UNIVERSITA' DEGLI STUDI DI NAPOLI "FEDERICO II"

Facoltà di Scienze Matematiche, Fisiche e Naturali

Global Optimization Methods for the Detection of Gravitational Waves

Filippo Riccio

Dottorato di Ricerca in Scienze Matematiche

XXI Ciclo

Advisor

Prof. Daniela di Serafino

Coordinator

Prof. Francesco de Giovanni

Acknowledgments

I would like to express my sincere gratitude to Prof. Daniela di Serafino for her support, continuous guidance, meticulous suggestions, and inexhaustible patience during the development of this thesis.

I am deeply grateful to Prof. Gerardo Toraldo, who played a key role in the course of my studies. His continuous encouragement and invaluable suggestions have always stimulated my work.

I would like to thank Prof. Leopoldo Milano for introducing me to the fascinating world of gravitational waves and Prof. Susana Gomez for her important comments and suggestions.

I would also like to thank Prof. Stefano Lucidi, Dr. Veronica Piccialli and Dr. Giampaolo Liuzzi not only for making available their implementation of DIRECT, used in the numerical experiments, but also for their friendly support and useful discussions.

I am thankful to Dr. Valentina De Simone and Prof. Giuseppe Di Maio for the affection shown to me. I would like to remember Prof. Marco D'Apuzzo for sharing with me his happiness to life.

I am also thankful to my friends Antonio, Giovanni, Giuseppe, Giusy, and Rocco. A special thanks to Giuseppina for her tender support in important and difficult moments.

Finally, I would like to express my deep gratitude to my parents, my sister and my wife, Mariafelicia, for their constant love and help in finding my way through life.

Contents

Preface

Many problems in pure and applied sciences can be formulated as global optimization problems in which the objective function is nonconvex and has many local minima. An intense research activity has been devoted to developing numerical solution methods, but there are still several open problems.

In this thesis we consider a global optimization problem arising in the detection of gravitational waves. The detection of such waves has a fundamental role in modern astrophysics. So far, only indirect evidences of the existence of gravitational waves have been provided because of the many difficulties arising in the detection process. Network of detectors have been recently deployed, but highly effective data analysis techniques are still needed to filter the output of the detectors.

Coalescing binary systems (neutron stars and/or black holes) are very promising sources of gravitational waves for ground-based laser interferometric detectors. For these sources the most widely used detection technique is the *generalized likelihood ratio test*, which corresponds to the application of the *matched filtering* technique. A crucial issue in this methodology is the solution of a box-constrained global optimization problem. This problem is hard to solve because of the strong nonlinearity of the objective function, the unavailability of its derivatives, the presence of many local solutions, 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, the solution of the optimization problem changes with the specific realization of the noise.

In the astrophysics community, this optimization problem is usually solved by

applying the *grid search* technique on a suitable discretization of the feasible domain that allows to satisfy certain accuracy requirements. However, this technique needs a large number of objective function evaluations and hence has a high computational cost. The reduction of such a cost is crucial in the overall detection process and represents the main motivation for our investigations.

In order to solve the above optimization problem we considered *genetic algorithms*. These algorithms are generally able to compute satisfactory solutions, especially when the objective function is a black box for which little or no additional information is available. We developed a real-coded genetic algorithm which exploits characteristic features of the problem itself. Special attention was devoted to the choice of the initial population and of the recombination operator. Numerical 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. Furthermore, the genetic algorithm largely outperforms other global optimization algorithms on significant instances of the problem.

To further reduce the execution time in the solution of the optimization problem, we developed a parallel version of our genetic algorithm using the *multiple-deme* approach. This approach allows a great flexibility in the design of the algorithm and hence a better adaptation to the problem. Numerical experiments showed that the parallel algorithm allows to increase the accuracy and the reliability of the sequential genetic algorithm, and to obtain results comparable to the grid search in terms of accuracy, but with a lower computation time.

This thesis is organized as follows. In Chapter 1 we give an introduction to Global Optimization, presenting results on the existence and the characterization of the solutions. Then we discuss general features of global optimization methods and give a classification of them in terms of convergence properties.

In Chapter 2 we introduce the problem of the detection of gravitational waves, highlighting its importance in modern astrophysics. We briefly illustrate ground-based laser interferometric detectors such as VIRGO and describe gravitational sig-

nals emitted by coalescing binary systems. Then we present the *generalized likelihood ratio test*, showing that the solution of a box-constrained global optimization problem is the main computational kernel of this methodology. Finally, we describe the grid search, which is currently the most used algorithm to solve the global optimization problem.

In Chapter 3 we present the genetic algorithm developed for the optimization problem under consideration. We briefly introduce genetic algorithms, outlining their structure and main components. We focus on the representation of the individuals and the choice of the genetic operators and related parameters, in order to design a suitable algorithm for our problem. Special attention is devoted to the initial population, which is chosen by combining information on the local variability of the objective function, derived from the knowledge of the physical problem, with a methodology which guarantees a good covering of the feasible domain. Furthermore, the recombination operator is designed for properly handling the constraints. Finally, the results of an extensive testing activity, devoted to the evaluation of the genetic algorithm, as well as to its comparison with other global optimization algorithms, are presented.

In Chapter 4 we describe a parallel version of the previous genetic algorithm. This version was developed for MIMD distributed memory systems, using the message-passing paradigm, in order to reduce the execution time while enhancing the effectiveness of the sequential algorithm. We first outline different approaches to the development of parallel genetic algorithms. Then we present our algorithm, based on the mutliple-deme approach, which uses several subpopulations that evolve independently, but exchange individuals occasionally through the migration operator. We focus on the migration operator to obtain a suitable migration strategy for the optimization problem. Numerical experiments show the effectiveness of our approach.

Chapter 1

Global Optimization

We first introduce global optimization, presenting results on the existence and the characterization of solutions. Then we deal with general features of any global optimization methods and given a classification according to their convergence properties.

1.1 Introduction

Global optimization is a relatively young field of applied mathematics. It fundamentally dates back to 1975-78, when the two volumes *Towards Global Optimisation*, by Szego and Dixon, appeared [1, 2]. These volumes are the first books containing a collection of papers that present different solution methods for global optimization problems with continuous variables.

The reason why global optimization remained marginal for a long time with respect to local optimization is simply that global optimization problems may be very difficult to deal with. A difficulty is related to the fact that differential calculus, which plays a fundamental role in local optimization for characterizing the solutions and for devising solution methods, cannot be generally applied to global optimization. Indeed, the differential of any order of a function at a point is a local notion, while global optimization need notions that must be "global" . Only in the case in

which the objective function is convex, local methods can be applied to solve global optimization problems since each local optimum is also global.

However, the field of global optimization has received increasing attention by researchers since many problems in pure and applied science can be formulated as global optimization problems: packing problems as the knapsack problem [3] or the Kepler's conjecture [4] in geometry, the travelling salesman problem [5] or the maximum clique problem [6] in graph theory, the protein folding [7] and equilibrium problems in chemistry [8], and the scheduling problem [9] in computer science, just to name a few. Furthermore, in many problems the objective function has many local optima, then methods are needed that distringuish among these local optima for locating the best possible one.

In order to characterize a solution of a global optimization problems several approaches have been investigated. These approaches either are difficult to be used for devising solution methods [10, 11, 12, 13] or can just be applied to optimization problems with special structure [14, 15, 12].

One more difficulty in the solution of global optimization problems is related to the development of efficient stopping criteria. These criteria require the knowledge of global information on the optimization problem. Indeed, in absence of global information the success of any method is possible only through a dense covering of feasible domain [16]. This result has a strong impact on the solution of problem in which the objective function is a *black box*, i.e. the only information on the problem is the value of the function is a given point. In these situations the experimental analysis assumes a fundamental role to choose the most suitable method for problems under consideration [17, 18].

1.2 Global Optimization Problems

1.2.1 Problem Formulation

In general an optimization problem can be written as

$$\begin{cases} \text{minimize } f(x) \\ \text{subject to } x \in A \end{cases} \tag{1.1}$$

The function $f : A \subseteq \mathbb{R}^n \to \mathbb{R}$ is called the *objective function* and the set $A \subseteq \mathbb{R}^n$ is called the *feasible domain*. Maximization problems are included in the formulation (1.1) because, if $\max\{f(x) \mid x \in A\}$ exists, then

$$\max\{f(x) \mid x \in A\} = -\min\{-f(x) \mid x \in A\};$$

thus, without loss of generality, we consider global optimization problems like (1.1) unless otherwise specified.

The aim of *global optimization* [19] is to solve one or both of the following problems:

Problem 1. *Find a point $x^* \in A$ such that $f(x^*) \leq f(x)$, for all $x \in A$.*

Problem 2. *Find the value $f^* = \min f(x)$, with $x \in A$.*

A solution x^* of Problem 1 is called a *global minimizer*; while the value f^* that is the solution of Problem 2 is called *global minimum*. The set of all global minimizers is

$$X_f^* = \{x^* \in A \mid f(x^*) \leq f(x)\ \forall x \in A\}.$$

The nature of global optimization problem depends on the characteristics of the objective function and the feasible domain. We focus on continuous global optimization problems which, in general, are classified as *unconstrained* problems when $A = \mathbb{R}^n$ or, more generally, A is an open subset of $\mathbb{R}^n$, and as *constrained* problems when A is defined by a set of equality and inequality constraints

$$A = \{x \in \mathbb{R}^n \mid g(x) \leq 0, h(x) = 0\},$$

where $g : \mathbb{R}^n \to \mathbb{R}^m$ and $h : \mathbb{R}^n \to \mathbb{R}^p$ are given functions.

1.2.2 Existence Conditions

Concerning the existence of a global minimizer, these situations may happen:

- the feasible domain A is empty;
- the feasible domain A is nonempty, but the objective function is unbounded from below;
- the feasible domain A is nontempty and the objective function is bounded from below, but global minimizers do not exist;
- global minimizers exist.

When the feasible domain A is a compact set, a sufficient condition for the existence of a global minimum is given by the following well-known theorem [20]

Theorem 1.2.1 (Weierstrass)**.** *Let A be a nonempty compact set of $\mathbb{R}^n$ and let f be a continuous real-valued function on A. Then there exists at least a global minimizer of f on A.*

We note that Theorem 1.2.1 is satisfied in weaker conditions. More precisely, given the definition

Definition 1.2.1. *Let A be a subset of $\mathbb{R}^n$ and let f be a real-valued function on A. The function f is said* lower semi-continuous *at a point $x_0 \in A$ if*

$$\liminf_{x \to x_0} f(x) \geq f(x_0).$$

the following theorem is satisfied [20]

Theorem 1.2.2. *Let A be a nonempty compact set of $\mathbb{R}^n$ and let f be a lower semi-continuous real-valued function on A. Then there exists at least a global minimizer of f on A.*

The previous theorem ensures the existence of a solution of a constrained global optimization problem provided that the objective function is lower semi-continuous.

When $A = \mathbb{R}^n$ we can obtain sufficient conditions for the existence of a global minimizer using the notion of either level set or coercivity. We start giving the following definition

Definition 1.2.2. *Let f be a real-valued function on $\mathbb{R}^n$ and $\alpha \in \mathbb{R}$. The* level set *of the function f is defined as*

$$L(f;\alpha) = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq \alpha\}.$$

Using Theorem 1.2.1 it is possible to prove the following theorem [20]

Theorem 1.2.3. *Let f be a continuous real-valued function on $\mathbb{R}^n$. If there exists a nonempty and compact level set of f, then f has at least a global minimizer in $\mathbb{R}^n$.*

Now we give the definition of n-coercivity.

Definition 1.2.3. *Let f be a real-valued function on $\mathbb{R}^n$. The function f is said* n-coercive *if*

$$\lim_{||\mathbf{x}||\to\infty} \frac{f(\mathbf{x})}{||\mathbf{x}||^n} = +\infty.$$

The following theorem holds [20]

Theorem 1.2.4. *Let f be a continuous and 0-coercive real-valued function on $\mathbb{R}^n$. Then f has at least a global minimizer on $\mathbb{R}^n$.*

From the previous results we can deduce that Problem 2 of finding the global minimum f^* is a well-posed problem. Indeed the previous theorems guarantee the existence of the global minimum which is obviously unique; the continuous dependence on the data is easly shown since for any continuous function f and g we have:

$$|f^* - g^*| \leq \sup |f(\mathbf{x}) - g(\mathbf{x})| = ||f - g||_\infty.$$

Conversly, the problem 1 is ill-posed because x^* may not be unique. Furthermore, even if it is unique, the continuous dependence on the data is not satisfied in the uniform topology. This means that there exist continuous functions with arbitrarily

small maximum absolute difference of the function values but global minimizers wide apart. An example is given by the following class of function f_δ [21]

$$f_\delta(x) = \cos(x) + \delta x, \text{ with } x \in [-2\pi, 2\pi].$$

If $\delta > 0$ then $x^* \simeq -\pi$, while if $\delta < 0$, $x^* \simeq \pi$. Therefore, if $\delta_1 < 0 < \delta_2$

$$||f_{\delta_1}(x) - f_{\delta_2}(x)|| \leq 2\pi|\delta_2 - \delta_1|,$$

but the distance between the global minimizers is approximately 2π.

1.2.3 Optimality Conditions

The goal of deriving necessary and sufficient conditions for a feasible point to be a global minimum is a very ambitious, unless the objective function has some special properties. The simplest property is the convexity, which leads to the equivalence between local and global optimality conditions. However, the convexity encompasses just a very restricted class of optimization problems. More general conditions can be derived by completing classical conditions for local optimality with some global conditions. Following [13] we consider $\bar{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$ and the class $\mathcal{F}$ of nonconvex functions $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ with the following properties:

$$D_f = \{x \in \mathbb{R}^n \mid f(x) < +\infty\} \text{ is nonempty;} \tag{1.2}$$

$$f \text{ is lower-semicontinuous on } \mathbb{R}^n; \tag{1.3}$$

$$f \text{ is 1-coercive on } \mathbb{R}^n. \tag{1.4}$$

These properties ensure that for all $f \in \mathcal{F}$ the lower bound is finite and achieved, and the set of its global minima is a nonempty compact set. We now define the *convex hull* of a function $f \in \mathcal{F}$ and of a set A

Definition 1.2.4. *The* convex hull *of a function $f \in \mathcal{F}$ is a function F such that*

$$F(x) = \sup\{g(x) \mid g : \mathbb{R}^n \to \bar{\mathbb{R}} \text{ convex}, g \leq f\},$$

Definition 1.2.5. *The* convex hull *of a set $A \subseteq \mathbb{R}^n$ as the smallest convex set $C(A)$ that contains A.*

It can be shown that $C(A)$ consists of all the convex combinations, $\lambda_1 x_1 + \cdots + \lambda_n x_n$, with $\lambda_i \geq 0$ and $\sum \lambda_i = 1$, of the elements of A. The following theorem links the minimization of a function f to that of its convex hull F [13]

Theorem 1.2.5. *Let $f \in \mathcal{F}$ and let F be its convex hull. The following results hold:*

$$\min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} F(x), \tag{1.5}$$

$$X_F^* = C(X_f^*), \tag{1.6}$$

where X_f^*, X_F^* are the sets of the global minimizers of the function f and its convex hull F, respectively.

Hence any global minimizer of the convex hull F is a convex combination of global minimizers of the function f. Furthermore, it is easy to note that x is a global minimizer of f if and only if x is a global minimizer of F and $F(x) = f(x)$. From Theorem 1.2.5 it follows that the convex hull might have a relevant rule in the solution of global optimization problems. Indeed we may attempt to solve a nonconvex problem by solving a convex problem where the objective function is the convex hull of the original problem. However, in general, finding the convex hull of a function is as difficult as computing its global minimum [13]. Only in some cases the convex hull can be explicitely described, as in the following theorems [20].

Theorem 1.2.6. *Let $v_1, \ldots, v_k$ be the vertices of a polytope $P \subseteq \mathbb{R}^n$. The convex hull F of a concave function f over P can be expressed as*

$$F(x) = \min_{\alpha} \sum_{i=1}^{k} \alpha_i f(v_i),$$

where

$$\sum_{i=1}^{k} \alpha_i v_i = x, \ \sum_{i=1}^{k} \alpha_i = 1, \alpha_i \geq 0, \ i = 1, \ldots, k.$$

Theorem 1.2.7. *Let S be the simplex generated by the vertices $v_0, v_1, \ldots, v_n \in \mathbb{R}^n$, and let f be a concave function defined on S. Then the convex hull of f over S is the affine function $l(x) = c^T x + b$ that is uniquely determined by the system of linear equations $f(v_i) = c^T v_i + b$, with $i = 0, \ldots, n$.*

In the case the objective function is differentiable we have the following theorem [10]:

Theorem 1.2.8. *Let $f \in \mathcal{F}$ be a differentiable function. Then x^* is a global minimizer of f if and only if*

$$\nabla f(x^*) = 0 \tag{1.7}$$

$$F(x^*) = f(x^*). \tag{1.8}$$

In such a case F is differentiable at x^* and $\nabla F(x^*) = 0$.

Equality (1.7) is a "global condition" added to the local optimality condition (1.8) to ensure that the stationary point x^* is a global minimizer. Furthermore we can conclude that for a differentiable function fm the stationary points are global minima that are global minimizers are those that satisfying

$$\{x^* \mid \nabla f(x^*) = 0\} \subset \{x^* \mid F(x^*) = f(x^*)\}.$$

Again this condition cannot be generally used to solve a global optimization problems, since, as we already noted, the determination of the convex hull is very difficult. However, we can use this condition in its negative form: a point x^* (stationary or not) such that $l < f(x^*)$, where l is un upper bound of $F(x^*)$ cannot be a global minimum.

Another approach for deriving characterizations of global minimizers is based on the measure theory. Following [11, 12], we assume that the feasible domain is a *robust set*, that is

Definition 1.2.6. *A set A of $\mathbb{R}^n$ is called* robust *if it is the closure of a nonempty bounded open set.*

According to the previous definition, a robust set has no isolated points. Thus, for each point of a robust set A, the following property is satisfied:

$$\forall x \in A \text{ and } \forall \epsilon \; \exists \; x' \mid x' \in I(x, \epsilon) \cap Int(A)$$

where $I(x, \epsilon)$ is a neighbourhood of x of radius ϵ, and $Int(A)$ is the interior of A. Moreover, we suppose that the objective function $f : A \to \mathbb{R}$ is continuous on A. The following theorem holds.

Theorem 1.2.9. *Let $A \subset \mathbb{R}^n$ be a robust set and let f be a continuous function on A. A point $x^* \in A$ is a global minimizer of f on A if and only if the level set $L(x^*) = \{x \in A \mid f(x) < f(x^*)\}$ has a null Lebesgue measure.*

When the objective function has a unique global minimizer we have the following characterization:

Theorem 1.2.10. *Let $A \subset \mathbb{R}^n$ be a robust set and let f be a continuous function on A. If the function f has a unique global minimizer $x^* \in A$, then*

$$\lim_{k \to \infty} \frac{\int_A x_i e^{-kf(x)} dx}{\int_A e^{-kf(x)} dx} = x_i^*$$

for $i = 1, \ldots, n$.

As before, these results are very difficult to use for devising a solution method for global optimization problems.

1.3 Global Optimization Methods

1.3.1 General Features

Several methods have been devised to solve global optimization problems [22, 23]. These methods may be divided in *deterministic* and *stochastic* methods, depending on whether or not they incorporate any stochastic element [24, 21]. All global optimization methods are iterative methods that generate a sequence of trial points $\{x_i\}_{i \in \mathbb{N}}$ in the feasible domain which converges in some sense to a solution.

In order to formalize previous notions, we first consider the following class of functions

$$\mathcal{F} = \{f : A \to \mathbb{R} \mid A \subseteq \mathbb{R}^n \text{ compact set with no isolated points}, f \text{ continuous}\}$$

and give the definition below.

Definition 1.3.1. *The class of functions $\mathcal{F}$ is called* sufficiently rich *if $\forall y \in \mathbb{R}, \forall x \in A, \forall f \in \mathcal{F}, \forall N$ open subset of A such that $x \in N$, there exists a function $g \in \mathcal{F}$ such that $g(x) = y$ and $g(x) = f(x), \forall x \in A \setminus N$*

Examples of functions in this class are the C^0, C^n and C^∞ functions, the Lipschitz-continuous functions, and the functions with Lipschitz-continuous derivatives. Let $\mathcal{X}$ be the set of all finite sequences in A. We define formally the notion of *local information*

Definition 1.3.2. *A* local information *for $\mathcal{F}$ is a function LI defined on $\mathcal{F} \times \mathcal{X}$ such that $\forall f, g \in \mathcal{F}, \forall X \in \mathcal{X}, \forall N$ open subset of A containing X, the following property is satisfied*

$$f(x) = g(x)\ \forall x \in N \Longrightarrow LI(f, X) = LI(g, X)$$

A local information may includes any information depending on function values and/or derivatives, but also any formula depending on them.

Let $\{x_i\}_{i \in \mathbb{N}}$ be sequence generated by a method and let X_k be the finite sequence $\{x_1, \ldots, x_k\}$. Now we define deterministic and stochastic sequential methods

Definition 1.3.3. *A* deterministic sequential method *on $\mathcal{F}$ is a method for which there is a local information function LI such that, for all $f \in \mathcal{F}$, the trial point x_{k+1} depends only on $LI(f, X_k)$.*

Definition 1.3.4. *A* stochastic sequential method *on $\mathcal{F}$ is a method for which there is a local information function LI such that, for all $f \in \mathcal{F}$, the trial point x_{k+1} depends on $LI(f, X_k)$ and an instance ω_{k+1} of a random variable.*

We denote with X_f the sequence of trial points generated by a method running on the function f. In the case of a stochastic sequential method, X_f is a random variable. Let $\omega = \{\omega_i\}_{i\in\mathbb{N}}$ and let $X_f(\omega)$ be an instance of X_f. We denote by $\bar{X}_f$ the closure of X_f and by X'_f the set of limit points. We note that X'_f is nonempty since A is compact and $\bar{X}_f = X_f \cup X'_f$. We recall that X^*_f denote the set of global minimizer. Following [25], we define two types of convergence.

Definition 1.3.5. *A method is said to "see" the global minimum of f if*

$$\bar{X}_f \cap X^*_f \neq \emptyset.$$

This type of convergence ensure that it is possible to construct a subsequence which converges to a global minimizer. This convergence is typical of methods for which the emphasis is on finding the global minimum.

Definition 1.3.6. *A method is said to "localize" the global minimizers if*

$$X'_f = X^*_f \quad (\textit{or weaker } \emptyset \neq X'_f \subseteq X^*_f).$$

This type of convergence ensure that subsequences of trial points converge only to global minimizers. This convergence is typical of methods which emphasize finding the global minimizers.

In the context of numerical methods, since we cannot carried out an infinite number of iterations, we relaxe Problems 1 and 2 in the following problems

Problem 3. *Find $x \in A$ that satisfies*

$$e(x, X^*_f) = \min_{x^*\in X^*_f} d(x, x^*) \leq \epsilon$$

where d is a given distance in $\mathbb{R}^n$, ϵ is a fixed tolerance and X^*_f is the set of all global minimizers.

Problem 4. *Find a value $f(x)$ that satisfies*

$$f(x) \leq f^* + \epsilon$$

where $\epsilon > 0$ is a fixed tolerance and f^* is the global minimum.

However also for these relaxed problems is difficult to establish when an approximation has computed with a fixed tolerance. In order to show this we give some theorems concerning the two forms of convergence defined above [25]. In the case of deterministic sequential methods the following theorems holds.

Theorem 1.3.1. *Any deterministic sequential method on $\mathcal{F}$ sees the global minimum of $f, \forall f \in \mathcal{F}$ if and only if*

$$\bar{X}_f = A \forall f \in \mathcal{F}.$$

Since localizing implies seeing, it follows immediately that a method localizing $\forall f \in \mathcal{F}$ implies that $\bar{X}_f = A \ \forall f \in \mathcal{F}$. On the other hand, the following result holds:

Theorem 1.3.2. *For any deterministic sequential method on $\mathcal{F}$, there exists a function $f \in \mathcal{F}$ for which the method fails to localize the global minimizer of f.*

We denote with B_f the event corresponding to "the method sees the global minimum of f" and with C_f the event "the method localizes the global minimizers". The analog of Theorems 1.3.1 and 1.3.1 for stochastic sequential methods are given below.

Theorem 1.3.3. *For any probability α and any stochastic sequential method, it results that $P(B_f) \geq \alpha, \forall f \in \mathcal{F}$ if and only if $P(x \in \bar{X}_f) \geq \alpha, \ \forall x \in A, \forall f \in \mathcal{F}$, where $P(\cdot)$ is a probability function.*

It follows immediately that $P(C_f) \geq \alpha, \forall f \in \mathcal{F}$ if and only if $P(x \in \bar{X}_f) \geq \alpha, \ \forall x \in A, \forall f \in \mathcal{F}$.

Theorem 1.3.4. *For any $\epsilon > 0$ and any stochastic sequential methods, there exists a function $f \in \mathcal{F}$ such that $P(C_f) < \epsilon$.*

The previous theorems imply that the convergence of any method that uses local information is possible only if the feasible domain is covered by a dense set of trial points. Moreover, the localization of the global minimizers is not possible for all functions in the class of sufficiently rich functions. Indeed, there exist functions for which deterministic methods fail or stochastic methods fail with arbitrarily

high probability. This have a strong impact on the solution of global optimization problems.

Even if a global optimization method converges to the global minimum, the determination of a stopping criterion requires "global" information on the class of problems under consideration. Examples of global information are the Lipschitz constant, the number of local minima, the value of global minimum, etc. However this information is not available in many applications. In these cases the experimental analysis assumes a fundamental role in the development of a global optimization method. In general, the development of any global optimization methods requires a trade-off between the computational cost and the quality of the solution. In the applications in which the computational cost has a fundamental importance several methods, which do not ensure the convergence, have been developed. Fundamentally, these methods are based on heuristics and in many problem they are be able to compute "satisfactory" solutions.

1.3.2 Classification of Methods

Previously we divided global optimization methods in two groups: determinstc and stochastic. Even if these methods are based on different philosophies, current advanced global optimization methods often combine these philosophies and cannot be put in any of the two groups. A more adequate approach to classify global optimization methods is based on their type of convergence [26]:

- *incomplete methods*, which are based on clever intuitive heuristics, but are not guaranteed to convergence to the global minimum;
- *asymptotically complete methods*, which ensure convergence to the global minimum with certainty or at least with probability one, but do not provide any means to known when the global minimum has been found;
- *complete methods*, which ensure convergence to the global minimum, assuming exact computations, and allow to establish that an approximation of the global

minimum has been found with a fixed tolerance when global information about the problem is known;

- *rigorous methods*, which ensure convergence to the global minimum and within given tolerances even in the presence of rounding errors, except in near-degenerate cases, where the tolerances may be exceeded.

Although incomplete methods do not provide a general guarantee of convergence, they are frequently applied with success to many difficult problems. The fundamental advantage of these methods is that they require little information about the optimization problem and for several problems they represent the only feasible choice.

Several incomplete methods are based on the analogies to natural processes, the global minimum usually represents some equilibrium state. Among these there are *smoothing methods* [27] which are based on the intuition that in nature macroscopic features are usually an average effect of microscopic details. The idea is fundamentally that of transforming the original problem into a problem with a single minimizer through the definition of a homotopy. This homopoty is defined by introducing an additional parameter t into the problem in such a way that $t = 0$ gives the original problem, while $t = 1$ gives either a related convex problem or a related problem with a unique and known global minimizer. Then a sequence of local optimization problem is solved for $t = t_1, t_2, \ldots, t_n$, where the t_i form a descreasing sequence starting at 1 and ending at 0. Each time, the solution of the previous problem is taken as the starting point for the current problem. The quality of the final result depends on the homotopy, and frequently it is a least a local minimizer.

Simulated annealing [28] takes its intuition from the fact that the heating and slow cooling of a metal brings it into a more uniformly crystalline state that corresponds to a state in which the enery takes its global minimum. In its original form, the simulated annealing method is provably convergent in a probabilistic sense, but exceedingly slow. Various ad hoc enhancements have been developed for improving its computational cost [29].

Genetic algorithms [30, 31] make use of analogies to biological evolution by using genetic operator for producing better and better individuals that represent candidate solutions to the problem at hand. The efficiency of these algorithm is strongly depended on the design and tuning of genetic operators. The main advantage of genetic algorithms is related to the exploitation of specific knowledge about the problem within the algorithms. This allow to improve considerably their efficiency.

Particle swarm [32] is based on the simulation of the social behaviour of a population of agents or particles. Some of its advantages are the simplicity of implementation and ease of parallelization. Furthermore, they depend on a few of parameters, so they do not require an intensive tuning phase.

The simplest asymptotically complete method is the *pure random search* [33], in which the trial points are chosen randomly with uniform distribution from the feasible domain. This method converge with probability 1 to the global minimum, but it is rather inefficient. In order to improve the efficiency of the pure random search, the *multistart methods* [34] have been introduced. In these methods a local optimization algorithm is applied for each random trial point. This approach has the disadvantages that the same local minima can be computed several times. In order to improve their efficiency these methods have been modified in different ways [35]. One of these is based on the use methods from the clustering theory [16]. The basic idea is to locate regions of attraction of local minima and to start one local search for each region of attraction.

An example of complete method is the *grid search*, which belongs to class of *passive strategies* [36] where each trial point does not depend on the function values at the other points. In these strategies, grids of finer and finer trial points are used to approximate the global minimum. The accuracy of the approximation is related to global information of the objective function. In particular for the class of Lipschitz functions, it is possible to compute an approximation of the global mimimum with a fixed tolerance and in a finite number of steps. However the grid search requires a huge number of function evaluations which grows exponentially with the dimension

of the problem.

In the context of Lipschitz optimization several methods have been developed as the Shubert algorithm [37] for univariate objective functions and its extensions to higher dimensions [38]. Generally speaking, these methods combine branching techniques with lower bound estimations of the objective function based on the knowledge of the exact value or an overestimation of the Lipschitz constant. However, in many applications it is hard or impossible to know the value of the Lipschitz constant. For this reason several methods have been developed that use local approximations of the Lipschitz constant or do not require the explicit knowledge of the Lipschitz constant [39]. Examples of complete methods that use only local information are DIRECT [40, 41] , MCS [42] and LGO [43]. All the three methods employ a branching strategy to guarantee convergence. This strategy generates a sequence of partitions of the feasible domain in which the diameters of all sets of partitions converge to zero. The main differences are in how and when to split the domain, and what is done within each set of partitions. Furthermore, since these methods do not use global information they must generate a dense set of trial points to ensure the convergence.

Finally, the rigorous methods [44, 45] fundamentally combine the interval analysis with strategies of branching and bound. These methods allow to determine a set of intervals that contains all the global minimizers, but they require very high computational costs. For these reasons these methods are generally used in computer-assisted proofs [26].

Chapter 2

The Detection of Gravitational Waves

We first introduce the problem of the detection of gravitational waves, highlighting its importance in the context of modern astrophysics. Then we outline the most used method for the detection of gravitational waves emitted from coalescing binary system, showing that a crucial issue is the solution of a global optimization problem. Finally we briefly describe the most used algorithm in astrophysics community to solve this optimization problem.

2.1 Introduction

The first studies on the gravitational waves are due to Einstein and the main result is known as the "quadrupole formula". This formula plays a role, in gravity theory, analogous to the dipole formula for electromagnetic radiation, showing that gravitational waves arise from accelerated masses exactly as electromagnetic waves arise from accelerated charges.

From the quadrupole formula and the weakness of the gravitational interaction it follows that gravitational waves are difficult to produce and very large masses moving at relativistic speeds are needed. For this reason, typical sources of gravitational

waves are astrophysical objects. Indeed, an indirect confirmation of the existence of gravitational waves has come from the study of binary neutron star systems. The most celebrated example is the "Hulse-Taylor" pulsar, B1913+16, reported by Hulse and Taylor in 1975 [46]. Thirty years of observation have shown an agreement between experimental results and predictions of general relativity. For this discover Hulse and Taylor were awarded the Nobel Prize in 1993.

Nowadays the detection of gravitational waves is one of the most awaited events in the modern astrophysics. 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 [47]. Several ground-based laser interferometric detectors are either operated or under deployment in Europe, United States, Japan and Australia, but gravitational waves have not yet been directly 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. This is because a model of the emitted waves is available and a relatively large number of events per year is expected (tens per year within a few hundred Mpc) [48]. In this case the most widely used detection technique is the generalized likelihood test, which exploits the waveform of the signal and assumes that the instrumental noise is a stationary white Gaussian stochastic process. A crucial issue in this methodology is the solution of a box-constrained global optimization problem.

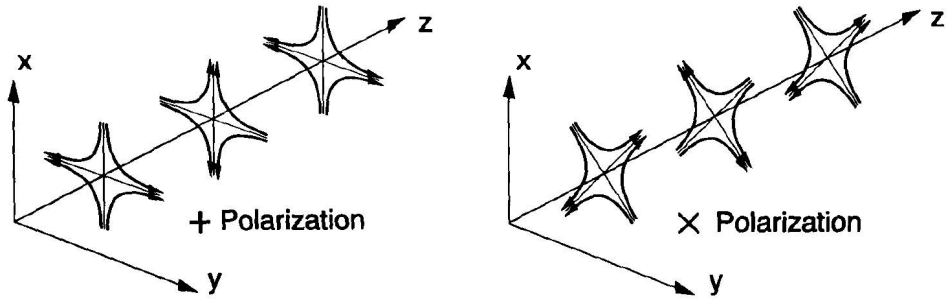

Figure 2.1: The lines of force associated with the two polarizations of a gravitational wave.

2.2 Ground-Based Laser Interferometric Detectors

According to the general relativity theory, a gravitational wave has two linear polarizations which are called *plus* (+) and *cross* (×). Associated with each polarization there is a gravitational wave field, h_+ or $h_\times$, which oscillates in time and propagates with the speed of light. Each wave field produces stretching and squeezing forces on any object through which it passes. If the object is small compared to the wavelength of gravitational waves, then the forces have the quadrupolar patterns shown in Figure 2.1. The names plus and cross are derived from the orientations of the axes that characterize the force patterns.

A ground-based laser interferometric detector consists of four masses hanging from vibration-isolated supports as shown in Figure 2.2. Furthermore the detector is equipped with an optical system for monitoring the separations between the masses. Two masses are near each other, at the corner of an "L", and one mass is at the end of each of the long arms of the L. The arm lengths are nearly equal, $L_1 \approx L_2 = L$. When a gravitational wave, with the high frequencies compared to the pendulum frequency of the masses, passes through the detector, it pushes the masses back and forth as if they were free from their suspension wires. As a result of that the

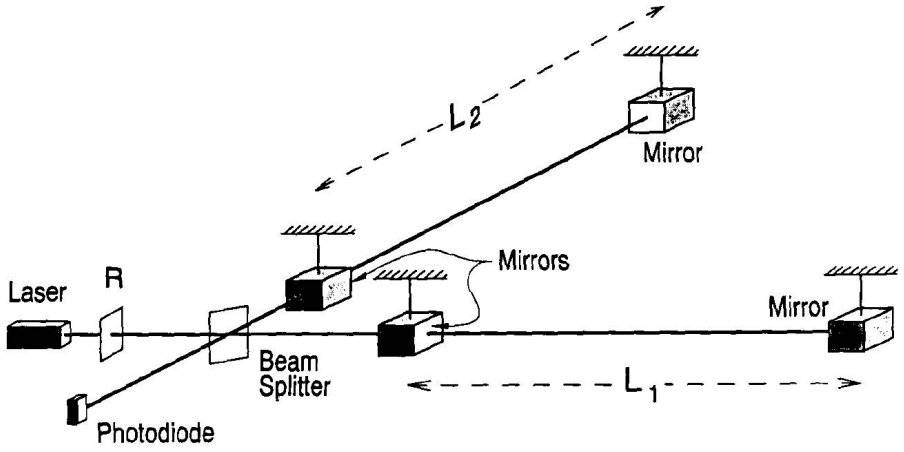

Figure 2.2: Schematic diagram of a ground-based laser interferometric detector.

difference between the lengths of the arms, $\Delta L \equiv L_1 - L_2$, changes over time. Using techniques of laser interferometry, this change is monitored in such a way that the variations in the output of the photodiode (the output of detector) are directly proportional to $\Delta L(t)$. Generally the output of the detector is a linear combination of the two wave fields:

$$\frac{\Delta L(t)}{L} = F_+ h_+(t) + F_\times h_\times(t) \equiv h(t).$$

The coefficients F_+ and $F_\times$ are called *antenna pattern* of the detector and depend on both the source and the detector. The combination $h(t)$ is called the *gravitational wave strain* that acts on the detector.

The strength of the gravitational waves can be estimated using the quadrupole formula as follow:

$$h \sim \frac{1}{c^2} \frac{4G(E_{kin}^{ns}/c^2)}{r}, \tag{2.1}$$

where c is the speed of light, G is the Newton's gravitation constant, E_{kin}^{ns} is the nonspherical part of kinetic energy of the gravitational waves and r is the distance of the source from the Earth. For highly compact, dynamical objetcs that radiate in the high frequency band, e.g. colliding and coalescing neutron stars and stellar-mass black holes, the nonspherical kinetic energy E_{kin}^{ns}/c^2 is of order the mass of the Sun. In particular, from (2.1) it follows that $h \sim 10^{-22}$ for such sources at the Hubble

distance (3000 Mpc, i.e. 10^{10} light years), $h \sim 10^{-21}$ at 200 Mpc, $h \sim 10^{-20}$ at the Virgo cluster of galaxies (15 Mpc) and $h \sim 10^{-17}$ in the outer reaches of our own Milky Way galaxy (20 kpc). These numbers set the scale of sensitivities that ground-based laser interferometric detectors seek to achieve: $h \sim 10^{-21}$ to 10^{-22}.

The major projects based on ground-based laser interferometric detectors are:

- **LIGO**. The Laser Interferometer Gravitational wave Observatory [49] consists of three operating detectors: a detector with four kilometer long arms in Livingston, Louisiana, as well as a pair of detectors with four and two kilometers long arms in Handford, Washington.

- **Virgo**. Virgo is an operating detector with three kilometer long arms built within a French-Italian collaboration in Cascina, near Pisa, Italy [50]. In most aspects, Virgo is quite similar to LIGO. A major difference is that Virgo employs a very sophisticated seismic isolation system that promises extremely good low frequency sensitivity.

- **GEO600**. GEO600 is a detector with six hundred meter long arms built within a German-English collaboration near Hanover, Germany [51]. Despite its shorter arms, GEO600 achieves a sensitivity comparable to the multi-kilometer instruments usign advanced interferometry techniques.

- **TAMA300**. TAMA300 is a detector with three hundred meter long arms near Tokyo [52]. It has been in operation for several years now and the TAMA team is currently designing a detector with three kilometer long arms, building on its experiences with the three hundred meter instrument.

- **ACIGA**. The Australian Consortium for Interferometric Gravitational wave Astronomy is currently building a detector with eighty meter long arms near Perth, Australia [53]. Such a detector would be particularly valuable to determine the location of sources on the sky since it is the only detector in the southern hemisphere.

Figure 2.3: The Virgo detector.

The sensitivity of each detector is limited by the presence of noise. There exist different sources of noise, but fundamentally the frequency band is limited by seismic noise, which gives a lower bound, and by shot noise, which is quantistic in nature and gives an upper bound [54]. However, for simplicity we assume that the noise is a *white noise*, i.e. a wide-stationary Gaussian stochastic process with mean 0 and variance 1 [55].

The problem considered in this thesis is connected with the Virgo detector. The construction of this laser interferometric detector was completed in June 2003 (see Figure 2.3). Currently, the project is in the "commissioning phase", i.e. it will run day and night listening to all gravitational signals which may arrive at any time and coming from any part of the Universe. The frequency range of Virgo extends from 10 to 6000 Hz. This range, as well as the very high sensitivity, should allow the detection of the gravitational radiation produced by supernovae and coalescence of binary systems in the Milk Way and in outer galaxies, for instance from the Virgo cluster of galaxies.

2.3 Gravitational Signals from Coalescing Binary Sistems

One of the most promising sources for ground-based laser interferometric detectors are coalescing binary systems, which consist of astrophysical compact objects such as neutron stars and/or black holes. The coalescence is the late stage in the evolution of a binary system. In this stage compact objects are under the influence of the strong gravitational fields of each other and are moving at relativistic speeds. For these reasons the dynamics of the system is very difficult to model. However, in the early stage it is possible to treat the problem of motion and to expand the general relativistic equations of motion and the wave generation formulas in a power series according to the post-Newtonian approximation [56].

For the purpose of detection it suffices to use the so-called restricted post-Newtonian approximation [57]. In this approximation the Post-Newtonian corrections to the amplitude of the gravitational signal are neglected, while the corrections to the phase are fully taken into account to the highest order possibile. In particular, we consider gravitational signals in the second-order restricted post-Newtonian approximation [58]. These signals represent the strain induced by gravitational waves at the detector and are modeled as *chirp* signals [59], i.e. signals in which the amplitude and the phase are functions increasing over time (see Figure 2.4)

$$h(t;\boldsymbol{\theta}) = A[\pi f(t-t_0;m_1,m_2)]^{(2/3)}\cos[\varphi(t-t_0)+\varphi_0] \tag{2.2}$$

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 signal, respectively, and m_1 and m_2 the masses of the coalescing binary system. The function $f(t-t_0;m_1,m_2)$ is the istantaneous gravitational wave frequency given implicitly by

$$t-t_0 = \tau_0\left[1-\left(\frac{f}{f_0}\right)^{-8/3}\right],$$

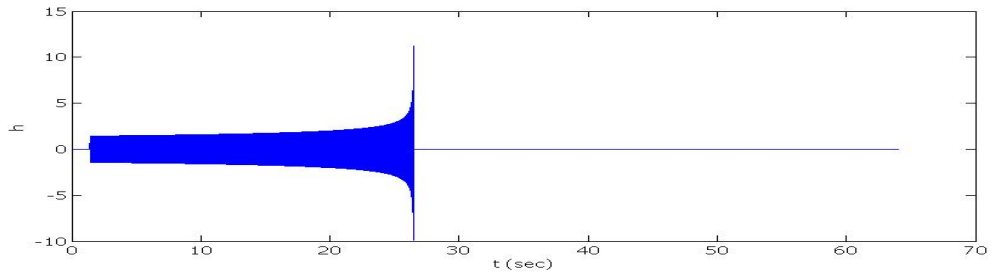

Figure 2.4: A gravitational signal with masses $m_1 = m_2 = 1.4$.

where f_0 is the lower frequency cutoff of the detector and τ_0 is a constant having the dimension of time:

$$\tau_0 = \frac{5}{256\pi f_0 \eta}(\pi M f_0)^{-5/3} \tag{2.3}$$

where $M = m_1 + m_2$ and $\eta = (m_1 m_2)/M^2$ is the total mass and the mass ratio of the binary system, respectively. The phase of gravitational signal $\varphi(t - t_0; m_1, m_2)$ is derived from the solution of a system of ordinary differential equations invonlving post-Newtonian approximations of the energy and the flux of binary system (see [60] for details).

2.4 Mathematical Formulation of the Detection Problem

The detection problem consists in deciding, through observation of the detector output, if a gravitational signal is present or not. Since the output is affected by noise, this decision requires the application of methods of the statistical inference theory. As a problem of decision, the detection problem is equivalent to one which, in statistical terminology, is called the problem of *testing hypotheses*: the hypothesis that the noise alone is present is to be tested, on the basis of the detector output, against the hypothesis that a gravitational signal is present. Generally the solution of the problem of testing hypotheses requires the knowledge of *a priori* information about the nature of signal and noise. In addition, it is necessary to establish a criterion with respect to which evaluating the "goodness" of a solution method.

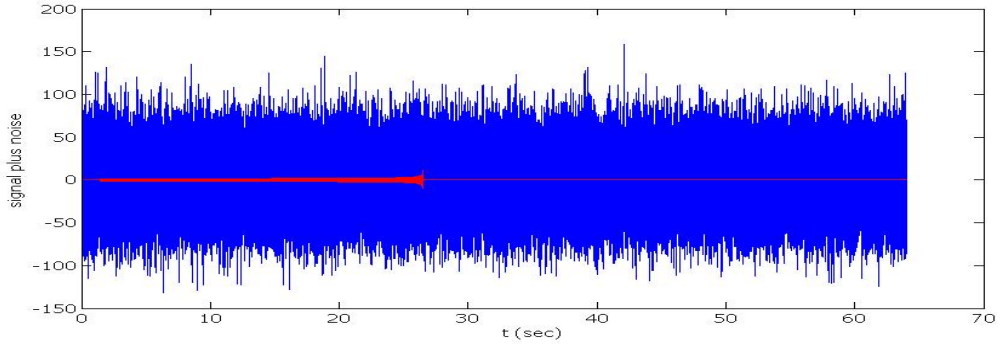

Figure 2.5: The output of the detector with noise plus gravitational signal with masses $m_1 = m_2 = 1.4$ and signal-to-noise ratio equal to 10.

In practice, the detector output is sampled with a certain time step, thus a segment of data is analized, which is an N-dimensional vector $\mathbf{x} = (x[0], \ldots, x[N-1])$; the corresponding sampled gravitational signal, if present, is an M-dimensional vector $\mathbf{h} = (h[0], \ldots, h[M-1])$, 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 $n_0 \in \{0, \ldots, N-M\}$. The set of detector outputs is a measurable space $(\mathbb{R}^N, \mathcal{B})$ where $\mathcal{B}$ is the Borel σ-algebra. This set is called *space of observations*. The detection problem can be written as

$$\begin{aligned} \mathcal{H}_0 : \quad & x[n] = w[n] && n = 0, ..., N-1 \\ \mathcal{H}_1 : \quad & x[n] = w[n] + h[n] && n = 0, ..., N-1 \end{aligned} \tag{2.4}$$

where $\mathbf{w} = (w[1], \ldots, w[N])$ is the noise of the detector. The hypothesis $\mathcal{H}_0$ means that the detector output is just noise, while the hypothesis $\mathcal{H}_1$ means that the detector output consists of gravitational signal plus noise. In Figure 2.5 we show an example of the output of the detector.

A *rule of decision* is a partition of the space of observations in two disjoint regions $\Gamma_0, \Gamma_1 \in \mathcal{B}$, $\Gamma_0 \cup \Gamma_1 = \mathbb{R}^n$, $\Gamma_0 \cap \Gamma_1 = \emptyset$ with Γ_0 corresponding to the hypothesis $\mathcal{H}_0$ and Γ_1 to the hypothesis $\mathcal{H}_1$. In practice, a rule of decision can be viewed as a

function $\delta : \mathbb{R}^n \rightarrow \mathbb{R}$ such that

$$\delta(\mathbf{x}) = \begin{cases} 0 \text{ if } \mathbf{x} \in \Gamma_0, \\ 1 \text{ if } \mathbf{x} \in \Gamma_1, \end{cases}$$

and we accept the hypothesis $\mathcal{H}_i$ if $\delta(\mathbf{x}) = i$ $(i = 0, 1)$. Four possible choices correspond to each rule of decision:

1. accept $\mathcal{H}_0$ when $\mathcal{H}_0$ is true;
2. accept $\mathcal{H}_0$ when $\mathcal{H}_1$ is true;
3. accept $\mathcal{H}_1$ when $\mathcal{H}_1$ is true;
4. accept $\mathcal{H}_1$ when $\mathcal{H}_0$ is true.

The first and third choices correspond to correct decisions; the second and fourth to incorrect ones. In the context of detection problems, the second and thrid choices are called *false alarm* and *correct detection* respectively.

The determination of a decision rule depends on the characteristics of the signal and the noise, but also on a priori knowledge about hypotheses. Since we cannot associate a cost with each possible choice and cannot assign a priori probabilities to the hypotheses, it is not possibile to use the criteria of Bayes and minimax [61]. In this situation an alternative is represented by the Neyman-Pearson criterion [61], which is based on the idea of maximizing the probability of correct detection subject to a chosen probability of false alarm. However, the application of this criterion requires that either the signal and the noise are completely known or it is possible to determine a *uniformly most powerful* test [62]. The first situation corresponds to the case in which the hypotheses test is *simple*, i.e. each hypothesis is associated with a single probability density. When one or both hypotheses are associated with a family of probability densities, depending on a certain set of parameters, the hypotheses test is called *composed*. A uniformly most powerful test has the property that the probabilities of false alarm and correct detection are independent

of the parameter set of the families of probability densities associated with the hypotheses. The determination of a uniformly most powerful test is possible in cases rarely met in applications. In our case the hypothesis test is composed, since the hypothesis $\mathcal{H}_1$ corresponds to a family of probability densities which depends on the parameters of the gravitational signal. Furthermore, because it is not possible to determine a uniformly most powerful test, we use the *generalized likelihood ratio test*, which consists of substituting the unknown parameters with a maximum likelihood estimates of them and of choosing the hypothesis $\mathcal{H}_1$ if

$$L(\mathbf{x};\hat{\boldsymbol{\theta}}) = \frac{p_1(\mathbf{x};\hat{\boldsymbol{\theta}})}{p_0(\mathbf{x})} > \gamma, \tag{2.5}$$

where $\hat{\boldsymbol{\theta}}$ is the maximum likelihood estimate of $\boldsymbol{\theta}$, $p_1(\mathbf{x};\boldsymbol{\theta})$ is the family of probability densities associated with the hypothesis $\mathcal{H}_1$, $p_0(\mathbf{x})$ is the probability density associated with the hypothesis $\mathcal{H}_0$, and γ is a suitable treshold which is related to probabilities of false alarm and correct detection. Since $\hat{\boldsymbol{\theta}}$ is the global minimizer of the density $p_1(\mathbf{x};\boldsymbol{\theta})$ and the density $p_0(\mathbf{x})$ is independent of the vector of parameters and positive, the test (2.5) can be written as

$$L(\mathbf{x};\hat{\boldsymbol{\theta}}) = \max_{\boldsymbol{\theta}} L(\mathbf{x};\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \frac{p_1(\mathbf{x};\boldsymbol{\theta})}{p_0(\mathbf{x})} > \gamma. \tag{2.6}$$

The main difficulties related to the application of the generalized likelihood ratio test are

- the solution of the global optimization problem in (2.6);
- the determination of the threshold γ.

The determination of the threshold is very difficult since, in general, a closed form expression of the probability density of the decision statistic in both hypothesis cannot be determined. However experimental considerations may allow to choose a suitable value of the threshold.

Since we assume that the noise of the detector is a white Gaussian stochastic process with mean 1 and variance 0, the (2.6) can be written as

$$\max_{\boldsymbol{\theta}} L(\mathbf{x};\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \exp\left[\sum_{n=0}^{N-1}(x[n]-h[n])^2 - \sum_{n=0}^{N-1} x^2[n]\right] > \gamma \tag{2.7}$$

It is possible to show that maximum likelihood estimates of the amplitude A and the initial phase φ_0 can be analytically computed [57]. Substituting this estimates and taking the logarithm of each member of test (2.7), we have

$$\log L(\mathbf{x}; \hat{A}, \hat{\varphi}_0, m_1, m_2, n_0) = \max_{m_1,m_2,n_0} \left[\left(\sum_{n=0}^{N-1} x[n] q_0[n-n_0] \right)^2 + \left(\sum_{n=0}^{N-1} x[n] q_{\pi/2}[n-n_0] \right)^2 \right] > \gamma' \tag{2.8}$$

where $\gamma' = \log \gamma$ and $\boldsymbol{q}_0 = (q_0[0], \ldots, q_0[N-1]), \boldsymbol{q}_{\pi/2} = (q_{\pi/2}[0], \ldots, q_{\pi/2}[N-1])$ are the sampled versions of the *quadrature components* of the gravitational signal, normalized with respect to the Euclidean norm in $\mathbb{R}^N$. The quadrature components of the gravitational signal $h(t; \boldsymbol{\theta})$ are defined as

$$h_0(t; m_1, m_2) = [\pi f(t; m_1, m_2)]^{2/3} \cos[\varphi(t; m_1, m_2)],$$
$$h_{\pi/2}(t; m_1, m_2) = [\pi f(t; m_1, m_2)]^{2/3} \cos[\varphi(t; m_1, m_2) + \pi/2]$$

Furthermore, it can be shown that the maximization in (2.8) can be carried out separately for (m_1, m_2) and n_0 [57], thus the determination of $\log L(\mathbf{x}; \hat{\boldsymbol{\theta}})$ requires the solution of the following box-constrained global optimization problem:

$$\underset{(m_1,m_2)\in\Omega}{\text{maximize}}\, F(m_1, m_2), \tag{2.9}$$

where

$$\Omega = \{(m_1, m_2) \in \mathbb{R}^2 \ : \ l \leq m_1, m_2 \leq u\}, \tag{2.10}$$

$$F(m_1, m_2) = \sqrt{\max_{n_0 \in \{0,\ldots,N-M\}} \left(C_0^2(n_0, m_1, m_2) + C_{\pi/2}^2(n_0, m_1, m_2) \right)}, \tag{2.11}$$

and $C_0(n_0, m_1, m_2)$ and $C_{\pi/2}(n_0, m_1, m_2)$ are the correlations between $\mathbf{x}$ and the normalized quadrature components of the gravitational signal $\mathbf{q}_0, \mathbf{q}_{\pi/2}$:

$$C_0(n_0, m_1, m_2) = \sum_{k=n_0}^{n_0+M-1} x[k] q_0[k-n_0], \tag{2.12}$$

$$C_{\pi/2}(n_0, m_1, m_2) = \sum_{k=n_0}^{n_0+M-1} x[k] q_{\pi/2}[k-n_0] \tag{2.13}$$

The correlations in (2.12) and (2.13) are carried out through the application of Discrete Fourier Transform (DFT), using the property that the DFT of the correlation

of two signals is equal to product between the DTF of each signal [63]. We note that this property is true only we use the zero-padding strategy, which consists of extending the signals $\mathbf{x}, \mathbf{q}_0, \mathbf{q}_{\pi/2}$ with zeros for obtaining signals with length $2N-1$.

Following [59] we define the Signal-to-Noise ratio (SNR) as

$$SNR \equiv E\{F(m_1, m_2)\} = \sqrt{\max_{n_0 \in \{0,\ldots,N-M\}} \left((\sum_{k=n_0}^{n_0+M-1} s[k] q_0[k-n_0])^2 + (\sum_{k=n_0}^{n_0+M-1} s[k] q_{\pi/2}[k-n_0])^2 \right)} \tag{2.14}$$

where $E\{\cdot\}$ is the mean value with respect to infinite realizations of the noise, and $\mathbf{s} = (s[0], \ldots, s[N-1])$ is the gravitational signal contained in the output of the detector which masses are $(\bar{m}_1, \bar{m}_2)$. We note that in (2.14) the output of the detector $\mathbf{x}$ is substituted with the gravitational signal $\mathbf{s}$. In the context of signal theory the function $F(m_1, m_2)$ is a *matched filter* [63], which has the property that the maximum SNR is obtained when $m_1 = \bar{m}_1$ and $m_1 = \bar{m}_2$, i.e. the masses of the signal $\mathbf{s}$ are equal to masses of the normalized quadrature components $\mathbf{q}_0, \mathbf{q}_{\pi/2}$ of a gravitational signal $\mathbf{h}$, which is called *template*. According this property, the matched filtering technique has been developed which consists of determining the template that maximize the SNR. We note that this determination is equivalent to solution of the global optimization problem in (2.9) [57]. In Section 2.5 we will briefly describe how contructing a finite family of template, i.e. a discretization of the feasible domain, in such a way that the maximum SNR with respect this family is not lower than a given percentage of the maximum SNR.

In this thesis we focus on the solution of the optimization problem (2.9), which is the most critical issue in the application of the generalized likelihood ratio test. The objective function F is a stochastic function, since the presence of noise. However, for each realization of the noise, the objective function F becomes a deterministic function and it can be shown that the problem (2.9) has a solution according to Theorem 1.2.1. Its solution is a difficult task, because the objective function F is highly nonlinear and with many local maxima (see Figure 2.6), and its derivatives are not available. Furthermore, its evaluation is computationally expensive, since it

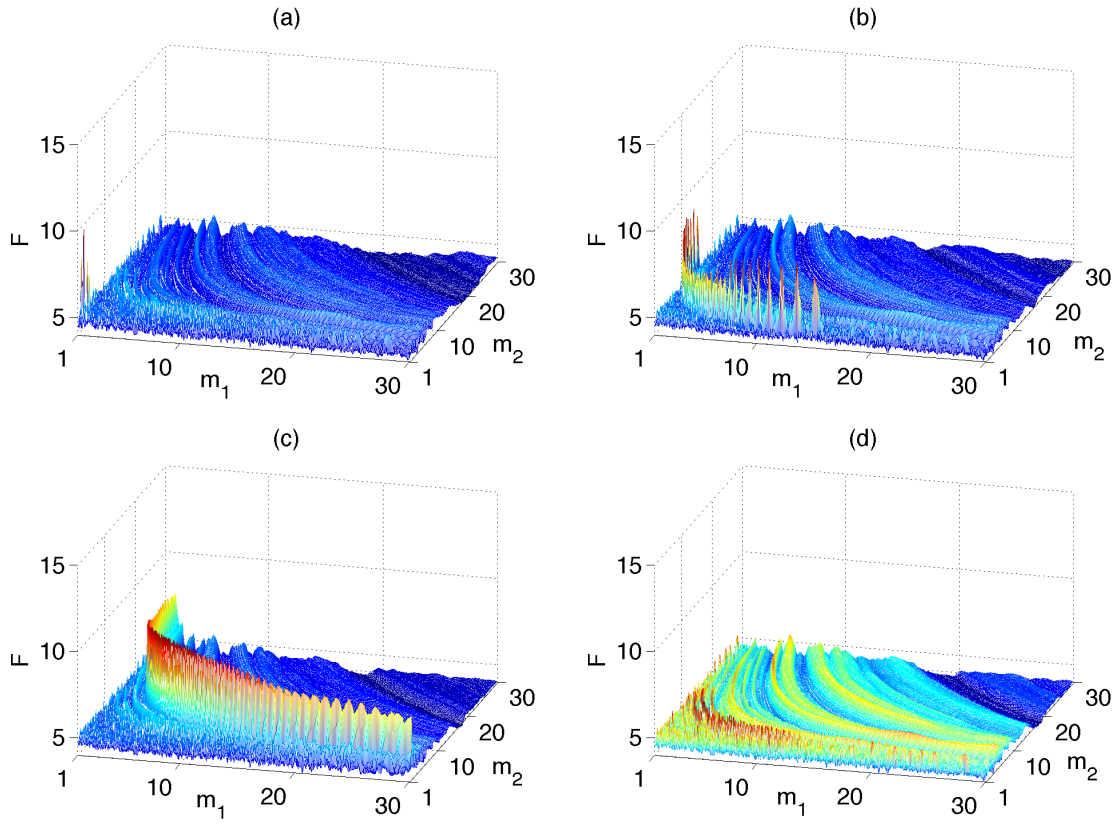

Figure 2.6: 3D plot of the objective function F, in case of noise plus gravitational signal from a binary system with masses $m_1 = m_2 = 1.4\ M_\odot$ (a), $m_1 = 1.4\ M_\odot$ and $m_2 = 10\ M_\odot$ (b), $m_1 = 5\ M_\odot$ and $m_2 = 10\ M_\odot$ (c), and in case of noise only (d). The SNR is equal to 10. $M_\odot$ denotes the solar mass.

requires the solution of two ordinary differential equations (ODEs) to generate the quadrature components of gravitational signal, and the execution of three FFTs of length N to compute the correlations of $\mathbf{x}$ with them. Common values of N are $O(10^5)$; the time for solving the ODEs depends on the masses of the gravitational signal (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).

2.5 Grid Search Method

In the astrophysics community, the most widely used method for solving problem (2.9) is the grid search. We note the this grid search is different to grid search de-

scribed in Section 1.3. Indeed, the physical knowledge of problem allows to construct only one grid of points in the feasible domain for guaranteeing a fixed "accuracy". Specifically, we 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. We recall that the discretization of feasible domain corresponds to choosing a finite family of template $\{\mathbf{h}(m_1, m_2)\}$, where $(m_1, m_2) \in \mathcal{K}$, $\mathcal{K} \subset \Omega$ and the cardinality $|\mathcal{K}| = k$. If we assume that $\mathbf{s} = \bar{A}\tilde{\mathbf{s}}$ is the gravitational signal contained in the output of the detector, with amplitude $\bar{A}$, $\tilde{\mathbf{s}}$ normalized with respect to Euclidean norm and with masses $(\bar{m}_1, \bar{m}_2)$, we have that [57]

$$SNR = E\{F(m_1, m_2)\} = \bar{A} \cdot \mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2)$$

where

$$\mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2) = \sqrt{\max_{n_0 \in \{0,\dots,N-M\}} \left(\Big(\sum_{k=n_0}^{n_0+M-1} \tilde{s}[k] q_0[k-n_0] \Big)^2 + \Big(\sum_{k=n_0}^{n_0+M-1} \tilde{s}[k] q_{\pi/2}[k-n_0] \Big)^2 \right)}.$$

We note that $0 \leq \mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2) \leq 1$, and $\mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2) = 1$ when $m_1 = \bar{m}_1$ and $m_2 = \bar{m}_2$, i.e. when the SNR is maximum. So the function $\mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2)$ is the fraction of the maximum SNR reduced when the template $\mathbf{h}$ is used to search the signal $\mathbf{s}$.

The aim is constructing the family $\{\mathbf{h}(m_1, m_2)\}$ in such a way that

$$\min_{(\bar{m}_1, \bar{m}_2) \in \Omega} \max_{(m_1, m_2) \in \mathcal{K}} \mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2) \geq MM \tag{2.15}$$

where $0 \leq MM \leq 1$ is a fixed number, called *minimal match*. This guarantee that for each signal $\mathbf{s}$ we have

$$SNR = \max_{(m_1, m_2) \in \mathcal{K}} E\{F(m_1, m_2)\} =$$

$$\bar{A} \cdot \mathcal{O}(m_1, m_2, \bar{m}_1, \bar{m}_2) \geq \bar{A} \cdot MM = \max SNR \cdot MM,$$

therefore the mean value of the maximum of F is not lower than a given percentage (minimal match) of the maximum of SNR.

In order to construct the family of template (grid of points), a metric tensor is defined in the space of *chirp time* τ_0 and τ_3 by the following formula [64]:

$$g_{ij} = -\frac{1}{2}\frac{\partial^2 \mathcal{O}}{\partial \tau_i \partial \tau_j}$$

where $i, j \in \{0, 3\}$. The chirp times can substitute the masses for individuating a gravitational signal. The value of τ_0 is given by (2.3) and τ_3 is defined as

$$\tau_3 = \frac{1}{f_0 \eta}(\pi M f_0)^{-2/3}$$

where the significance of f_0, η, M is given in Section 2.3. These chirp time can also be inverted in terms of the masses:

$$M = \frac{5}{32\pi^2 f_0}\frac{\tau_3}{\tau_0}, \qquad \eta = \frac{1}{8\pi f_0 \tau_3}\left(\frac{32\pi\tau_0}{5\tau_3}\right)^{2/3}$$

and the correspondence between the two sets of parameters (m_1, m_2) and (τ_0, τ_3) is illustred in Figure 2.7. The advantage of the space of the chirp times is that the metric tensor is locally constant in this space. This allow to determine a rectangular lattice of point which satisfied the relation (2.15) (for details see [57]).

Generally, a minimal match of at least 97% is required, leading to a large number of templates, i.e. grid points and hence of objective function evaluations; e.g., a grid of 27379 points is needed to get a minimal match of 97% over the space of masses $[1, 30] \times [1, 30]$, which represents our feasible domain (see Figure 2.8). 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 (2.9), while achieving a comparable accuracy in the mean value of the maximum, is a main goal in the application of the generalized likelihood ratio test, since it increases the number of data segments that can be analyzed. For this reason, 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

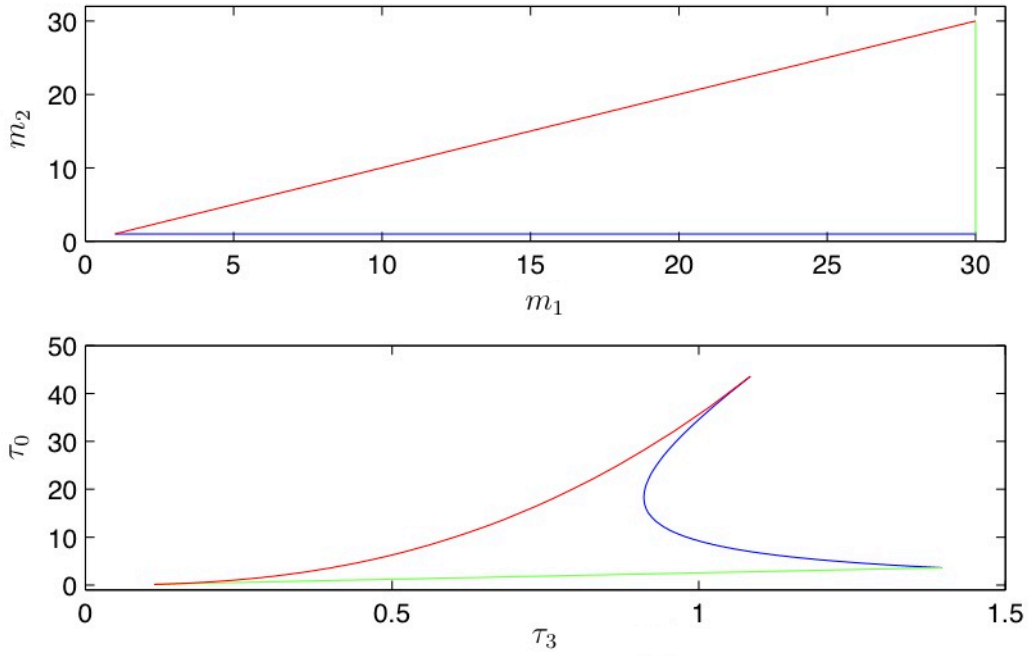

Figure 2.7: The correspondence between the two sets of parameters (m_1, m_2) and (τ_0, τ_3).

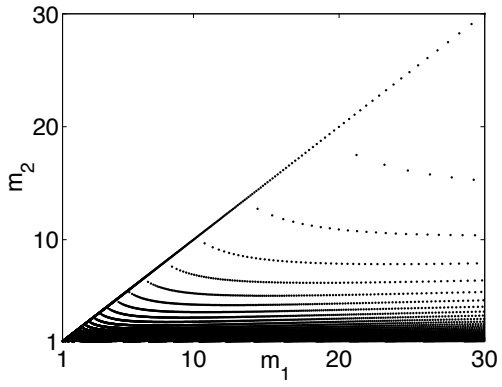

Figure 2.8: Grid ensuring a minimal match of 97% over the domain $[1, 30] \times [1, 30]$ (27379 points).

is then used on the sub-domains to find a good approximation of the maximum [48, 59, 65]. 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.

Chapter 3

A Genetic Algorithm for the Detection Problem

We first provide a general description of genetic algorithms, outlining their structure and main components. Then we describe a real-coded genetic algorithm that we developed for the problem under consideration, showing how the information on the problem has been exploited to design the genetic operators. We report numerical experiments performed on a representative set of test problems, showing that the developed genetic algorithm allows a strong reduction of the computational cost with respect to the grid search, which is generally used to solve this problem. A comparison with other global optimization algorithms show that the genetic algorithm is much more efficient on the most significant and difficult set of problem instances.

3.1 Introduction

The idea of simulating natural evolution as a tool for modelling and analyzing complex systems becames possible in the 1960s, when relatively inexpensive digital computers started to be available [66]. With different approaches three paradigms have been developed nearly simultaneously: *evolutionary strategies* for solving difficult

optimization problem with continuous real variables [67]; *evolutionary programming* in the context of artificial intelligence [68] and *genetic algorithms* for studying the behaviour of adaptive systems that are capable of dealing with an uncertain and changing environment [69, 70]. Nowadays these paradigms are considered instances of a more general class of methods that are called *evolutionary methods* [66, 71].

The genetic algorithms are the most popular and extended class of evolutionary methods. They have received increasing attention by researchers and have been applied with success to many problems in science and engineering [72]. These algorithms are able to compute satisfactory solutions of global optimization problems, especially when the objective function is a noisy or multiextremal black box for which little or no additional information, such as derivatives, is available.

A genetic algorithm is an iterative method that operates on a population of individuals that represent potential solutions to a given problem. Each individual in the population is assigned a measure of its "goodness" through a *fitness* function, which, in the context of optimization, is the objective function. Initially, a population of individuals is randomly generated. Then the evolution of this population is guided through the application of genetic operators, which mimic the corresponding natural process. In practice these operators are applied iteratively to obtain better and better populations. In solving optimization problems, a solution is given by the fittest individual after the last evolution step.

Many issues occur in the development of a genetic algorithm: representations of individuals, choice of initial population, design of the genetic operators and their tuning to the problem to be solved. The effectiveness of a genetic algorithm strongly depends on how these issues are dealt with. Furthermore, incorporation of a priori information on the specific problem usually leads to improvements in the performance of the algorithm. On the other hand, the main disadvantage of genetic algorithms is the very limited possibility of developing a theory of convergence. Indeed the study of dynamics of genetic algorithms represent a difficult challenge from theoretical point of view. Many approaches have been investigated [73], but they

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

Figure 3.1: Basic structure of a genetic algorithm.

concern a "vanilla" version of a genetic algorithm that does not correspond to many versions meet in the applications [72]. For this reason an experimental analysis is needed for evaluating the effectiveness of genetic algorithms.

A detailed description of genetic algorithms is beyond the scope of this thesis. In the section 3.2 we present briefly a general description of genetic algorithms and we refer to [74, 75] for a deep description.

3.2 Genetic Algorithms

A genetic algorithm is an heuristic search algorithm based on the principles of natural evolution and genetics. Basically, a genetic algorithm generates a sequence of populations $P_t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_N^t\}$, where $t = 0, 1, 2, \ldots$ identifies the so-called generation. The evolution of the populations $\{P_t\}_{t\in\mathbb{N}}$ is guided through *genetic operators* such as *selection*, *recombination* and *mutation*, which mimic the corresponding natural processes. The basic structure of a genetic algorithm is outlined in Figure 3.1.

Optimization is a main application area of genetic algorithms. The reason for this is the analogy between evolutionary processes and optimization problems. In this context genetic algorithms belong to class of *incomplete methods*. The development

of a genetic algorithm involves the following issues:

- *representation* of the individuals;
- choice of an *initial population*;
- mechanism for the *selection* of parents;
- *recombination* of parents for producing offspring;
- mechanism for the *replacement* of parents;
- *mutation* of individuals;
- choice of the parameters of the algorithm (size of the population, probability of recombination, probability of mutation, etc.).

Representation of the Individuals

The choice of an appropriate representation of individuals is a fundamental issue in the development of a genetic algorithm. It is strictly associated with the choice of genetic operators and for this reason it affects the way in which the algorithm explores the feasible domain.

In early genetic algorithms each individual was represented with a fixed length string of symbols from the alphabet $\mathcal{A} = \{0, 1\}$ (*binary representation*) [76]. In this case the original problem is transformed into a discrete problem and the connection between the original and the transformed problem is established by an invertible function called *encoding function*:

$$g : \mathcal{A}^l \to A,$$

where A is the feasible domain and l is the length of the string. The inverse of this function, called *decoding function*, is needed for decoding any individual and for evaluating its fitness using the objective function. Each component of a string

is called *gene* and its possible values are called *alleles*. The space $\mathcal{A}^l$ is *the genotype space* and A is the *phenotype space*.

The use of the binary representation has been motivated by an intense research activity devoted to analyze the dynamics of genetic algorithms. Different approaches have been investigated that are based on the schema theory [77], the dynamical system theory [78], the Markov chain theory [79] and statistical mechanics [80]. These approaches have been allowed to obtain some results about convergence of a "vanilla" version of genetic algorithms. However because this vanilla version does not provide "good" performance in many problems meet in the applications, several variants have been developed at which do not apply the previous convergence results. In particular it has been investigated alternative representations which has been turned out more adequate for specific class of problems [74]. Among these the *real-coded* representation [81], in which every individuals is a feasible point, have been received very attention in the context of continuous optimization problems. This representation will take again in the following.

Choice of the Initial Population

The generation of the initial population has been little investigated with respect to other issues of the genetic algorithms. Generally, each individual of an initial population P_0 is randomly chosen from a uniform distribution, using some pseudo-random number generator. Just recently a numerical evidence has been provided that the initial population may strongly affect the behaviour of the algorithm 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 [82]. This combination allows to improve both the speed of convergence of the genetic algorithm and the goodness of the computed solution.

Selection of the Parents

The selection is one of the main operators used in the genetic algorithms. The purpose of this operator is to emphasize the best individuals in a population. It does not produce new individuals, but it choose, from the current population P_t, a mating pool P'_t of relatively good individuals that will generate offspring through the recombination. The goodness of each individual is measured through the fitness function. The basic idea of the selection is that individuals with better fitness must have a higher probability to be chosen. The choice of an appropriate selection operators is fundamental for obtaining satisfactory results and depends on characteristics of considered problem.

Several selection operators have been devised which differ in the way the number of copies are associated to better individuals. Selection operators can be divided in two groups [83]:

- *deterministic selection operators*, which sort the population according to their fitness and deterministically choose the better individuals;
- *stochastic selection operators*, which assign a probability of selection to each individual according to their fitness and choose the best individuals depending on a probability ditribution.

Recombination of the Parents

The aim of recombination operator is to combine the features of the parents to form offspring, with the possiblity that two good parents may generate a better individual. The recombination is a function

$$\mathcal{R} : P'_t \times P'_t \to O_t,$$

where P'_t is the mating pool of selected individuals at generation t and O_t is the offspring set. This operator is not applied to all the individuals of the mating pool. The number of actual parents dependes 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.

Several recombination operators have been developed which can be divided in three groups [84]:

- *discrete crossover operators*. With these operators the value of each gene in the offspring concides with the value of this gene in one of the parents, i.e. the values of the genes in the parents are not transformed for obtaining the values of the genes in the offspring. Example of these operators are *simple*, *two-point* and *uniform* crossover operators;

- *aggregation-based crossover operators* which use a deterministic function that combines the values of the gene of the parents to generate the value of genes of offspring. The *arithmetical*, *geometrical* and LX operators are representatives of this group;

- *neighborhood-based crossover operators* which determine the genes of the offspring extracting values from intervals definined by neighborhoods associated with the genes of the parents through probability distributions. Example are BLX-α, *simulated binary crossover* and FR operators.

Replacement of the Parents

In order to choose which offspring will survive, two main approaches can be adopted: the *overlapping-generation* model and the *nonoverlapping-generation* model [66]. In the former case, the parents in the mating pool P'_t and the offspring O_t will compete with each other for survival producing a new population P''_t. In the latter, all parents P'_t die at each generation t and the offspring O_t compete for survival producing a new population P''_t .

Mutation

The mutation operator is asimed at randomly altering some individuals in the population, in order to introduce genetic diversity. It is an unary operator which creates a new individual by changing an existing individual. Therefore it is a function

$$\mathcal{M} : P_t'' \to P_{t+1}$$

where P_t'' is the population obtained after the replacement and P_{t+1} the mutate population. Generally the mutation is applied at the end of each generation, so the mutated population is the population at successive generation, hence the subscript $t+1$. The mutation is applied on single gene of an individual $\mathbf{x}_i^t$. 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 $r < P_M$.

Several mutation operators have been proposed which differ for the region of the feasible domain containing the new individual. For example, in *uniform mutation* [74] this region is the feasible domain, in *real number creep mutation* [85] the size of the region can be changed according to a parameter and *nonuniform mutation* [74] the size of the region descreases when the number of generations increases.

3.3 Development of a Genetic Algorithm for the Detection Problem

In Section 3.2 we described briefly the main components of a genetic algorithm. Now we present in detail the algorithm that we designed for problem (2.9) In particular we show how the specific knowledge on the problem has been incorporated into the developed algorithm.

3.3.1 Representation of the Individuals

When the genetic algorithms are used to solve continuous optimization problems the binary representation meets certain difficulties. The main difficulty is known as *Hamming cliff* [86]. This happens when the distance between two individuals in the genotype space is very different to the distance in the phenotype space. Numerical experiments have shown that this may produce problems under some conditions, such as the convergence towards no global minimizers [86]. The Hamming cliff can be solved by using the Gray code [86], however this approach does not improve the effectiveness of the algorithm. In order to solve this problem, the use of real representation has been examined. Theoretical investigations have shown that the good properties of genetic algorithms do not depend on the binary representation. Specifically, different approaches have been examined to analyze the dynamics of a genetic algorithm with a real representation, such as generalizations of the schema theory [87], virtual alphabet theory [88] and stochastic process theory [89, 90]. Furthermore numerical experiments have shown that the use of a real representation allows to improve both the accuracy of the solution and the efficiency of the genetic algorithms [91]. Indeed, the main advantages of this representation are the ability of performing a local tuning of the solutions and the possibility of avoiding the coding/decoding of individuals. For these reasons we choose to use the real representation in which an individual is a point $\boldsymbol{m} = (m_1, m_2)$ of Ω and its genes are the single masses m_i, $i = 1, 2$.

3.3.2 Initial Population

As already observed, a good initial population should combine genetic diversity with uniform coverage [82]. On the other hand, in our problem, a suitable choice of the templates for the matched filtering technique leads to a nonuniform discretization of the feasible set, with points clustered in the areas where the objective function shows greater variability (see Figure 2.8). 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 introduce the information provided by the grid is to randomly select the individuals from a grid G corresponding to a widely accepted value of minimal match, such as 97%. In order 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 [82]. 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. An example of this stragegy is showed in the figure 3.2. The parameter b is chosen to guarantee that no large areas of the feasible domain are left uncovered by the initial population. Note that $b = 0$ corresponds to an initial population randomly selected from G, whereas $b^2 = N_p$ corresponds to an initial population resulting only from the grid-based NSS. Because of the symmetry of the global optimization problem with respect to m_1 and m_2, only the triangle $m_1 \geq m_2$ of Ω, and the elementary boxes covering this triangle, 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 using the following formula:

$$q = 1 + int(rnd(0,1) \cdot N_p), \tag{3.1}$$

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.

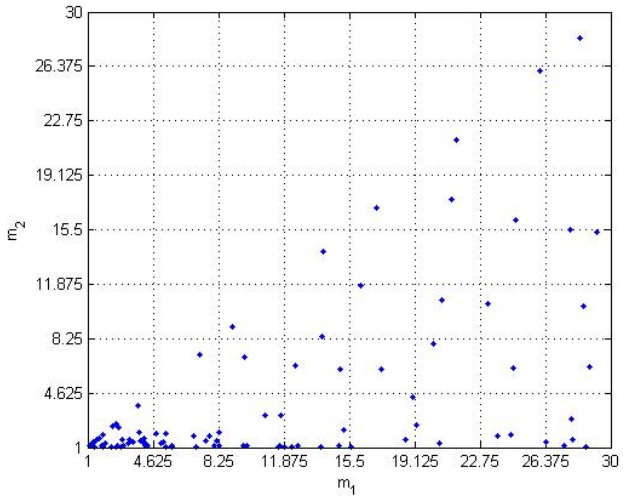

Figure 3.2: Example of the devised strategy for the choice of initial population.

3.3.3 Selection of Parents

The selection of parents to form the mating pool 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.

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 [92]. If the takeover time is large then the selective pressure is small, and vice versa. The selective pressure provides information on the number of generations after which the mutation becomes the primary operator of exploration. For our problem, we choose the *binary tournament without replacement*, that has a medium selective pressure with respect to other selection operators [93], 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 best 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 according to the rule (3.1), where the current number of individuals is used instead of N_p at each step of the tournament.

3.3.4 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. The pair of parents are formed by taking 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 [86, 84]. 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

$$m_i^j = rnd(I_i) \quad (i = 1, 2;\ j = 1, 2, 3), \tag{3.2}$$

where the action interval I_i is defined as

$$I_i = [g_i - \alpha M_i, G_i + \alpha M_i],$$

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 $|\alpha| < 1$.

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. In particular, $\alpha > 0$

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 that have been already visited. For $\alpha = 0$ the so-called flat recombination is obtained, in which m_i^j is randomly chosen between the corresponding genes of its parents. In our problem we considered $\alpha = 0.5$, a choice which allows to balance exploration and exploitation [86], 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 (2.10). 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.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 (3.2), 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.

3.3.5 Replacement of Parents

The overlapping-generation model is 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.

3.3.6 Mutation

In order to explore the feasible domain uniformly in the first generations, and locally in the later generations, we use the non-uniform mutation described in [74]. In this case, a gene m_i to be mutated becomes a new gene m_i^{new} according to the following formula:

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

where r is a random number taken from a uniform distribution in $(0, 1)$ and $\Delta(k, y)$ is defined as

$$\Delta(k, y) = y \cdot \left(1 - r^{\left(1 - \frac{k}{N_G}\right)^2}\right),$$

where k is the number of generations obtained so far and N_G is the maximum number of generations of the GA.

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 parents, i.e. we preserve a copy of the best individual by avoiding mutating it.

3.4 Numerical Experiments

3.4.1 Evaluation of the Genetic Algorithm

Extensive computational experiments were carried out to evaluate the effectiveness of our Genetic Algorithm (GA) approach in the solution of the considered 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 GA described in the previous section.

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 [57]:

- $\bar{m}_1 = \bar{m}_2 = 1.4M_\odot$ (two neutron stars),
- $\bar{m}_1 = 1.4M_\odot$ and $\bar{m}_2 = 10M_\odot$ (one neutron star and one black hole),
- $\bar{m}_1 = 5M_\odot$ and $\bar{m}_2 = 10M_\odot$ (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 131072, while the length M of the signal varies with the masses (51207 for $\bar{m}_1 = \bar{m}_2 = 1.4$, 10823 for $\bar{m}_1 = 1.4$ and $\bar{m}_2 = 10$ and 3216 for $\bar{m}_1 = 5$ and $\bar{m}_2 = 10$). 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 [94], 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 $\bar{m}_1 = \bar{m}_2 = 1.4$, 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 [95]. These problems are also the most difficult to be solved, as shown by the results of the computational experiments 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 [96], as implemented in the GNU Scientific Library (version 1.11). The tests were run on a Linux PC with an Intel Core 2 DUO E7300 processor, clock frequency of 2.66 GHz, 4 GB of RAM, and 3 MB of cache memory; it runs Linux Ubuntu 8.10, 64 bit version, with kernel 2.6.27. For each set of test problems, the GA was run using 30 different seeds for initializing

parameter	value
N_P (number of individuals)	100
P_R (probability of recombination)	0.7
P_M (probability of mutation)	0.05
N_G (maximum number of generations)	50

Table 3.1: Values of the GA parameters

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, 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 detection of gravitational wave problem [97]. 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 in Table 3.1. These values were selected mainly on the basis of computational experiments, which are 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 [98, 99]. 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:

- 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);

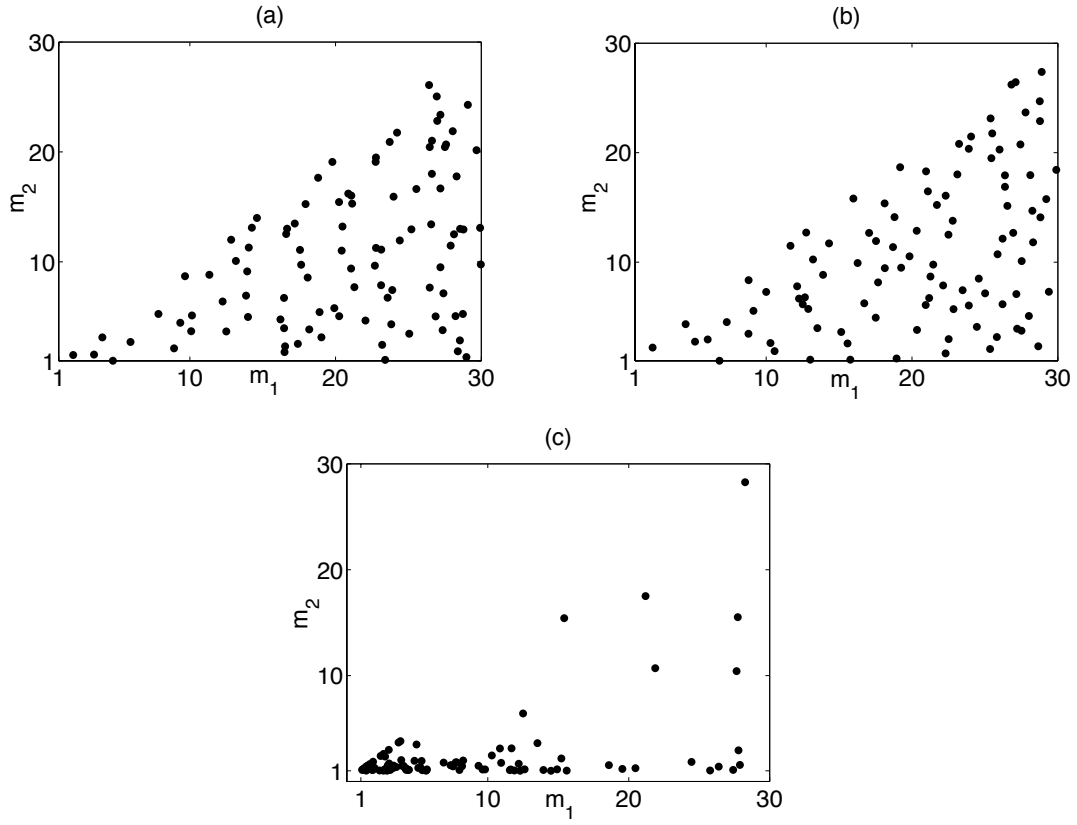

Figure 3.3: Initial populations of 100 individuals generated by using the RAND (a), RAND-NSS (b) and RAND-GRID (c) strategies.

- 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 section 3.3.2 (RAND-GRID).

Initial populations generated with these three strategies are shown in Figure 3.3. Table 3.2 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 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 other hand, a very strong improvement can be observed for $\bar{m}_1 = \bar{m}_2 = 1.4$ 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 $\bar{m}_1 = 1.4$ and $\bar{m}_2 = 10$, whereas it does not produce any improvement the case $\bar{m}_1 = 5$ and $\bar{m}_2 = 10$. Similar results hold if PBLX-0.5 is used as recombination operator, as shown in Table 3.3. However, we note that for $\bar{m}_1 = \bar{m}_2 = 1.4$ 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 Section 3.3.4. The corresponding results are reported in Table 3.4 for SBLX-0.5 and in Table 3.5 for PBLX-0.5. By looking at Table 3.4, we see that $b = 0$ and $b = 4$ lead to very close results for all the test sets; their

$\bar{m}_1$	$\bar{m}_2$	*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

Table 3.2: GA behaviour with different strategies for the selection of the initial population. SBLX-0.5 is used as recombination operator.

$\bar{m}_1$	$\bar{m}_2$	*init. population*	*fmean*	*fstd*	*success (%)*	*relerr*
1.4	1.4	RAND	8.3385	2.0264	59.9	0.170
		RAND-NSS	8.2689	2.0305	57.3	0.177
		RAND-GRID	9.7036	1.1397	92.8	0.034
1.4	10	RAND	9.4983	1.1781	92	0.076
		RAND-NSS	9.4099	1.2029	92.2	0.085
		RAND-GRID	10.0773	1.1449	96.4	0.020
5	10	RAND	10.2667	1.0384	98.8	0.002
		RAND-NSS	10.2758	1.0316	99.3	0.001
		RAND-GRID	10.1286	1.0717	97.8	0.015

Table 3.3: GA behaviour with different strategies for the selection of the initial population. PBLX-0.5 is used as recombination operator.

behaviour is satisfactory, as shown by the mean value of the computed maximum, which, in the problem at hand, 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 $\bar{m}_1 = \bar{m}_2 = 1.4$ 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 $\bar{m}_1 = \bar{m}_2 = 1.4$, while slightly improves it for $\bar{m}_1 = 5$ and $\bar{m}_2 = 10$. The previous comments apply also to the results in Table 3.5, 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 carried out by using the BLX-0 operator, which never violates the box constraints, led to poor percentages of success.

We finally observe that the mean and the maximum number of objective function evaluations over all the experiments, in N_G generations of the GA, are 5821 and 5914, respectively, i.e. less than 22% of those required by the grid search (27379). 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 3.4 and 3.5. Therefore, the GA approach allows a significant saving of the computational time with respect to the grid search.

3.4.2 Comparison with other Global Optimization Algorithms

The GA was compared with three global optimization algorithms: Price's controlled random search (CRS) [100], particle swarm pattern search (PSwarm) [101], and DI-

SBLX-0.5								
$\bar{m}_1$	$\bar{m}_2$	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

Table 3.4: GA behaviour with the RAND-GRID strategy, varying the parameter b in the grid-based NSS, and with the SBLX-0.5 recombination operator.

RECT [40]. 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 thesis; 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

PBLX-0.5								
$\bar{m}_1$	$\bar{m}_2$	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

Table 3.5: GA behaviour with the RAND-GRID strategy, varying the parameter b in the grid-based NSS, and with the PBLX-0.5 recombination operator.

evaluations is achieved. In our experiments the previous tolerance and maximum number were set to 10^{-4} and 30000, 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 [101] for details). A maximum number of 30000 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 [102]. A maximum number of 30000 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} [40]. The possibility of choosing between an initial population randomly extracted from a uniform distribution (default choice) or generated using the RAND-GRID strategy was added to CRS and PSwarm.

A further stopping criterion was introduced in the three previous implementa-

tions, which was combined with the other criteria through a logical "or":

$$F_{GA} - F_{MAX} < F_{GA} \cdot TOL, \tag{3.3}$$

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 the SBLX-0.5 recombination operator) 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 $\bar{m}_1 = 5$ and $\bar{m}_2 = 10$, $b = 0$ 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 performed to compute a solution "close" to the GA solution.

The previous optimization solvers were run on all the test problems described in Section 3.4.1. 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. CRS and PSwarm generally perform better with the RAND-GRID initial population, therefore we discuss the results obtained with this strategy.

As shown by the results in Table 3.6, CRS and DIRECT are more efficient than the GA on the test set corresponding to $\bar{m}_1 = 5$ and $\bar{m}_2 = 10$, since they satisfy criterion (3.3) 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 criterion (3.3) 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 criterion (3.3) satisfied in 64.4% of the cases). For $\bar{m}_1 = 1.4$ and $\bar{m}_2 = 10$, CRS and DIRECT do not outperform the GA (see Table 3.7). CRS achieves a high percentage of success, i.e. 98-99%, with criterion (3.3) satisfied in more than 91% of the cases; the number of objective function evaluations has a mean value ranging

$\bar{m}_1 = 5, \bar{m}_2 = 10$							
algorithm	*b*	*fmean*	*fstd*	*success (%)*	*stop (%)*	*evmean*	*evstd*
CRS	0	10.2268	1.0160	100	100	789	524
	4	10.2518	1.0208	100	100	940	614
	8	10.2915	1.0247	100	99.8	1331	893
PSwarm	0	9.7705	1.2621	92	34.1	694	834
	4	9.8434	1.1879	93.7	35	722	539
	8	10.0475	1.0658	97.3	40.7	794	1016
DIRECT	–	10.2900	1.0414	100	100	851	602

Table 3.6: Performance of CRS, PSwarm and DIRECT for $\bar{m}_1 = 5$, $\bar{m}_2 = 10$. The RAND-GRID strategy, with different values of the parameter b, is used by CRS and PSwarm.

from 3312 to 3926 (depending on the value of b in the RAND-GRID strategy), but its largest value varies between 15332 and 16359, resulting much greater than for the GA. DIRECT gets 100% of success, with 90% of runs satisfying (3.3), but the mean value of the number of objective function evaluations is 12438; furthermore the algorithm stops in 10% of the cases because the maximum number of objective function evaluations has been reached. As for $\bar{m}_1 = 5$ and $\bar{m}_2 = 10$, PSwarm is less effective than the GA, since it achieves a smaller percentage of success, i.e. at most 95.9%, with criterion (3.3) satisfied in about 20% of the cases. The performance of CRS, PSwarm and DIRECT strongly deteriorates for $\bar{m}_1 = \bar{m}_2 = 1.4$, as shown by the results reported in Table 3.8 (the *stop* column reports the percentage of runs where the algorithm stops by satisfying criterion (3.3)). 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 criterion (3.3) 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

$\bar{m}_1 = 1.4, \bar{m}_2 = 10$							
algorithm	*b*	*fmean*	*fstd*	*success (%)*	*stop (%)*	*evmean*	*evstd*
CRS	0	10.2604	1.0464	99.4	95.3	3312	2014
	4	10.2285	1.0991	98.3	94.3	3612	2334
	8	10.1921	1.1192	98.2	91.2	3926	2607
PSwarm	0	9.6536	1.0676	93.4	17.4	609	680
	4	9.6855	1.0542	95.9	20.1	741	990
	8	9.6342	1.0798	93.0	21.6	809	672
DIRECT	–	10.2483	1.0424	100	90	12438	7797

Table 3.7: Performance of CRS, PSwarm and DIRECT for $\bar{m}_1 = 1.4$, $\bar{m}_2 = 10$. The RAND-GRID strategy, with different values of the parameter b, is used by CRS and PSwarm.

$\bar{m}_1 = \bar{m}_2 = 1.4$							
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

Table 3.8: Performance of CRS, PSwarm and DIRECT for $\bar{m}_1 = \bar{m}_2 = 1.4$. The RAND-GRID strategy, with different values of the parameter b, is used by CRS and PSwarm.

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 those obtained with the GA if a number of objective function evaluations much larger than 30000 is allowed. Of course, in this case DIRECT is far from being competitive with the GA and the grid search.

Chapter 4

A Parallel Version of the Genetic Algorithm

4.1 Introduction

Parallel genetic algorithms have been extensively investigated since they are easy to implement and can achieve significant gains not only in terms of efficiency, but also of effectiveness with respect to sequential genetic algorithms [103, 104]. There several approach to parallelize a genetic algorithm. These depends on the following issues:

- how the fitness is evaluted;
- if the genetic operators are applied locally or globally with respect to feasible domain;
- if single or multiple subpopulations are used;
- how individuals are exchanged if multiple subpopulations are used.

The simplest approach for parallelizing a genetic algorithm is to execute multiple copies of the same genetic algorithm. Each copy starts with a different initial population and evolve independently. When all the copies halt the individual with best

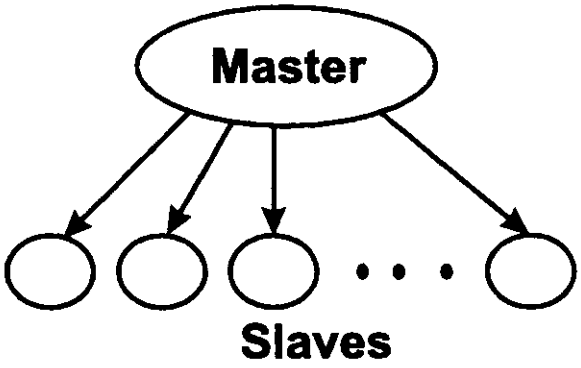

Figure 4.1: A schematic of a master-slave parallel genetic algorithms.

fitness among the copies is assumed as solution. The advantage of this approach is the reduction of the possibility that all copies converge prematurely to the same local solution. Another approach consists of considering several populations that evolve independently, but exchange some individuals among the populations. This allows to avoid a premature convergence and improve the accuracy of the solution through the share of high quality solutions. A different approach is to partition the feasible domain into disjoint subset and to execute a genetic algorithm in each subset. The previous approaches led four main classes of parallel genetic algorithm that are discussed in the following subsections.

4.1.1 Master-Slave Parallel Genetic Algorithms

Master-slave parallel genetic algorithms have a single population with a panmictic structure, i.e. they use global selection, recombination and replacement operators. This means that any individual may compete and mate with any other (see Figure 4.1). Usually, only the evaluation of the fitness is carried out in parallel: a master process stores the population, applies the genetic algorithm operators and distributes the individuals among the slaves, which compute the fitness values and return them to the master. This parallel approach does not change the behaviour of the sequential algorithm, unless an asynchronous master-slave model is used, where the next generation step can be started even if all the fitness values of the cur-

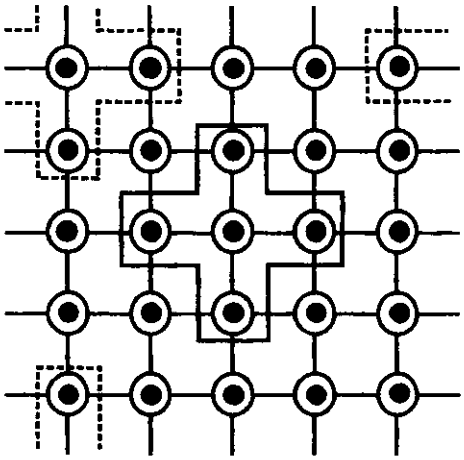

Figure 4.2: A schematic of a cellular parallel genetic algorithm.

rent population have not yet been received by the master. Master-slave parallel genetic algorithms are well suited for symmetric multiprocessors, but can be easily implemented on distributed-memory systems.

4.1.2 Cellular Parallel Genetic Algorithms

Cellular (or fine-grained) parallel genetic algorithms use a single spatially-structured population. The structure is often a toroidal grid, with each individual assigned to a grid cell. The cells are grouped into small neighbourhoods and the genetic operators are applied within each neighbourhood (see Figure 4.2). The evaluation of the fitness is performed concurrently for all the individuals. The overlap among neighborhoods allows some interaction among the individuals, so that the best one may be (slowly) diffused in the whole population. These parallele genetic algorithms are called cellular for their analogy to cellular automata with stochastic transition rules. They naturally fit massively parallel processor systems.

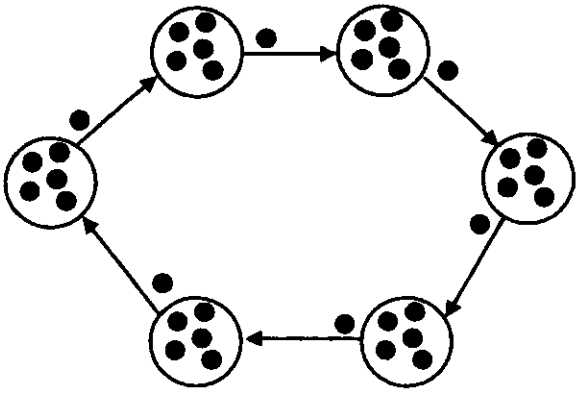

Figure 4.3: A schematic of a multiple-deme parallel genetic algorithm.

4.1.3 Multiple-Deme Parallel Genetic Algorithms

Multiple-deme parallel genetic algorithms use several subpopulations (demes) that evolve independently, but exchange individuals occasionally, thus resembling the island model in population genetics (see Figure). The genetic operators are separately applied to each deme; the exchange of individuals is performed through the so-called *migration* operator at certain steps of the evolution. Since the size of the demes is generally smaller than the population used by the corresponding sequential genetic algorithm, one expects that the parallel algorithm converges faster. On the other hand, the migration is applied reinject diversity into the demes, to avoid premature convergence to low-quality solutions. The migration operator depends on several parameters, that may significantly affect the behaviour of the parallel genetic algorithm and produce strong changes with respect to its sequential counterpart (more details on these parameters are given later). Therefore, an intensive research activity is devoted to understand the complex effects of this operator. Multiple-deme parallel genetic algorithm are usually implemented on MIMD distributed-memory systems.

4.1.4 Hierarchical Parallel Genetic Algorithm

Hierarchical parallel genetic algorithms combine different algorithms of the previous classes in a two-level structure, therefore they are also called hybrid. Most of these algorithms apply multiple-deme parallel genetic algorithms at the upper level and master-slave or cellular ones at the lower level, i.e. on each deme, but other combinations have been also considered. The hierachical approach adds a degree of complexity to the study of parallel genetic algorithms, but appears promising for the implementation on modern multi-core and many-core architectures.

4.2 A Multi-Deme Parallel Genetic Algorithm for the Detection Problem

We choose the multiple-deme approach because it allows a great flexibility in the design of the parallel genetic algorithm and hence a better adaptation to the problem [105]. Furthermore, we are interested in running the algorithm on distributed-memory systems. We note that, since the evaluation of the objective function in (1.1) is expensive, the cost for the migration of individuals is negligible. By using a SPMD programming model and assuming that each processor is associated to a deme, the structure of our parallel genetic algorithm can be described as shown in Figure 4.4. The choice of the initial deme and the genetic operators acting on it, except the migration, are the same as in the sequential genetic algorithm.

As previously observed, the choice of the migration strategy is fundamental for the behaviour of a multiple-deme parallel genetic algorithm. In the following we describe the main issues characterizing the migration [103, 104]. and the choices made in our algorithm.

- *Topology.* It defines the interconnections among the demes, i.e. the neighbours to/from which a deme can send/receive individuals. The topology affects the diversity of the demes; the higher the connectivity, or the shorter the diameter,

```
ON EACH PROCESSOR
initialize the deme
while (stopping criterion not satisfied)
    select the parents
    recombine pairs of parents to generate offspring
    replace some parents with some offspring
    mutate the resulting population
    if (this is a migration step)
       exchange some individuals with other demes,
       i.e. migrate
    end if
end while
```

Figure 4.4: Basic structure of the multiple-deme PGA (SPMD model).

the faster the best individuals are diffused among the demes, thus leading to more homogeneous subpopulations. To keep genetic diversity we choose a *ring* topology, in which the large diameter allows the generation of significantly different best individuals in the various demes. Specifically, each deme sends individuals to the right and receives them from the left.

- *Selection/Replacement of migrants.* This is the strategy used to select the individuals of the deme that must migrate and those that must be replaced by the immigrants. We apply a strategy consistent with the basic principles of our sequential GA: keeping genetic diversity while ensuring that the best individual is preserved. Therefore, the set of individuals that migrate consists of a copy of the best one plus a group of individuals selected by using a discrete uniform distribution. The individuals that must be replaced are also chosen randomly; the best individual is not replaced even if it is selected.

- *Migration rate.* This is the number of individuals that migrate, R_M. Generally,

the larger the migration rate, the lower the diversity among the demes. R_M can be set as a fixed value or as percentage of the size of the deme. We use the second approach in our algorithm.

- *Migration interval.* This is the number of generations between two consecutive migrations. Large migration intervals generally imply more diversity. The migration interval is usually set to a fixed constant value I_M (synchronous migration), as in our algorithm. However, it can be also dynamically set for each deme, e.g. using a given probability to decide if the migration will take place in the deme (asynchronous migration).

4.2.1 Numerical Experiments

The Parallel Genetic Algorithm (PGA) was implemented in the C language, in double precision, using MPI [106] for the inter-process data communication, and the Mersenne Twister algorithm [96] for generating pseudo-random numbers from a uniform distribution.

Different sets of test problems, representative of different configurations of a coalescing binary system, were generated using the LAL library [94], as explained in Chapter 3. The experiments reported here concern the most significant and difficult to solve among these test sets, which corresponds to a gravitational signal emitted by a binary system with masses $\bar{m}_1 = 1.4M_\odot$ and $\bar{m}_2 = 1.4M_\odot$ ($M_\odot$ denotes the solar mass); 30 instances of strictly white noise were added to this signal, obtaining 30 sequences to be analysed. The length of each sequence is $N = 131072$, while the length of the signal is $M = 51072$. The SNR was set to 10 and the lower and upper bounds on the masses, l and u, were set to 1 and 30, respectively. We recall that the SNR is the reference value for the mean of the computed maximum of F.

The total population of the PGA was divided into equal-sized demes and each deme was associated to a single processor.[1] The algorithm was stopped when each

[1]When N_P was not divisible by the number of processors, the remainder was uniformly distributed among the processors.

deme reached the maximum number of generations, N_G. As for the sequential GA, a threshold equal to 8 was used to evaluate the robustness of the algorithm, i.e. its ability to detect a signal. The probabilities of recombination and mutation and the maximum number of generations were chosen as in the sequential algorithm, i.e. $P_R = 0.7$, $P_M = 0.05$ and $N_G = 50$. The first experiments were performed with the same population size used in Chapter 3, i.e. $N_P = 100$; further experiments were executed with $N_P = 200$ to analyse the (possible) gain in the accuracy of the solution versus the computational cost. By numerical experiments we found that the PGA, in our problem, is not significantly affected by the value of the migration rate M_R and the migration interval M_I (note that M_R is expressed as a fraction of N_P). For each instance of the test problem the PGA was run 30 times, varying the seed for the initialization of the pseudo-random number generator.

For comparison purpose, we applied also a parallel version of the grid-search algorithm, obtained by dividing the grid into subgrids and by assigning a subgrid to each processor. Since the grid was generated by the LAL library as a sequence of points, each subgrid was formed by uniformly dividing the sequence among the processors.

The test were run on a Linux cluster with 8 dual-core nodes, available at the Department of Mathematics of the Second University of Naples. Each node comprises an Intel Core 2 DUO E7300 processor, with clock frequency of 2.66 GHz, 4 GB of RAM, and 3 MB of cache memory; it runs Linux Ubuntu 8.10, 64 bit version, with kernel 2.6.27. The nodes are connected by a Fast Ethernet network. The PGA code was compiled by using gcc v. 4.3.2, using the implementation of MPI provided by mpich v. 1.2.7, the Mersenne Twister available in the GNU Scientific Library v. 1.1, and the LAL library v. 5.2.

In Tables 4.1 and 4.2 we report the results obtained by the PGA varying the number of processors (*procs*), with $N_P = 100$ and $N_P = 200$, respectively; for $N_P = 100$ we did not consider more than 8 processors to avoid too small demes. The *tmean* and *tstd* columns contain the mean execution times, in seconds, over

900 runs (30 realizations of noise $\times$ 30 seeds for the initialization of the pseudo-random number generator) and the corresponding standard deviations, respectively; the *speedup* column contains the speedup values related to the mean execution times; finally, $fmean$ and *succ* show the mean values of the computed maximum of the objective function F and the percentage of successes, i.e. of runs in which the maximum of F exceeded the detection threshold. We recall that this percentage cannot be greater than 96.7, since for one of the 30 instances of the problem the maximum value of F is below the threshold (the maximum computed with the grid search is 7.02). In Table 4.3 we show the execution times (sec.) and the speedup values obtained with the parallel grid search on the same grid used by the PGA in the generation of the initial demes (27379 points). Note that, for each number of processors, all the 30 instances of the problem have about the same execution time, since the cost of a single function evaluation basically depends on the grid point (m_1, m_2), and the same grid is used for all the experiments.

We see that, for both the population sizes, the PGA requires an execution time that is much smaller than the time for the grid search on the same number of processors. Specifically, for $N_P = 100$ the PGA time is less than 1/3 of the grid-search time, while for $N_P = 200$ it is less than 2/3 (about 2/5 on 16 processors). The speedup of the PGA is always satisfactory, while the speedup of the grid-search significantly deteriorates on 16 processors. This is due to the fact that the time for a function evaluation is highly variable (from 0.03 sec. to 0.3 sec.) and hence a uniform distribution of the grid points among the processors may lead to load imbalance. The variability of the time for a function evaluation produces also some load imbalance in the PGA. By looking at the values of the computed maxima, we see that the use of multiple demes along with the migration strategy generally enhances the mean value of the maximum of F with respect to the sequential GA, even when the size of the demes is very small (12-13 individuals). It also improves the success of the algorithm for $N_P = 200$, while slightly reduces it on 4 and 8 processors for $N_P = 100$. We note that the PGA with $N_P = 200$ can compute a

M_R	M_I	*procs*	*tmean*	*tstd*	*speedup*	*fmean*	*succ* (%)
—	—	1	403.63	62.10	—	9.8617	93.7
0.1	5	2	223.92	27.52	1.80	9.9181	93.9
		4	119.59	15.24	3.38	9.8680	91.2
		8	60.68	6.28	6.65	9.9129	92.4
0.1	10	2	222.41	24.42	1.81	9.9584	94.3
		4	118.06	12.40	3.42	9.9414	92.7
		8	60.01	5.92	6.73	9.9165	92.7
0.3	5	2	223.80	28.62	1.80	9.9161	93.4
		4	119.77	14.95	3.37	9.8684	91.7
		8	61.55	6.37	6.56	9.9383	93.2
0.3	10	2	219.49	28.26	1.84	9.9130	93.7
		4	117.89	13.07	3.42	9.9349	92.8
		8	60.46	6.05	6.68	9.9574	92.8

Table 4.1: Performance of the PGA with $N_P = 100$.

M_R	M_I	*procs*	*tmean*	*tstd*	*speedup*	$fmean$	*succ* (%)
—	—	1	774.84	116.89	—	9.9719	95.6
0.1	5	2	436.58	45.42	1.77	10.05	96.4
		4	236.36	15.71	3.28	10.09	96.6
		8	125.84	8.25	6.16	10.10	95.7
		16	62.86	4.01	12.33	10.11	96.3
0.1	10	2	433.58	45.27	1.79	10.01	96.0
		4	233.69	15.08	3.32	10.09	96.7
		8	123.81	7.02	6.26	10.12	96.4
		16	62.25	4.75	12.45	10.11	96.2
0.3	5	2	431.10	48.27	1.80	10.04	96.3
		4	235.26	17.60	3.29	10.09	96.7
		8	127.58	7.58	6.07	10.11	96.4
		16	63.84	4.46	12.14	10.10	96.0
0.3	10	2	428.90	49.42	6.57	10.03	96.3
		4	232.42	16.35	3.33	10.09	96.4
		8	124.74	7.29	6.21	10.13	96.4
		16	62.77	4.93	12.34	10.11	96.4

Table 4.2: Performance of the PGA with $N_P = 200$.

$fmean = 10.0424,\ succ = 96.7\%$		
procs	*time*	*speedup*
1	1382.46	—
2	692.64	1.99
4	426.65	3.24
8	226.03	6.11
16	156.15	8.85

Table 4.3: Performance of the parallel grid search.

maximum value of F that is greater than the one obtained by the grid search, with a percentage of successes which is very close to the maximum one.

Conclusions

The aim of the research activity described in this thesis was the design and the development of numerical methods to solve a global optimization problem in the context of gravitational wave detection.

We developed a real-coded genetic algorithm tailored to the optimization problem under consideration. Our algorithm is able to compute solutions that are comparable, in terms of accuracy, to those obtained by the grid search, which is widely used for solving the optimization problem. On the other hand, we found that our algorithm allows a strong reduction of the computation cost with respect to the grid search, thus providing a more powerful tool in the analysis of the noisy data of detectors. The genetic algorithm resulted also much more efficient than other well-established global optimization algorithms (controlled random search, particle swarm pattern search and DIRECT) on the most significant and difficult set of problem instances. The key issue in designing our algorithm was the choice of the initial population by taking into account characteristic features of the problem. This idea, coupled with suitable genetic operators and a careful handling of the contraints, led to a quite efficient and robust algorithm for our problem.

In order to reduce the execution time of our genetic algorithm and to improve its effectiveness, we developed a parallel version of it for MIMD distributed memory systems. The parallel algorithm is based on the *multiple-deme* approach, in which many subpopulations (demes) evolve separately, but exchange individuals through a migration operator. We selected a suitable migration strategy according to the characteristics of the problem. Computational experiments on the most significant

and difficult instances of the problem showed that the parallel algorithm allows to increase the accuracy and the reliability of the sequential genetic algorithm. Furthermore the results are comparable, in terms of accuracy, with a parallel version of the grid search, but are obtained with a lower computation time.

Bibliography

[1] L.C.W. Dixon and G.P. Szego. *Towards global optimisation*. North-Holland Publishing Company., 1975.

[2] L.C.W. (ed.) Dixon and G.P. (ed.) Szego. *Towards global optimisation 2*. North-Holland Publishing Company., 1978.

[3] K. Hans, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer Verlag, 2005.

[4] G. Szpiro. *Kepler's conjecture*. John Wiley & Sons, 2003.

[5] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The traveling Salesman Problem*. Princeton University Press, 2006.

[6] T. S. Motzkin and E. G. Strauss. Maxima for graphs and a new proof of a theorem of turan. *Canadian Journal of Mathematics*, 17(533-540), 1965.

[7] A. Neumaier. Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Review*, 39(407-460), 1997.

[8] C. M. McDonald and C. A. Floudas. Global optimization for the phase and chemical equilibrium problem: application to the nrtl equation. *Computers & Chemical Engineering*, 19:1111–1139, 1995.

[9] K. Anstreicher. Recent advances in the solution of quadratic assignment problems. *Mathematical Programming B*, 2003:27–42, 97.

[10] J.-B. Hiriart-Urruty. When is a point x satisfying $\nabla f(x) = 0$ a global minimum of f? *Am. Math. Mon.*, 93:556–558, 1986.

[11] Z. Quan. Optimality conditions for global optimization (i). *Acta Mathematicae Applicatae Sinica*, 2(1):66–78, 1985.

[12] Z. Quan. Optimality conditions for global optimization (ii). *Acta Mathematicae Applicatae Sinica*, 2(2):118–132, 1985.

[13] J.-B. Hiriart-Urruty. Conditions for global optimality. In *Handbook of global optimization*, volume 2 of *Nonconvex Optim. Appl.*, pages 1–26. Kluwer Acad. Publ., Dordrecht, 1995.

[14] A. M. Rubinov and Z. Y. Wu. Optimality conditions in global optimization and their applications. *Math. Program.*, 120(1, Ser. B):101–123, 2009.

[15] Z. Y. Wu. Sufficient global optimality conditions for weakly convex minimization problems. *J. Global Optim.*, 39(3):427–440, 2007.

[16] Aimo Törn and Antanas Zilinskas. *Global optimization.* Lecture Notes in Computer Science. 350. Berlin etc.: Springer-Verlag. X, 255 p. , 1989.

[17] C. C. McGeoch. Experimental analysis of algorithms. In *Handbook of global optimization, Volume 2*. Kluwer Academic Publishers, 2002.

[18] Charoenchai Khompatraporn, János D. Pintér, and Zelda B. Zabinsky. Comparative assessment of algorithms and software for global optimization. *J. Glob. Optim.*, 31(4):613–633, 2005.

[19] A. H. G. Rinooy Kan and G. T. Timmer. Global optimization. In G. L. Nemhauser, M. J. Todd, and A. H. G. Rinooy Kan, editors, *Handbook in Operations Research and Management Science*, volume 1. North-Holland Publishing Company., 1989.

[20] R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to global optimization*. Kluwer Academic Publishers, 1995.

[21] F. Archetti and F. Schoen. A survey on the global optimization problem: General theory and computational approaches. *Annals of Operations Research*, 1:87–110, 1984.

[22] R. Horst and P. M. Pardalos, editors. *Handbook of global optimization*. Kluwer Academic Publishers, 1995.

[23] P. M. Pardalos and H. E. Romeijn, editors. *Handbook of global optimization*, volume 2. Kluwer Academic Publishers, 2002.

[24] L.C.W. Dixon. The global optimisation problem: An introduction. In *Towards global optimisation 2*. North-Holland Publishing Company., 1978.

[25] C.P. Stephens and W. Baritompa. Global optimization requires global information. *J. Optimization Theory Appl.*, 96(3):575–588, 1998.

[26] Arnold Neumaier. Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica*, 13:271–369, 2004.

[27] W. Forster. Homotopy methods. In R. Horst and P. M. Pardalos, editors, *Handook of Global Optimization*. Kluwer Academic Publishers, 1995.

[28] M. Locatelli. Simulated annealing algorithms for continuous global optimization. In P. M. Pardalos and H. E. Romeijn, editors, *Handbook of global optimization*, volume 2. Kluwer Academic Publishers, 2002.

[29] P. Salamon, P. Sibani, and R. Frost. *Facts, Conjectures, and Improvements for Simulated Annealing*. SIAM, 200.

[30] D. E. Goldberg. *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[31] M. Melanie. *An Introduction to Genetic Algorithms*. The MIT Press, 1999.

[32] C. Blum and D. Merkle, editors. *Swarm Intelligence*. Springer, 2008.

[33] C. G. E. Boender and H. E. Romeijn. Stochastic methods. In *Handbook of global optimization*. Kluwer Academic Publishers, 1995.

[34] R. Marti. Multi-start methods. In *Handbook of Metaheuristics*. Springer New York, 2003.

[35] G. R. Wood and Z. B. Zabinsky. Stochastic adaptive search. In P. M. Pardalos and H. E. Romeijn, editors, *Handbook of global optimization*, volume 2. Kluwer Academic Publishers, 2002.

[36] F. Archetti and B. Betro. A priori analysis of deterministic strategies for global optimization problems. In *Towards global optimisation 2.* North-Holland Publishing Company., 1978.

[37] B. O. Shubert. A sequential method seeking the global maximum of a function. *SIAM Journal on Numerical Analysis*, 9(3):379–388, 1972.

[38] R. H. Mladineo. An algorithm for finding the global maximum of a multimodal multivariate function. *Mathematical Programming*, 34:188–200, 1986.

[39] P. Hansen and B. Jaumard. Lipschitz optimization. In R. Horst and P. M. Pardalos, editors, *Handbook of global optimization*. Kluwer Academic Publishers, 1995.

[40] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

[41] D. R. Jones. Direct global optimization. In C. A. Floudas, editor, *Encyclopedia of optimization*. Springer, 2009.

[42] Waltraud Huyer and Arnold Neumaier. Global optimization by multilevel coordinate search. *J. Global Optim.*, 14(4):331–355, 1999.

[43] János D. Pintér. *Global optimization in action. Continuous and Lipschitz optimization: algorithms, implementations and applications.* Kluwer Academic Publishers, 1996.

[44] R. B. Kearfott. *Rigorous global search: continuous problems.* Kluwer Academic Publishers, 1996.

[45] R. B. Kearfott. Interval analysis: unconstrained and constrained optimization. In C. A. Floudas, editor, *Encyclopedia of optimization.* Springer, 2009.

[46] R. A. Hulse and J. H. Taylor. Discovery of a pulsar in a binary system. *Astrophysical Journal*, 195:L51, 1975.

[47] K. S. Thorne. Gravitational radiation. In S. W. Hawking and W. Israel, editors, *300 Years of Gravitation*, pages 330–458. Cambridge University Press, 1987.

[48] L. Milano, F. Barone, and M. Milano. Time domain amplitude and frequency detection of gravitational waves from coalescing binaries. *Phys. Rev. D*, 55(8):4537–4554, 1997.

[49] F. J. Raab. The ligo project: Progress and prospects. In E. Coccia, G. Pizzella, and F. Ronga, editors, *Gravitational Wave Experiments.* World Scientific, 1994.

[50] A. Giazotto and et al. The virgo experiment: status of the art. In E. Coccia, G. Pizzella, and F. Ronga, editors, *Gravitational Wave Experiments.* World Scientific, 1994.

[51] B. Willke and et al. The geo 600 gravitational wave detector. *Classical and Quantum Gravity*, 19:1377–1387, 2002.

[52] K. Tsubono. 300-m laser interferometer gravitational wave detector (tama300) in japan. In E. Coccia, G. Pizzella, and F. Ronga, editors, *Gravitational Wave Experiments.* World Scientific, 1994.

[53] D. E. McClelland and H. A. Bachor, editors. *Gravitational Astronomy: instrument design and astrophysical prospects*. World Scientific, 1990.

[54] D. G. Blair, editor. *The detection of gravitational waves*. Cambridge University Press, 1991.

[55] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 3rd edition, 1991.

[56] L. Blanchet. Post-newtonian gravitational radiation. In *Einstein's Field Equations and Their Physical Implications: Selected Essays in Honour of Jurgen Ehlers*. Springer, 2000.

[57] S. Babak, R. Balasubramanian, D. Churches, T. Cokelaer, and B. S. Sathyaprakash. A template bank to search for gravitational waves from inspiralling compact binaries I: physical models. *Classical and Quantum Gravity*, 23:5477–5504, 2006.

[58] L. Blanchet, B. Rlyer, and A. G. Wiseman. Gravitational waveforms from inspiralling compact binaries to second-post-Newtonian order. *Class. Quantum Grav.*, 13:575–584, 1996.

[59] S. D. Mohanty. 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.

[60] T. Damour, B. R. Iyer, and B. S. Sathyaprakash. Comparison of search templates for gravitational waves from binary inspiral. *Physical Review D*, 63(4):044023, 2001.

[61] H. V. Poor. *An introduction to signal detection and estimation*. Springer, 2 edition, 1994.

[62] B. C. Levy. *Principles of signal detection and parameter estimation*. Springer, 2008.

[63] A. V. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-time signal processing*. Prentice-Hall, 2 edition, 1999.

[64] B. J. Owen. Search templates for gravitational waves from inspiraling binaries: choice of template spacing. *Phys. Rev. D*, 53(12):6749–6761, 1996.

[65] A. S. Sengupta, S. Dhurandhar, and A. Lazzarini. Faster implementation of the hierarchical search algorithm for detection of gravitational waves from inspiraling compact binaries. *Phys. Rev. D*, 67(8):082004, 2003.

[66] K. A. De Jong. *Evolutionary Computation: a unified approach*. MIT press, 2006.

[67] I. Rechenberg. Cybernetic solution path of an experimental problem. In *Library Translation 1122*. Farnborough: Royal Aircraft Establishment, 1965.

[68] L. Fogel, A. Owens, and M. Walsh. *Artificial intelligence through simulated evolution*. John Wiley & Sons, New York, 1966.

[69] J. Holland. Outline for a logical theory of adaptive systems. *Journal of the Association for Computing Machinery*, 9:297–314, 1962.

[70] J. Holland. Nonlinear environments permitting efficient adaptation. In *Computer and Information Sciences II*. Academic Press, 1967.

[71] A. E. Eiben and J. E. Smith. *Introduction to evolutionary computing*. Springer, 2003.

[72] James E. Smith. Genetic algorithms. In M. P. Panos and H. E. Romeijn, editors, *Handbook of global optimization, Volume 2*. Kluwer Academic Publishers, 2002.

[73] C. R. Reeves and J. E. Rowe. *Genetic algorithms: principles and perspectives*. Kluwer Academic Publishers, 2003.

[74] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, 3 edition, 1998.

[75] S. N. Sivanandam and S. N. Deepa. *Introduction to genetic algorithms*. Springer, 2008.

[76] T. Back. Binary strings. In *Evolutionary computation 1. Basic algorithms and operators*. Institute of Physics Publishing, 2000.

[77] J. H. Holland. *Adaptation in natural and artificial systems*. The MIT Press, 1992.

[78] M. D. Vose. *The simple genetic algorithm*. The MIT Press, 1999.

[79] A. E. Nix and M. D. Vose. Modeling genetic algorithms with markov chains. *Annals of Mathematics and Artificial Intelligence*, 5(1):79–88, 1992.

[80] A. Prugel-Bennet and J. L. Shapiro. Analysis of genetic algorithms using statistical mechanics. *Physical Review Letters*, 72(9):1305–1309, 1994.

[81] D. B. Fogel. Real-valued vectors. In *Evolutionary computation 1. Basic algorithms and operators*. Institute of Physics Publishing, 2000.

[82] H. Maaranen, K. Miettinen, and A. Penttinen. On initial populations of a genetic algorithm for continuous optimization problems. *Journal of Global Optimization*, 37(3):405–436, 2007.

[83] K. Deb. Introduction to selection. In *Evolutionary computation 1. Basic algorithms and operators*. Institute of Physics Publishing, 2000.

[84] F. Herrera, M. Lozano, and A. M. Sánchez. A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study. *International Journal of Intelligent Systems*, 18(3):309–338, 2003.

[85] L. Davis. *Handbook of genetic algorithms*. Van Nostrand Reinhold, 1991.

[86] F. Herrera, M. Lozano, and J. L. Verdegay. Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. *Artificial Intelligence Review*, 12(4):265–319, 1998.

[87] N. J. Radcliffe. Forma analysis and random respectful recombination. In *Proceedings of the 4th International Conference on Genetic Algorithms*, 1991.

[88] D. E. Goldberg. Real-coded genetic algorithms, virtual alphabets, and blocking. *Complex Systems*, 5:139–167, 1991.

[89] F. Palmieri and Q. Xiaofeng. Theoretical analysis of evolutionary algorithms with a infinite population size in continuous space. part i: basic properties of selection and mutation. *IEEE Transactions on Neural Networks*, 5(1):102–119, 1994.

[90] F. Palmieri and Q. Xiaofeng. Theoretical analysis of evolutionary algorithms with a infinite population size in continuous space. part ii: analysis of the diversification role of crossover. *IEEE Transactions on Neural Networks*, 5(1):120–129, 1994.

[91] Z. Michalewicz and C. Z. Janikow. Genetic algorithms for numerical optimization. *Statistics and Computing*, 1:75–91, 1991.

[92] P. J. B. Hancock. A comparison of selection mechanisms. In *Evolutionary computation 1. Basic algorithms and operators*. Institute of Physics Publishing, 2000.

[93] T. Blickle. Tournament selection. In *Evolutionary computation 1. Basic algorithms and operators*. Institute of Physics Publishing, 2000.

[94] B. Allen and et al. *LAL Software Documentation*. Revision 1.44, 2005.

[95] A. R. Rasio and S. L. Shapiro. Coalescing binary neutron stars. *Classical and Quantum Gravity*, 16(6):R1–R29, 1999.

[96] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, 1998.

[97] S. Mitra, S. V. Dhurandhar, and L. S. Finn. Improving the efficiency of the detection of gravitational wave signals from inspiraling compact binaries: Chebyshev interpolation. *Physical Review D*, 72:102001, 2005.

[98] T. Back. Mutation parameters. In T. Back, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary computation 2: advanced algorithms and operators*, pages 142–151. IOP Publishing, Bristol, 2000.

[99] P. Vajda, A.E. Eiben, and W. Hordijk. Parameter control methods for selection operators in genetic algorithms. In *Parallel problem solving from nature - PPSN X*, Lecture Notes in Computer Science, pages 620–630. Springer, Berlin/Heidelberg, 2008.

[100] W.L. Price. A controlled random search procedure for global optimisation. *Computer Journal*, 20:367–370, 1977.

[101] A.Ismael F. Vaz and Luís N. Vicente. A particle swarm pattern search method for bound constrained global optimization. *Journal of Global Optimization*, 39(2):197–219, 2007.

[102] G. Liuzzi, S. Lucidi, and V. Piccialli. A direct-based approach exploiting local minimizations for the solution of large-scale global optimization problems. *Computational Optimization and Applications*, 2008.

[103] E. Alba. *Parallel metaheuristics. A new class of algorithms.* Wiley Interscience, 2005.

[104] E. Cantú-Paz. *Efficient and Accurate Parallel Genetic Algorithms.* Kluwer Academic Publishers., 2000.

[105] D. di Serafino and F. Riccio. On the application of multiple-deme parallel genetic algorithms in astrophysics. *Proceedings of the 18th Euromicro International Conference on Parallel Distributed and Network-Based Computing, Conference Publishing Services.* to appear.

[106] Marc Snir, Steve Otto, Steven Huss-Lederman, David W. Walker, and Jack J. Dongarra. *MPI: The Complete Reference. Vol. 1 – The MPI Core.* Scientific and Engineering Computation. The MIT Press, Cambridge, MA, second edition, 1998.