J Glob Optim (2010) 48:41–55
DOI 10.1007/s10898-010-9525-9

A genetic algorithm for a global optimization problem arising in the detection of gravitational waves

Daniela di Serafino · Susana Gomez · Leopoldo Milano · Filippo Riccio · Gerardo Toraldo

Received: 8 January 2010 / Accepted: 12 January 2010 / Published online: 30 January 2010

Abstract The detection of gravitational waves is a long-awaited event in modern physics and, to achieve this challenging goal, detectors with high sensitivity are being used or are under development. In order to extract gravitational signals emitted by coalescing binary systems of compact objects (neutron stars and/or black holes), from noisy data obtained by interferometric detectors, the matched filter technique is generally used. Its computational kernel is a box-constrained global optimization problem with many local solutions and a highly nonlinear and expensive objective function, whose derivatives are not available. To tackle this problem, we designed a real-coded genetic algorithm that exploits characteristic features of the problem itself; special attention was devoted to the choice of the initial population and of the recombination operator. Computational experiments showed that our algorithm is able to compute a reasonably accurate solution of the optimization problem, requiring a much smaller number of function evaluations than the grid search, which is generally used to solve this problem. Furthermore, the genetic algorithm largely outperforms other global optimization algorithms on significant instances of the problem.

Keywords Global optimization · Genetic algorithm · Detection of gravitational waves

D. di Serafino (✉) · F. Riccio
Department of Mathematics, Second University of Naples, Caserta, Italy
e-mail: daniela.diserafino@unina2.it

S. Gomez
Institute of Applied Mathematics, National University of Mexico, Mexico City, Mexico

L. Milano
Department of Physical Sciences, University of Naples Federico II, Naples, Italy

G. Toraldo
DIAAT, University of Naples Federico II, Portici, Italy

1 Introduction

The detection of gravitational waves (DGW) from astrophysical sources is a very challenging goal in modern physics. A direct evidence of the existence of such waves will provide a validation of Einstein's general relativity theory and will open a path toward a new view of the universe [1]. Networks of detectors have been recently deployed with unprecedented capabilities, but gravitational waves have not yet been observed because of many difficulties arising in the detection process. Among them, the weakness of the gravitational signal and the rarity of the events that produce such waves call for highly effective data analysis techniques to filter the detector data streams.

Coalescing binary systems of compact objects (neutron stars and/or black holes) are very promising sources of gravitational waves for ground-based laser interferometric detectors. In this case, the most widely used detection technique is the *matched filter*, which exploits the waveform of the signal and assumes that the instrumental noise is a stationary white or whitened Gaussian stochastic process (see [2] and the references therein). A crucial issue in this methodology is the solution of a box-constrained global optimization problem, which is hard to solve because of the strong nonlinearity of the objective function, the unavailability of its derivatives, the presence of many local maxima, and the high computational cost of its evaluation. Furthermore, the objective function is a stochastic process because of the presence of noise, and hence, for a given gravitational signal, its maximum value changes with a specific realization of the noise.

In the astrophysics community, this optimization problem is usually solved by applying the *grid search*, which evaluates the objective function in a suitable discrete set of points belonging to the feasible domain (the grid) [2]. This set can be built to ensure that the maximum over it satisfies certain accuracy requirements (see Sect. 2), but this implies a very large number of grid points, and hence of objective function evaluations. To reduce this number, hierarchical strategies based on the grid search have been proposed, where a coarse grid or another optimization approach is initially applied to identify "promising" sub-domains, and a fine grid is then used on the sub-domains to find a good approximation of the maximum [3–5]. However, such strategies might lead to disregard, in the first phase, a sub-domain containing the solution, thus increasing the probability of missing the signal even if it is present.[1]

We investigate the application of Genetic Algorithms (GAs) to the above global optimization problem, with the aim of reducing the computational cost with respect to the grid search. These algorithms have the advantage that a priori information on the problem can be easily incorporated in the design of genetic operators, allowing the solution of difficult problems (see, e.g., [6]). Specifically, we present a real-coded GA designed by taking into account characteristic features of the problem and analyze its performance vs. the grid search and other well-established global optimization algorithms. Computational experiments on representative instances of the optimization problem show the effectiveness of our GA.

2 Mathematical formulation of the DGW problem

The DGW problem basically consists in discovering if the output of the detector contains a gravitational signal or if it is just the noise background. In the presence of a signal, this output is generally modeled as

[1] The matched filter technique provides a solution to the DGW problem with a certain confidence, identified by the so-called probabilities of false alarm and detection (see Sect. 2).

$$x(t) = r(t) + h(t; \boldsymbol{\theta}),$$

where t is the time, $r(t)$ is the noise, $h(t; \boldsymbol{\theta})$ is the gravitational signal and $\boldsymbol{\theta}$ is a vector of parameters. We assume that $r(t)$ is strictly white noise, i.e. a wide-sense stationary Gaussian stochastic process with mean 0 and variance 1 [7]. We focus our attention on gravitational wave signals emitted by coalescing binary systems, which can be modeled as *chirp* signals [3], i.e.

$$h(t; \boldsymbol{\theta}) = Aa(t - t_0; m_1, m_2) \cos(\varphi(t - t_0; m_1, m_2) + \varphi_0),$$

where

$$\boldsymbol{\theta} = (A, \varphi_0, t_0, m_1, m_2),$$

with A, φ_0 and t_0 denoting the amplitude, the initial phase, and the arrival time of the signal, respectively, and m_1 and m_2 the masses of the coalescing binary system. The functions $a(t-t_0; m_1, m_2)$ and $\varphi(t-t_0; m_1, m_2)$ are expressed in the so-called second-order restricted post-Newtonian approximation [3]. We note that the vector of parameters $\boldsymbol{\theta}$ is unknown. In practice, the output of the detector is sampled with a certain time step, thus a segment of data is analyzed, which is an N-dimensional vector $\boldsymbol{x} = (x[0], \ldots, x[N-1])$; the corresponding sampled gravitational signal, if present, is an M-dimensional vector $\boldsymbol{h}(\boldsymbol{\theta}) = (h[0], \ldots, h[M-1])$, with $M < N$ (the dependence on $\boldsymbol{\theta}$ has been neglected for simplicity). In the following, we assume $\boldsymbol{\theta} = (A, \varphi_0, n_0, m_1, m_2)$, i.e. we substitute the arrival time t_0 with the index n_0 of the corresponding sample, where $n_0 \in \{0, \ldots, N - M\}$.

As noted in Sect. 1, a widely used technique for solving the DGW problem is the matched filter, which is an optimal linear filter for detecting signals of known shape in stationary Gaussian noise [2,8,9]. Its application requires the following steps:

1. correlating the output of the detector with a family of templates, consisting of chirp signals $\boldsymbol{h}(\boldsymbol{\theta})$, with $\boldsymbol{\theta}$ varying in a suitable manifold;
2. finding the maximum of the correlations with respect to all the parameters;
3. comparing this maximum with a suitable threshold to decide if the output of the detector contains a gravitational signal (a detection is announced if the maximum exceeds the threshold).

This procedure is based on the observation that the highest Signal-to-Noise Ratio (SNR)[2] of the filter output is obtained when the values of the parameters identifying the template are the same as in the signal, and that this SNR is equal to the maximum, with respect to the signal parameters, of the mean value of the correlation, which is a stochastic process because of the stochastic nature of the noise [2]. The choice of the threshold is related to the probability of false alarm, i.e. of stating a detection in the absence of a signal, and to the probability of detection, i.e. of stating a detection when the output contains a gravitational signal [11].

It can be shown that the maximization in step 2 can be performed with respect to only three parameters, i.e. the masses m_1 and m_2, and the index n_0; furthermore, it can be carried out separately for (m_1, m_2) and n_0 [2]. Thus, steps 1 and 2 of the matched filter can be formulated as the following global optimization problem:

$$\underset{(m_1, m_2) \in \Omega}{\text{maximize}} \quad F(m_1, m_2), \tag{1}$$

[2] We assume that the SNR is defined as $\sqrt{E^2/\sigma^2}$, where $E^2 = (2/f)\sum_i h[i]$, σ^2 is the power of the noise, assumed to be white Gaussian, f is the sampling frequency, and $h[i]$ are the samples of the signal. This definition is adopted by the LAL package [10], which has been used for the generation of test problems in our computational experiments (see Sect. 4).

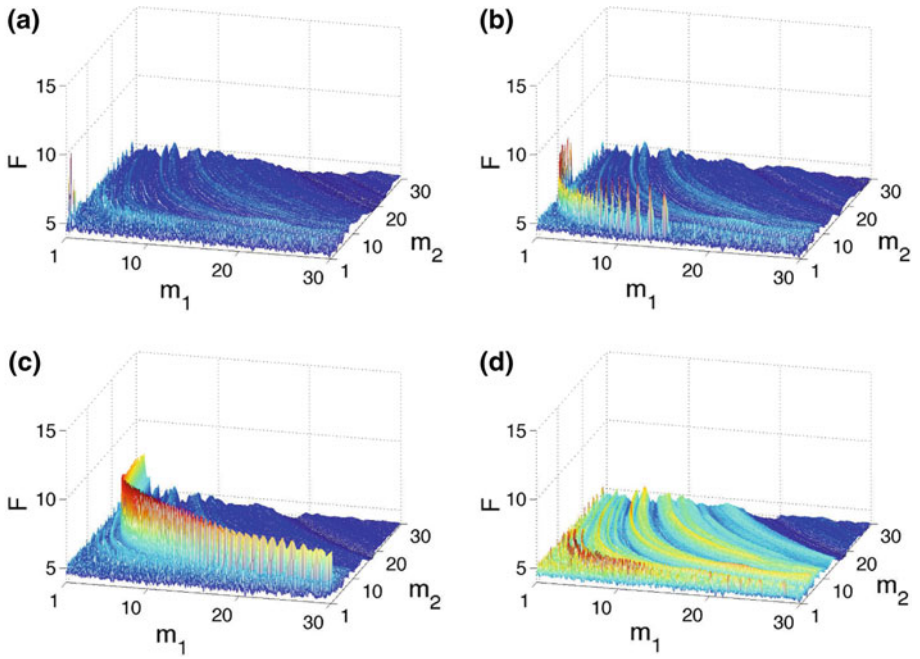

Fig. 1 3D plot of the objective function F, in case of noise plus gravitational signal from a binary system with masses $m_1 = m_2 = 1.4\ M_\odot$ (**a**), $m_1 = 1.4\ M_\odot$ and $m_2 = 10\ M_\odot$ (**b**), $m_1 = 5\ M_\odot$ and $m_2 = 10\ M_\odot$ (**c**), and in case of noise only (**d**). The SNR is equal to 10. $M_\odot$ denotes the solar mass

where

$$\Omega = \{(m_1, m_2) \in \Re^2 \ : \ l \leq m_1, m_2 \leq u\}, \tag{2}$$

$$F(m_1, m_2) = \sqrt{\max_{n_0 \in \{0,\ldots,N-M\}} \left(C_0^2(n_0, m_1, m_2) + C_{\pi/2}^2(n_0, m_1, m_2) \right)},$$

and $C_0(n_0, m_1, m_2)$ and $C_{\pi/2}(n_0, m_1, m_2)$ are the correlations between $\boldsymbol{x}$ and the normalized quadrature components of the template, $\hat{\boldsymbol{h}}_0(m_1, m_2)$ and $\hat{\boldsymbol{h}}_{\pi/2}(m_1, m_2)$ [11]:

$$C_0(n_0) = \sum_{k=n_0}^{n_0+M-1} x[k]\hat{h}_0[k - n_0], \quad C_{\pi/2}(n_0) = \sum_{k=n_0}^{n_0+M-1} x[k]\hat{h}_{\pi/2}[k - n_0]$$

(the dependence on m_1 and m_2 has been neglected for simplicity). We note that the maximum SNR is the maximum of the mean value of the objective function in (1); as shown in Sect. 4, this property can be used to assess the "accuracy" of the solution of problem (1) computed by using a selected algorithm.

Problem (1) is the most critical issue in the application of the matched filter. Its solution is a difficult task, because the objective function F is highly nonlinear and with many local maxima (see Fig. 1), and its derivatives are not available. Furthermore, its evaluation is computationally expensive, since it requires the solution of two ordinary differential equations (ODEs) to generate the quadrature components of each template [12], and the execution of three FFTs of length N to compute the correlations of $\boldsymbol{x}$ with them [3]. Common values of N are $O(10^5)$; the time for solving the ODEs depends on the masses of the gravitational signal

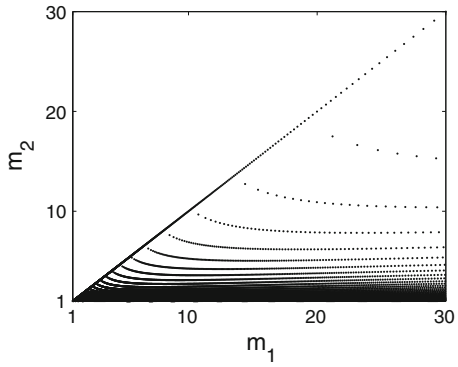

Fig. 2 Grid ensuring a minimal match of 97% over the domain $[1, 30] \times [1, 30]$ (27,379 points)

(the smaller the masses the larger the time), and is highly variable (from about 2% to 650% of the time for computing the correlations, in our experience). In the astrophysics community, the most widely used method for solving problem (1) is the grid search; it discretizes the feasible domain Ω by using a suitable grid of points and evaluates the objective function F at each point to determine an approximation of the global maximum. The search for the maximum can be carried out in half of the feasible domain, since F is symmetric with respect to m_1 and m_2. The main reason for choosing the grid search is that it provides information on the "accuracy" of the computed maximum. Indeed, the grid can be built by ensuring that the mean value of the maximum of F over it is not lower than a given percentage of the maximum SNR, called *minimal match* [2]. A minimal match of at least 97% is required, leading to a large number of grid points and hence of objective function evaluations; e.g., a grid of 27,379 points is needed to get a minimal match of 97% over the domain $[1, 30] \times [1, 30]$ (see Fig. 2). We also note that the grid is highly non-uniform, with more points in the regions where the objective function may have greater variability.

Reducing the computational cost in the solution of problem (1), while achieving a comparable accuracy in the mean value of the maximum, is a main goal in the application of the matched filter, since it increases the number of data segments that can be analyzed. In the next section we present a genetic algorithm developed to achieve this goal.

3 A genetic algorithm for the DGW problem

The genetic algorithms (GAs) are a class of evolutionary algorithms which apply the principles of natural evolution to find an optimal solution to a problem [13,14]. In a GA, an initial population of candidate solutions, called *individuals* or *chromosomes*, is randomly taken, so that coverage of the search space is assumed; three basic operators, *selection*, *recombination* (or *crossover*) and *mutation*, which mimic the corresponding natural processes, are iteratively applied to evolve the individuals toward a better population. A *fitness* function is used to measure the "goodness" of each individual. A solution to the problem is given by the fittest individual after the last evolution step. The basic structure of a GA is outlined in Fig. 3.

For numerical optimization problems on continuous domains of $\Re^n$, real-coded GAs, i.e. GAs where each individual is represented as a real n-dimensional vector having as genes the vector components [15], appear a more natural choice than binary-coded GAs. In this case, for problem (1), an individual is a pair of masses $\boldsymbol{m} = (m_1, m_2)$ and its genes are the single

initialize the population with random potential solutions
while (stopping criterion not satisfied)
    select parents
    recombine pairs of parents to generate offspring
    replace some parents with some offspring
    mutate the resulting population
end while

Fig. 3 Basic structure of a genetic algorithm

masses $m_i, i = 1, 2$. The way GAs perform strongly depends on the design of the genetic operators and their tuning to the specific problem under consideration. Next we describe the genetic operators as they have been designed in our algorithm. Before them, we discuss the choice of the initial population, which plays a fundamental role in the solution of our problem.

3.1 Initial population

Maaranen et al. recently provided numerical evidence that the initial population may strongly affect the speed of GAs and that a "good" initial population should combine *genetic diversity*, i.e. the ability to reach the whole feasible set during the evolution process, with *uniform coverage*, i.e. a spatial distribution in the feasible set which avoids clustering and uncovered regions [16]. On the other hand, in the GDW problem, a suitable choice of the templates for the matched filter technique leads to a nonuniform discretization of the feasible set, with points clustered in the areas where the objective function may show greater variability. This is an a priori specific information on the problem which can be included into the selection of the initial population. The most straightforward way to do this is to randomly select the individuals from a grid G corresponding to a widely accepted value of minimal match, such as 97%.

To foster a uniform coverage, we combined the previous strategy with a *nonaligned systematic sampling* (NSS), in which the feasible box Ω is splitted into b^2 elementary boxes with equal side lenghts, and one individual is selected in each elementary box according to some rule [16]. In our case, the individual is randomly chosen among the points of G belonging to the box; furthermore, the algorithm has been slightly modified to handle the (possible) case that a box does not contain any grid point. Given the size N_p of the population, i.e. the number of its individuals, the NSS is applied first, to select a part of the initial population; then, the remaining individuals are randomly taken from G. The parameter b is generally chosen to guarantee that no large areas of the feasible domain are left uncovered; $b = 0$ corresponds to an initial population randomly selected from G, whereas $b^2 = N_p$ to an initial population resulting only from the grid-based NSS. Because of the symmetry of the problem with respect to m_1 and m_2, only the triangle $m_1 \geq m_2$ of Ω, and the elementary boxes covering it, are actually considered.

The random selection of any individual from G is performed by labelling each point of G with an integer number from 1 to N_p and by choosing the point $q = 1 + int(rnd(0, 1) \cdot N_p)$, where $rnd(0, 1)$ is a random number from a uniform distribution in $(0, 1)$ and $int(x)$ is the integer part of x. The same rule is applied in each elementary box, considering only the points of G contained into the box.

3.2 Selection of parents

The purpose of the selection operator is to choose, from the current population, a mating pool of individuals that will potentially generate offspring through the recombination of their chromosomes. The selection of these individuals, called parents, is based on the principle of elitism: the individuals with higher fitness have higher probability to be picked. On the other hand, population diversity must be kept in order to avoid a premature convergence of the genetic algorithm, and therefore too much elitism in the selection might result in a serious drawback, especially when many local solutions exist. Several selection operators have been devised, such as proportional selection, tournament selection, rank-based selection and truncation selection [17,18]. These operators are characterized by the so-called *selective pressure*, which is related to the takeover time, i.e. the number of generations needed by the best individual in the initial population to fill up the whole population, by the application of the selection operator alone [17]. If the takeover time is large then the selective pressure is small, and vice versa.

For our problem, we choose the *binary tournament without replacement*, that has a medium selective pressure with respect to the other selection operators [18], and hence appears suitable for handling the existence of a large number of local solutions. Furthermore, the binary tournament does not require for the individuals in the population to be ranked. This operator randomly picks two individuals from the population and selects the one with better fitness as potential parent to be put into the mating pool; the picked individuals are removed from the population and the process is repeated again, until no individuals are available. This procedure is repeated twice, to have a number of parents equal to the size N_p of the population. In this way the best individual is selected at least twice and the worst one is discharged. We also note that the same individual can be present in the mating pool twice, depending on its fitness. The random selection of each individual is carried out with the rule described at the end of Sect. 3.1, where the current number of individuals is used instead of N_p at each step of the tournament.

3.3 Recombination

Once the mating pool is defined, pairs of individuals are randomly taken from it and mated. The number of actual parents depends on a parameter $P_R \in (0, 1)$, called *probability of recombination*; for each individual in the mating pool, a random number r from a uniform distribution in (0,1) is generated and, if $r < P_R$, the individual is selected as parent. A pair of parents is formed by two individuals consecutively selected.

As basic recombination operator we choose the *BLX-α* one, which is a well established and studied technique for real-coded genetic algorithms [15,19]. Each pair of parents $\boldsymbol{m}^m, \boldsymbol{m}^f$ generates three offspring, $\boldsymbol{m}^1, \boldsymbol{m}^2, \boldsymbol{m}^3$. The recombination is carried out separately on each gene by taking

$$m_i^j = rnd(I_i), \quad I_i = [g_i - \alpha M_i, G_i + \alpha M_i], \quad i = 1, 2, \quad j = 1, 2, 3, \tag{3}$$

with $g_i = \min\{m_i^m, m_i^f\}$, $G_i = \max\{m_i^m, m_i^f\}$, $M_i = G_i - g_i$, and $|\alpha| < 1$. I_i is referred to as action interval. We note that α is related to the size of the region around each parent, and thus its value controls the degree of "resemblance" to a parent; $\alpha > 0$ fosters *exploration*, i.e. the tendency to expand the search space, whereas $\alpha < 0$ fosters *exploitation*, i.e. the tendency to deepen the knowledge in areas of the search space already visited. For $\alpha = 0$ the flat recombination is obtained, in which m_i^j is randomly chosen between the corresponding

genes of its parents. We considered $\alpha = 0.5$, a choice allowing to balance exploration and exploitation [15], since the new gene has the same probability to lie inside or outside the interval defined by its parents.

However, taking $\alpha > 0$ might bring to an action interval which is not included in $[l, u]$, where l and u are defined in (2). In order to handle the box constraints, we devised two variants of the BLX-α strategy. In the former, if $I_i \not\subset [l, u]$, one considers as action interval the largest feasible interval $I_i^S \subset I_i$ obtained by symmetrically shrinking I_i, i.e.

$$I_i^S = [g_i - \bar{\alpha} M_i, G_i + \bar{\alpha} M_i],$$

where $\bar{\alpha}$ is the largest value such that I_i^S is feasible. This strategy is called *SBLX-α*. In the latter, a gene m_i^j generated according to (3), that does not belong to $[l, u]$, is replaced by its projection onto this interval. This strategy is called *PBLX-α*. We observe that SBLX-α is more conservative than PBLX-α since it works on a smaller action interval; furthermore, the closer is a parent gene to one of its bounds, the higher is the probability for the generated genes to be equal to the parent one.

3.4 Replacement of parents

In order to choose which individuals will survive, two main approaches can be adopted: the *overlapping-generation* model and the *nonoverlapping-generation* model [18]. In the former case, the parents and the offspring will compete with each other for survival; in the latter, all parents die at each generation and the offspring compete for survival.

The overlapping-model is more elitist, thus implying a loss of genetic diversity which is likely to lead to premature convergence to a local maximum. Because of the specific features of our problem, we decided to use a nonoverlapping-generation model, in which every pair of parents generates three offspring and the two offspring with better fitness survive. This simple model is combined with an elitist strategy guaranteeing that a copy of the best individual in the current population is forced to be selected into the new one. This individual is not replaced by its offspring.

3.5 Mutation

The mutation operator is aimed at randomly altering some individuals in the population, in order to introduce genetic diversity. We use the non-uniform mutation described in [14]. In this case, a gene m_i to be mutated becomes a new gene m_i^{new} according to the following formula:

$$m_i^{\text{new}} = \begin{cases} m_i + \Delta(k, u - m_i) & \text{if } r \geq 0.5, \\ m_i + \Delta(k, m_i - l) & \text{if } r < 0.5, \end{cases}$$

where r is a random number taken from a uniform distribution in $(0, 1)$ and $\Delta(k, y) = y(1 - r^{(1-k/N_G)^2})$, with k equal to the number of generations obtained so far and N_G to the maximum number of generations of the GA. This operator allows to explore the feasible domain uniformly in the first generations, and locally in the later generations.

The number of genes to be mutated depends on a parameter P_M called *probability of mutation*. For each gene of each individual, a random number r is taken from a uniform distribution in (0,1) and the gene is mutated if $r < P_M$. To avoid the best individual to be lost through the generations, we use an elitist strategy as in the replacement of the parents, i.e. we preserve a copy of the best individual by avoiding mutating it.

4 Computational experiments

Extensive computational experiments were carried out to evaluate the effectiveness of our GA in the solution of the DGW problem and its competitiveness with the grid search. Special attention was devoted to analyzing the effects of different choices of the initial population and of recombination strategies handling the box constraints, which are the most distinctive features of the algorithm. Further work was devoted to comparing the GA with other, well-established, global optimization algorithms.

We generated three sets of test problems, in which the detector output consists of strictly white noise and gravitational signal from three pair of masses, corresponding to three possible types of configurations of the coalescing binary system [2]:

- $m_1 = m_2 = 1.4 M_\odot$ (two neutron stars),
- $m_1 = 1.4 M_\odot$ and $m_2 = 10 M_\odot$ (one neutron star and one black hole),
- $m_1 = 5 M_\odot$ and $m_2 = 10 M_\odot$ (two black holes),

where $M_\odot$ denotes the solar mass. For each pair of masses we considered 30 realizations of noise, thus obtaining a set of 30 detector outputs to be analyzed. The length N of such outputs is 131,072, while the length M of the signal varies with the masses (51,207 for $m_1 = m_2 = 1.4$, 10,823 for $m_1 = 1.4$ and $m_2 = 10$ and 3,216 for $m_1 = 5$ and $m_2 = 10$). For all the problems, a SNR equal to 10 was chosen. The lower bound l and the upper bound u on the masses, defining the feasible domain, were set to 1 and 30, respectively. All the data were obtained by using the LAL package [10], which is gaining wide acceptance as a reference tool for gravitational wave data analysis.

We note that the most significant set of test problems is the one corresponding to $m_1 = m_2 = 1.4$, since binary systems of neutron stars are known to exist and, for some of them, general relativistic effects in the binary orbits have been accurately measured [20]. These problems are also the most difficult to solve, as shown by the results reported in this section. Furthermore, problems related to the same type of binary configuration show the same level of difficulty, therefore we do not consider other values for the pair of masses.

Our GA was implemented in the C language, in double precision, using the Mersenne twister pseudo-random number generator [21], as implemented in the GNU Scientific Library (version 1.11). For each set of test problems the GA was run using 30 different seeds for initializing the above generator. The algorithm was stopped when the maximum number of generations, N_G, was achieved. We note that we did not stop the algorithm as soon as the detection threshold was exceeded (see step 3 of the matched filter in Sect. 2), since in this case the computed maximum of the objective function may be very far from the actual one, thus providing poor information on the signal. On the other hand, the algorithm might not be able to compute a maximum exceeding the threshold. However, the threshold was used to evaluate the performance of the algorithm, as explained below. A threshold equal to 8 was chosen, which is a typical value in the DGW problem [22]. We also verified experimentally that other stopping criteria, e.g. based on the variation of the masses, may halt the algorithm prematurely. The GA parameters, i.e. the probabilities of recombination and mutation, the size of the initial population and the maximum number of generations, were set as follows: $P_R = 0.7$, $P_M = 0.05$, $N_P = 100$ and $N_G = 50$. These values were selected mainly on the basis of computational experiments, not reported here for the sake of space; the choice of the values for P_R and P_M was suggested also by the literature [23,24].

Our first experiments were aimed at studying the impact of the choice of the initial population on the GA behaviour. We compared three different strategies:

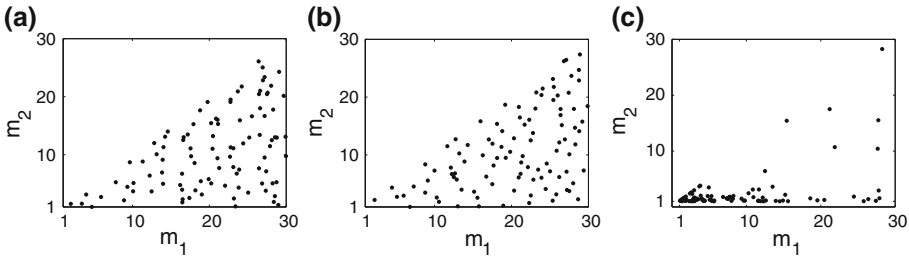

Fig. 4 Initial populations of 100 individuals generated by using the RAND (**a**), RAND-NSS (**b**) and RAND-GRID (**c**) strategies

Table 1 GA behaviour with different strategies for the selection of the initial population

m_1	m_2	Init. population	fmean	fstd	success (%)	relerr
1.4	1.4	RAND	6.8133	1.6225	20.9	0.322
		RAND-NSS	6.9053	1.6852	24.0	0.312
		RAND-GRID	9.8058	1.2655	92.0	0.024
1.4	10	RAND	9.6069	1.5763	87.9	0.065
		RAND-NSS	9.5351	1.5918	86.0	0.072
		RAND-GRID	10.2358	1.0716	99.0	0.004
5	10	RAND	10.2734	1.0383	99.0	0.001
		RAND-NSS	10.2767	1.0319	99.3	0.001
		RAND-GRID	10.1993	1.0448	98.7	0.008

SBLX-0.5 is used as recombination operator

– random generation of individuals from a uniform distribution in $[1, 30] \times [1, 30]$ (RAND);
– NSS with $b^2 = N_P$ elementary boxes, with a random choice of individuals from a uniform distribution in each box (RAND-NSS);
– combination of random choice of individuals from the grid corresponding to a minimal match of 97% and of grid-based NSS with $b = 4$, as explained in Sect. 3.1 (RAND-GRID).

Initial populations generated with these three strategies are shown in Fig. 4. Table 1 shows the numerical results obtained by running the GA with the three strategies, using SBLX-0.5 as recombination operator, for each set of test problems. The mean value of the computed maximum of the objective function F over 900 runs (30 realizations of noise $\times$ 30 seeds for the pseudo-random number generator) and the related standard deviation are reported in the *fmean* and *fstd* columns; the percentage of runs in which the maximum of F exceeds the selected threshold, and hence a signal detection is stated, is reported in the *success* column; finally, the absolute value of the difference between the mean value of the maximum of F computed by the grid search and that computed by the GA, divided by the first one, is reported in the *relerr* column (this also includes the runs where the maximum computed by the GA does not exceed the threshold). We recall that the reference value for the mean of the computed maximum of F is the SNR, i.e. 10.

As expected, a choice of the initial population which provides a uniform coverage of the feasible domain without taking into account the specific characteristics of the objective function (RAND-NSS) does not produce any significant improvement with respect to a uniform random choice of the population in the whole feasible domain (RAND). On the

Table 2 GA behaviour with the RAND-GRID strategy, varying the parameter b in the grid-based NSS, and with the SBLX-0.5 recombination operator

SBLX-0.5								
m_1	m_2	b	fmean	fstd	success (%)	relerr	evmean	evstd
1.4	1.4	0	9.8617	1.2057	93.7	0.018	4202	1404
		4	9.8058	1.2655	92.0	0.024	4206	1454
		8	9.6272	1.4666	87.2	0.041	4232	1465
1.4	10	0	10.2529	1.0399	99.4	0.003	3992	1476
		4	10.2358	1.0716	99.0	0.004	4046	1448
		8	10.1993	1.1088	98.3	0.008	4021	1522
5	10	0	10.1592	1.0421	98.8	0.012	3608	1589
		4	10.1993	1.0448	98.7	0.008	3577	1663
		8	10.2669	1.0361	98.9	0.002	3638	1615

other hand, a very strong improvement can be observed for $m_1 = m_2 = 1.4$ when the problem-driven approach (RAND-GRID) is adopted; indeed, for this set of test problems, neglecting the information provided by the grid leads to a mean value of the maximum of F that is very far from the SNR and exceeds the threshold in at most 24% of the runs. The problem-driven approach produces also a significant improvement in the case $m_1 = 1.4$ and $m_2 = 10$, whereas it does not produce any improvement the case $m_1 = 5$ and $m_2 = 10$. Similar results hold if PBLX-0.5 is used as recombination operator. However, we report that for $m_1 = m_2 = 1.4$ the use of PBLX-0.5 produces higher percentages of success when the RAND and RAND-NSS strategies are used (59.9% and 57.3%, respectively); this is due to the fact that the projection of a gene that is out of its bounds onto the interval with endpoints the corresponding genes of the parents produces more individuals close to (1.4, 1.4). The previous results show that a selection of the initial population based on the a priori knowledge of the problem is a key issue for the performance of the GA.

Taking into account the previous considerations, we performed a deeper analysis of the GA behaviour with the RAND-GRID strategy, varying the value of the parameter b in the grid-based NSS, and applying the recombination rules SBLX-0.5 and PBLX-0.5 described in Sect. 3.3. The corresponding results are reported in Table 2 for SBLX-0.5 and in Table 3 for PBLX-0.5. By looking at Table 2, we see that $b = 0$ and $b = 4$ lead to very close results for all the test sets; their behaviour is satisfactory, as shown by the mean value of the computed maximum, which, in our problem, can be considered very close to the mean value of the maximum over the grid (see the *relerr* column), and by the high percentage of success. We note that the lower percentage of success for $m_1 = m_2 = 1.4$ is also due to the fact that one of the 30 instances of this class of problems has a maximum value of F lower than the detection threshold (the maximum computed by the grid search algorithm is 7.02). Therefore, the success cannot exceed 96.7% in this case. The choice $b = 8$, in which more than one third of the population is generated by the grid-based NSS, degrades the GA performance for $m_1 = m_2 = 1.4$, while slightly improves it for $m_1 = 5$ and $m_2 = 10$. The previous comments apply also to the results in Table 3, concerning PBLX-0.5. However, we see that SBLX-0.5 leads to slightly greater mean values of the maximum of F; it generally gives also greater percentages of success (about 99%) for the problems with larger masses. This suggests that a more conservative strategy to handle the constraints should be preferred. It is worth noting that computational experiments with the BLX-0 operator, which never violates the box constraints, led to poor percentages of success.

Table 3 GA behaviour with the RAND-GRID strategy, varying the parameter b in the grid-based NSS, and with the PBLX-0.5 recombination operator

PBLX-0.5								
m_1	m_2	b	fmean	fstd	success (%)	relerr	evmean	evstd
1.4	1.4	0	9.7514	1.0979	93.6	0.029	3865	1568
		4	9.7036	1.1397	92.8	0.034	3827	1615
		8	9.5714	1.2943	88.4	0.047	3885	1616
1.4	10	0	10.1143	1.1139	96.8	0.016	3654	1660
		4	10.0773	1.1449	96.4	0.020	3579	1660
		8	10.0442	1.1221	96.4	0.023	3559	1692
5	10	0	10.1344	1.0571	98.0	0.014	3437	1698
		4	10.1286	1.0717	97.8	0.015	3480	1682
		8	10.2434	1.0449	99.4	0.004	3682	1648

We observe that the mean and the maximum number of objective function evaluations over all the experiments, in N_G generations of the GA, are 5,821 and 5,914, respectively, i.e. less than 22% of those required by the grid search (27,379). Furthermore, the actual number of objective function evaluations to achieve the computed optimal solution is generally lower, as shown by its mean value (*evmean*) and standard deviation (*evstd*) reported in Tables 2 and 3. Therefore, the GA approach allows a significant saving of the computational time with respect to the grid search.

The GA was also compared with three global optimization algorithms: Price's controlled random search (CRS) [25], particle swarm pattern search (PSwarm) [26], and DIRECT [27]. CRS and PSwarm are population-based, i.e. they maintain a population of candidate solutions evolving toward an optimal solution. DIRECT generates a sample of points that, as the number of iterations goes to infinity, form a dense subset of the search space. A description of the previous algorithms is beyond the scope of this paper; for details the reader is referred to the above references. We only note that CRS is "fully" heuristic, in the sense that no convergence results are available for it (at least for its original version); PSwarm, under appropriate assumptions, is globally convergent with probability 1 to first-order critical points; finally, DIRECT is deterministic, since its so-called "everywhere dense" convergence property guarantees that the algorithm is able to generate points arbitrarily close to a global optimum.

CRS was implemented in C, using the same pseudo-random number generator chosen for our GA. The following stopping criterion was applied: the difference between the maximum and the minimum value of F in the current population is lower than a specified tolerance, or the maximum number of objective function evaluations is achieved. In our experiments the previous tolerance and maximum number were set to 10^{-4} and 30,000, respectively. A C implementation of PSwarm was downloaded from http://www.norg.uminho.pt/aivaz/pswarm/. It combines different stopping criteria, based on various concepts (the norm of the velocity vector, the mesh size parameter and the clustering of the particles); default values were used for the related tolerances, as well as for the various parameters of the algorithm (see [26]). A maximum number of 30,000 objective function evaluations was also imposed, as for CRS. Finally, a Fortran 90 implementation of DIRECT was provided by G. Liuzzi, S. Lucidi and V. Piccialli, who developed it as a part of the work described in [28]. A maximum number of 30,000 function evaluations was used to stop this algorithm too. The parameter ϵ, used to

identify the so-called potentially optimal hyperintervals, was set to 10^{-4} [27]. The possibility of choosing between an initial population randomly extracted from a uniform distribution (default) or generated by RAND-GRID was added to CRS and PSwarm.

A further stopping criterion was introduced in the three previous implementations, which was combined with the other criteria through a logical "or":

$$F_{\mathrm{GA}} - F_{\mathrm{MAX}} < F_{\mathrm{GA}} \cdot TOL, \tag{4}$$

where, for each problem instance in a test set, F_{GA} is the mean value of the maxima of F computed by the GA (with SBLX-0.5) over the corresponding 30 runs, F_{MAX} is the maximum value of F at the current iteration of the algorithm under consideration, and TOL is a tolerance, set to 10^{-3} in our experiments. Note that, in CRS and PSwarm, F_{GA} refers to the GA using the same initial population, while in DIRECT it refers to the GA with the RAND-GRID strategy, using the best value of b for each test set ($b = 8$ for $m_1 = 5$ and $m_2 = 10$, $b = 0$ for the remaining problems). This criterion was introduced to compare the three solvers with the GA in terms of the number of objective function evaluations for computing a solution "close" to the GA solution.

The previous optimization solvers were run on all the test problems described at the beginning of this section. An initial population of 100 individuals was chosen for CRS and PSwarm, using both the default and the RAND-GRID strategy. Like the GA, the two non-deterministic algorithms were run 30 times for each problem instance. For the sake of space, we report in detail only the results concerning the most significant test set ($m_1 = m_2 = 1.4$) and comment more briefly on the other results. CRS and PSwarm generally perform better with RAND-GRID, therefore we discuss the results obtained with this strategy.

CRS and DIRECT are more efficient than the GA on the test set corresponding to $m_1 = 5$ and $m_2 = 10$, since they satisfy criterion (4) on 100% of the problems, and hence achieve 100% of success, with a number of objective function evaluations smaller than the GA. PSwarm is less efficient, since it gets at most 97.3% of success, with (4) satisfied in 40.7% of the cases (actually, this is the only case where the RAND strategy produces better results than the RAND-GRID one, showing 97.9% of success with (4) satisfied in 64.4% of the cases). For $m_1 = 1.4$ and $m_2 = 10$, CRS and DIRECT do not outperform the GA. CRS achieves a high percentage of success, i.e. 98–99%, with (4) satisfied in more than 91% of the cases; the number of objective function evaluations has a mean value ranging from 3,312 to 3,926 (depending on the value of b in the RAND-GRID strategy), but its largest value varies between 15,332 and 16,359, resulting much greater than for the GA. DIRECT gets 100% of success, with 90% of runs satisfying (4), but the mean value of the number of objective function evaluations is 12,438; furthermore the algorithm stops in 10% of the cases because the maximum number of objective function evaluations has been reached. As for $m_1 = 5$ and $m_2 = 10$, PSwarm is less effective than the GA, since it achieves a smaller percentage of success, i.e. at most 95.9%, with (4) satisfied in about 20% of the cases. The performance of CRS, PSwarm and DIRECT strongly deteriorates for $m_1 = m_2 = 1.4$, as shown by the results reported in Table 4 (the *stop* column reports the percentage of runs where the algorithm stops by satisfying (4)). The percentage of success of the three algorithms is very low (at most 43.4% with PSwarm), as well as the percentage of cases where (4) is satisfied (at most 27% with CRS), showing that the three algorithms are not able to compute solutions as good as the GA ones. Actually, the mean of the computed optimal values is smaller than 8, i.e. it does not reach the threshold used to measure the success of the algorithms. The worst results are obtained with DIRECT, which achieves only 23.3% of success and a mean of the optimal values equal to 7.2031. On the other hand, we verified that DIRECT, according to its convergence properties, is able to get solutions comparable to the GA ones, if a number

Table 4 Performance of CRS, PSwarm and DIRECT for $m_1 = m_2 = 1.4$

$m_1 = m_2 = 1.4$							
Algorithm	b	fmean	fstd	success (%)	stop (%)	evmean	evstd
CRS	0	7.7629	1.9030	40.3	24.1	5478	3828
	4	7.5542	1.8758	35.3	22.1	5839	3888
	8	7.4862	1.9000	31.6	27.0	5977	3912
PSwarm	0	7.8202	2.0224	43.2	21.6	1783	3948
	4	7.8268	2.0145	43.4	22.8	2258	4660
	8	7.7596	2.0119	40.9	23.9	3039	5838
DIRECT	–	7.2031	1.8919	23.3	10.0	28029	6562

The RAND-GRID strategy, with different values of the parameter b, is used by CRS and PSwarm

of objective function evaluations much larger than 30,000 is allowed. Of course, in this case DIRECT is far from being competitive with the GA and the grid search.

In conclusion, we developed a GA tailored to the DGW optimization problem, which is able to compute solutions comparable to those obtained by the grid search, but allows a strong reduction of the computational cost, thus providing a more powerful tool in the analysis of the noisy outputs of interferometers. The GA resulted also much more efficient than the CRS, PSwarm and DIRECT algorithms on the most significant and difficult set of problem instances used in the experiments. Future work will be devoted to improving the GA by introducing additional a priori information on the problem, and to exploring the effects of using multiple populations. We are also interested in comparing the GA with further global optimization algorithms [29,30].

Acknowledgments We thank G. Liuzzi, S. Lucidi and V. Piccialli for making available the implementation of DIRECT used in the computational experiments. We thank also the anonymous referees, whose comments and suggestions led us to improve the paper. This work was supported by the Italian MIUR, under the PRIN Project *Nonlinear Optimization, Variational Inequalities and Equilibrium Problems*, and by the Italian MAE and the Mexican CONACYT, under the Scientific and Technological Cooperation Project *Inverse Problems in Hydrology: Novel Numerical Methodologies*. Susana Gomez was supported also by INdAM-GNCS.

References

1. Thorne, K.S.: Gravitational radiation. In: Hawking, S.W., Israel, W. (eds.) 300 Years of Gravitation, pp. 330–458. Cambridge University Press, Cambridge (1987)
2. Babak, S., Balasubramanian, R., Churches, D., Cokelaer, T., Sathyaprakash, B.S.: A template bank to search for gravitational waves from inspiralling compact binaries I: physical models. Class. Quantum Grav. **23**, 5477–5504 (2006)
3. Mohanty, S.D.: Hierarchical search strategy for the detection of gravitational waves from coalescing binaries: extension to post-newtonian waveforms. Phys. Rev. D **57**(2), 630–658 (1998)
4. Sengupta, A.S., Dhurandhar, S., Lazzarini, A.: Faster implementation of the hierarchical search algorithm for detection of gravitational waves from inspiraling compact binaries. Phys. Rev. D **67**(8), 082,004 (2003)
5. Milano, L., Barone, F., Milano, M.: Time domain amplitude and frequency detection of gravitational waves from coalescing binaries. Phys. Rev. D **55**(8), 4537–4554 (1997)
6. Barrón, C., Gómez, S., Romero, D., Saavedra, A.: A genetic algorithm for Lennard–Jones atomic clusters. Appl. Math. Lett. **12**(7), 85–90 (1999)
7. Papoulis, A.: Probability, Random Variables, and Stochastic Processes. 3rd edn. McGraw-Hill, New York (1991)

8. Sathyaprakash, B.S., Dhurandhar, S.V.: Choice of filters for the detection of gravitational waves from coalescing binaries. Phys. Rev. D **44**(12), 3819–3834 (1991)
9. Dhurandhar, S.V., Sathyaprakash, B.S.: Choice of filters for the detection of gravitational waves from coalescing binaries. II. Detection in colored noise. Phys. Rev. D **49**(4), 1707–1722 (1994)
10. Allen, B., et al.: LAL Software Documentation. Revision 1.44 (2005). Available at http://www.lsc-group.phys.uwm.edu/lal/
11. Mohanty, S.D., Dhurandhar, S.V.: Hierarchical search strategy for the detection of gravitational waves from coalescing binaries. Phys. Rev. D **54**(12), 7108–7128 (1996)
12. Blanchet, L., Rlyer, B., Wiseman, A.G.: Gravitational waveforms from inspiralling compact binaries to second-post-Newtonian order. Class. Quantum Grav. **13**, 575–584 (1996)
13. Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
14. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer, New York (1998)
15. Herrera, F., Lozano, M., Verdegay, J.L.: Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. Artif. Intell. Rev. **12**(4), 265–319 (1998)
16. Maaranen, H., Miettinen, K., Penttinen, A.: On initial populations of a genetic algorithm for continuous optimization problems. J. Glob. Optim. **37**(3), 405–436 (2007)
17. Bäck, T., Fogel, D.B., Michalewicz, Z. (eds.): Evolutionary Computation 1: Basic Algorithms and Operators. IOP Publishing, Bristol (2000)
18. De Jong, K.A.: Evolutionary Computation: A Unified Approach. MIT press, Cambridge (2006)
19. Herrera, F., Lozano, M., Sánchez, A.M.: A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study. Int. J. Intell. Syst. **18**(3), 309–338 (2003)
20. Rasio, A.R., Shapiro, S.L.: Coalescing binary neutron stars. Class. Quantum Grav. **16**(6), R1–R29 (1999)
21. Matsumoto, M., Nishimura, T.: Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comp. Sim. **8**(1), 3–30 (1998)
22. Mitra, S., Dhurandhar, S.V., Finn, L.S.: Improving the efficiency of the detection of gravitational wave signals from inspiraling compact binaries: Chebyshev interpolation. Phys. Rev. D **72**, 102,001 (2005)
23. Back, T: Mutation parameters. In: Back, T., Fogel, D.B., Michalewicz, Z. (eds.) Evolutionary Computation 2: Advanced Algorithms and Operators, pp. 142–151. IOP Publishing, Bristol (2000)
24. Vajda, P., Eiben, A., Hordijk, W.: Parameter control methods for selection operators in genetic algorithms. In: Parallel problem solving from nature—PPSN X, Lecture Notes in Computer Science, pp. 620–630. Springer, Berlin/Heidelberg (2008)
25. Price, W.: A controlled random search procedure for global optimisation. Comput. J. **20**, 367–370 (1977)
26. Vaz, A.I.F., Vicente, L.N.: A particle swarm pattern search method for bound constrained global optimization. J. Glob. Optim. **39**(2), 197–219 (2007)
27. Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitzian optimization without the Lipschitz constant. J. Optim. Theory Appl. **79**(1), 157–181 (1993)
28. Liuzzi, G., Lucidi, S., Piccialli, V.: A direct-based approach exploiting local minimizations for the solution of large-scale global optimization problems. Comput. Optim. Appl. (2008)
29. Strongin, R.G.: Algorithms for multi-extremal mathematical programming problems employing the set of joint space-filling curves. J. Glob. Optim. **2**(4), 357–378 (1992)
30. Strongin, R.G., Sergeyev, Y.D.: Global Optimization with Non-Convex Constraints Sequential and Parallel Algorithms, Nonconvex Optimization and its Applications vol. 45. Kluwer, Dordrecht (2000)