 2006;170(2):329–49.
20. Lamghari A, Dimitrakopoulos R. A diversified tabu search approach for the open-pit mine production scheduling problem with metal uncertainty. Eur J Oper Res. 2012;222(3):642–52.
21. Vieira DAG, Lisboa AC. Line search methods with guaranteed asymptotical convergence to an improving local optimum of multimodal functions. Eur J Oper Res. 2014;235(1):38–46.
22. Gardeux V, Chelouah R, Siarry P, Glover F. Em323: a line search based algorithm for solving high-dimensional continuous non-linear optimization problems. Soft Comput. 2011;15(11):2275–85.
23. Gardeux V, Chelouah R, Siarry P, Glover F. Unidimensional search for solving continuous high-dimensional optimization problems. In: ISDA'09 – ninth international conference on intelligent systems design and applications, 2009. Los Alamitos: IEEE Comput. Soc.; 2009. p. 1096–101.
24. Glover F. The 3-2-3, stratified split and nested interval line search algorithms. In: Research report, OptTek systems. Boulder. 2010.
25. Grosan C, Abraham A. Hybrid line search for multiobjective optimization. In: Perrot R, Chapman B, Subhlok J, de Mello R, Yang L, editors. High. Lecture notes in computer science. vol. 4782. Berlin: Springer; 2007. p. 62–73.
26. Floudas C, Pardalos P. Handbook of test problems in local and global optimization. Norwell: Kluwer Academic; 1999.
27. Price WL. Global optimization by controlled random search. J Optim Theory Appl. 1983;40(3):333–48.
28. Price K, Storn RM, Lampinen JA. Differential evolution: a practical approach to global optimization (natural computing series). New York: Springer; 2005.
29. Ivorra B, Redondo JL, Ramos AM, Santiago JG. Design sensitivity and mixing uniformity of a micro-fluidic mixer. Phys Fluids. 2016;28(1):012005.
30. Danckwerts PV. Continuous flow systems. Chem Eng Sci. 1953;2(1):1–13.
31. Glowinski R, Neittaanmäki P. Partial differential equations: modelling and numerical simulation. Computational methods in applied sciences. Netherlands: Springer; 2008.
32. Debiane L, Ivorra B, Mohammadi B, Nicoud F, Poinsot T, Ern A, et al. A low-complexity global optimization algorithm for temperature and pollution control in flames with complex chemistry. Int J Comput Fluid Dyn. 2006;20(2):93–8.
33. Ivorra B, Ramos AM, Mohammadi B. Semideterministic global optimization method: application to a control problem of the Burgers equation. J Optim Theory Appl. 2007;135(3):549–61.
34. Isebe D, Azerad P, Bouchette F, Ivorra B, Mohammadi B. Shape optimization of geotextile tubes for sandy beach protection. Int J Numer Methods Eng. 2008;74(8):1262–77.
35. Ivorra B, Mohammadi D, Dumas L, Durand O, Redont P. Semi-deterministic vs. genetic algorithms for global optimization of multichannel optical filters. Int J Comput Sci Eng. 2006;2(3):170–8.
36. Hirsch M, Pardalos P, Resende M. Speeding up continuous GRASP. Eur J Oper Res. 2010;205(3):507–21.
37. Ivorra B. Optimisation globale semi-deterministe et applications industrielles. ANRT-grenoble. 2006.
38. Hendrix E, Ortigosa P, García I. On success rates for controlled random search. J Glob Optim. 2001;21(3):239–63.
39. Storn R, Price K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim. 1997;11(4):341–59.
40. Knight JB, Vishwanath A, Brody JP, Austin RH. Hydrodynamic focusing on a silicon chip: mixing nanoliters in microseconds. Phys Rev Lett. 1998;80(17):3863–6.