DISCLAIMER: This chapter is an extract of the HDR Manuscript of B. Addis "A journey through optimization: from global to discrete optimizatio and back".

Therefore, the references where B. Addis is co-author are separated from the others and coding letters are used: J for journals, C for conferences with review, B for book chapters and finally O for all the others. Only contributions that are referred in the text are reported.

Contents

1 Some contributions on Global optimization

Global optimization, even if largely applied also outside the optimization community, or maybe for this very reason, is a branch of optimization quite new and less known even by optimization experts and practitioners. Therefore, I decided to add a short section (Section 1.1) to introduce some general concepts and the notation I need to present my contributions in a more organic way.

The expert reader can skip this part, and must be advised that if she/he decides differently, she/he will find inaccuracies and missing parts due to the effort of make this part accessible and short. In fact, this section is not meant to be a survey on global optimization, neither a short version of it (the interested reader can refer to [32] for a quite comprehensive overview), but it has been thought as a "story-telling", in the sense that I describe some few concepts and methods (that are the building blocks used in my research) and explain the reasoning behind them. As a consequence, my description is biased by my own research experience.

1.1 A very short tour on global optimization

We will focus on continuous optimization:

$$P \begin{cases} \min f(x) \\ x \in S \subset \mathbb{R}^n \end{cases}$$

where $f : S \subset \mathbb{R}^n \Leftarrow \mathbb{R}$ is a sufficiently smooth function and $S \subset \mathbb{R}^n$ is a compact set. Under these hypotheses the **existence** of the minimum value $f^\star = f(x^\star)$ is guaranteed, where

$$x^\star \in S : f(x^\star) \leq f(x) \quad \forall x \in S$$

represents the/a corresponding global optimum point[1].

The problem that remains open is to determine **where** the global optimum is located and/or **which value** it assumes. For many years, global optimization as a discipline was put aside, because it was considered too difficult and, even worst, ill-defined. In fact, in 1978, Dixon proved that problem P is inherently unsolvable in a finite number of steps.

[1] multiple points can correspond to the same objective value, in this case we talk about equivalent solutions

Just to give the proof's flavor (that can be found in Dixon's paper [13]), let us consider a box-constrained optimization problem:

$$P \begin{cases} \min f(x) \\ x \in [a,b]^n \end{cases}$$

Suppose that we have sampled a given number of points in the feasible region, and evaluated the objective function f on all of them (see Figure 1.1a). Using this information, we can infer a "given shape" for f and, therefore, the position of the global optimum point (indicated by a full dot in Figure 1.1b). But, the collection of a new sample (as the one shown in Figure 1.1c) can prove that our guess was completely wrong. A new "guess" on the shape of f can be made as shown by the smooth line in Figure 1.1d. Then, a new sample can be found such that the procedure can be repeated again and again.

A little more formally, given a finite number of sample points, no matter how many, an infinite number of functions can interpolate such data, even evaluating in their correspondence not only the objective function f, but also its n-order derivatives (f, f', f'', ... f^n). Therefore, it is impossible to determine an unique "shape" for the objective function, and, as a consequence, a candidate point to be the global optimum.

Some additional information is needed to have a well-define global optimization problem. For example, if we can assume that problem P is a convex problem, i.e. f and S are convex, than any local optimum corresponds also to a global optimum, and the problem becomes solvable, at least numerically (see [4],[38] for comprehensive guides on convex optimization). In a very simplified way, such global information (convexity) allows to determine bounds on the behaviour of f "among two samples"[2], and therefore to concentrate the search in the region where "forcefully" the optimum is located. To come back to our example, if f is convex, then the new sample added in Figure 1.1c cannot exist, and the global optimum can be searched in the interval $[5, 7]$.

Convexity is a very strong assumption and it does not hold for many interesting optimization problems, nevertheless, there exists other forms of global information that allow to determine an (exact) solution method. Just to give some examples, we mention: quadratic programming[3], Lipschitz-continuous functions on a simple domain (box, simplex), low order polynomial functions, difference of convex problems, etc.

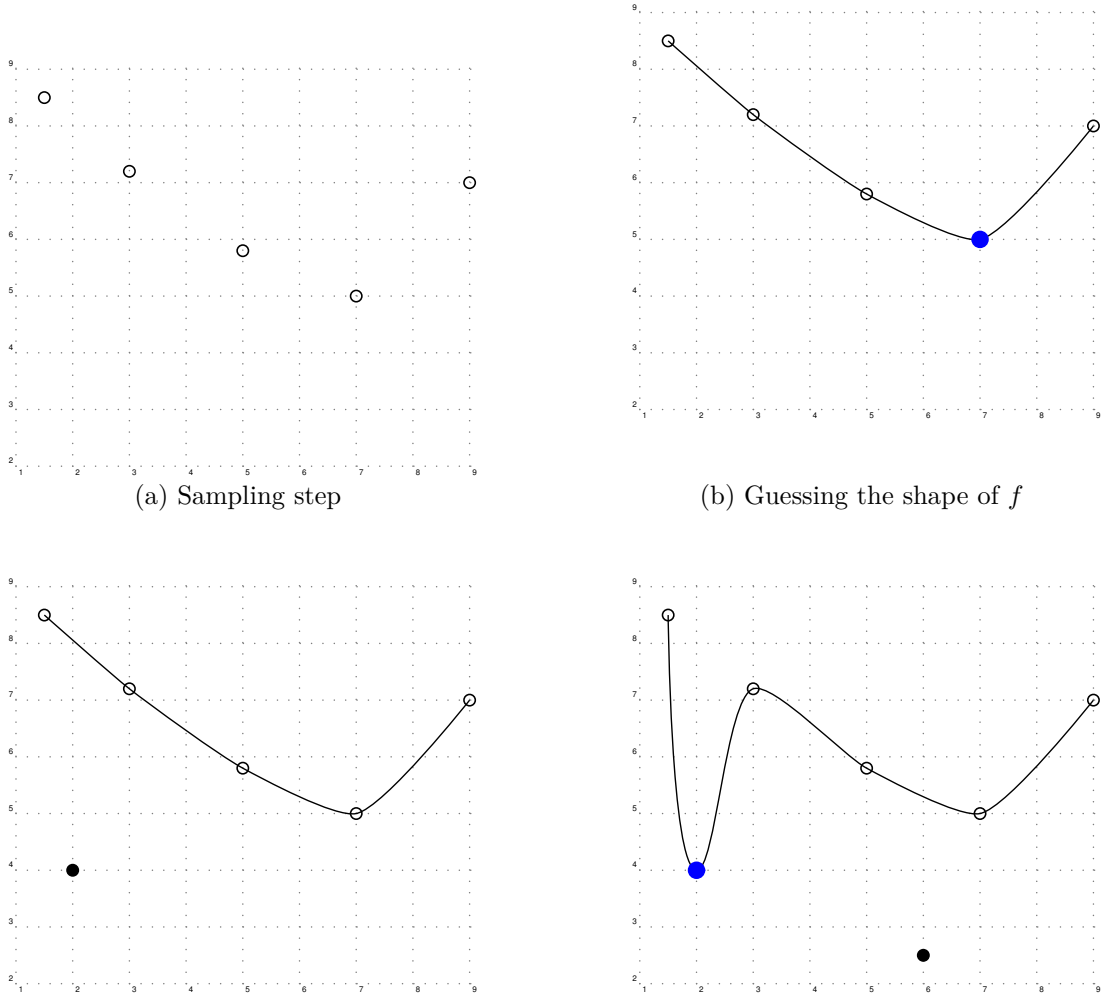
Even if weaker than convexity, these assumptions are not always valid, therefore the question remains open whether global optimization methods can be developed and applied to problems that do not present any information of these types.

Before trying to answer this question, two other questions must be answered: what does it mean exactly solving a non-linear (global) optimization problem and which kind of problems we are interested in.

To answer the first one, let us focus on numerical solutions and consider to accept solutions under a certain precision ϵ. In this case, we can consider that solving problem

[2] f is convex iff given two points x_1, x_2 any convex combination of them $y = \alpha x_1 + (1-\alpha)x_2$ with $\alpha \in [0,1]$ is such that $f(y) \leq \alpha f(x_1) + (1-\alpha)f(x_2)$

[3] Any concave quadratic function on a polyhedron assumes its minimum on a vertex.



(a) Sampling step

(b) Guessing the shape of f

(c) Sampling a new point

(d) A new guess for the shape of f, and ... a new sample

Figure 1.1: Looking for the global optimum

P means finding a point belonging to set:

$$S_\epsilon(f^\star) = \{x \in S : f(x) \leq f^\star + \epsilon\} \tag{1.1}$$

(see for example Figure 1.2) or, if we are more concerned about the distance from the "exact" global optimum, than about its value, in set:

$$B_\epsilon(x^\star) = \{x \in S : d(x^\star, S) \leq \epsilon\} \tag{1.2}$$

where $d(x, A)$ is a suitable distance between a point x and a set A. We need to observe

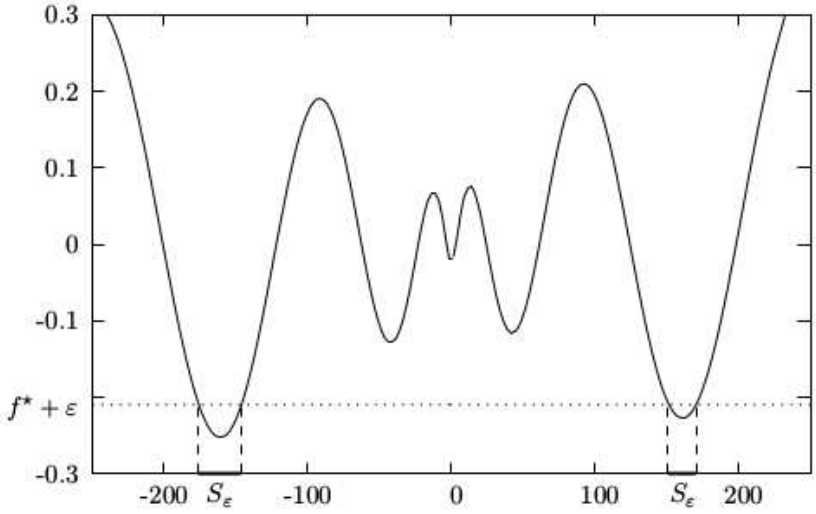
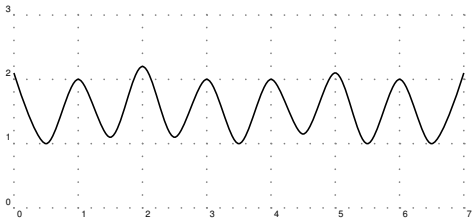


Figure 1.2: Numerical global optimum points

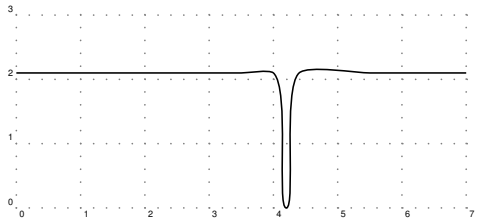
that, even finding a point in S_ϵ or B_ϵ is still a difficult problem (and ill-defined without any additional information).

To answer the second question, let us consider the three examples reported in Figure 1.3. The first two cases can be considered difficult to solve, but not interesting in practice. If all local optima are almost the same in value (as shown in Figure 1.3a), they can be numerically equivalent under the accepted numerical precision (or the precision needed by the application), therefore finding just a single local optimum can be enough. An almost flat objective with a single "hole" (as shown in Figure 1.3b) can produce an almost unsolvable problem (at least in probability). But, even if the global optimum is found, it would be "not useful" in practice, in fact a tiny change on the coordinates of such global optimum point could lead a significantly different objective value, and therefore with finite numerical precision it would be very difficult to use.

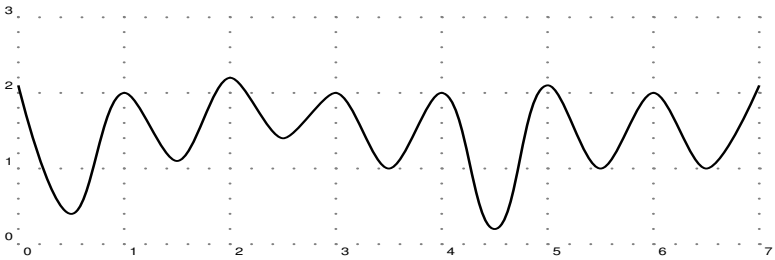
We are then left with the family of problems where one or some of the local optima are "significantly" different from the others in terms of value and/or position. For these problems, methods based on random sampling can guarantee a convergence in probability. If any additional information (global or local) is available, then basic strategies can be enriched to obtain better performances. In the following, I introduce some very basic meta-heuristics and I try to explain by examples what I mean with the terms local and global information and how this information can be used to modify such algorithms.



(a) Many "almost" equivalent optima



(b) Golf hole on an otherwise flat region



(c) Some local optima are "significantly" different from the others

Figure 1.3: Examples of global optimization problems

1.1.1 Some very simple meta-heuristic methods

The simplest *stochastic*[4] algorithm is known as Pure Random Search (PRS). At each iteration a random sample is selected[5] in the feasible region and it is illustrated in Algorithm1. Function GloballyGenerate() usually correspond to uniform generation in S. A classical stopping rule is given by a maximum number of iteration (N), whereas different rules, like the ones based on "apparent" convergence of the algorithm (no more improvement points found after a given number of iterations), can also be used.

Algorithm 1 `Pure Random Search`

procedure PURE RANDOM SEARCH(N)
  $x^\star$ =GloballyGenerate()
  while $n < N$ **do**
    z=GloballyGenerate()
    if $f(z) < f(x^\star)$ **then**
      $x^\star = z$
  return $x^\star, f(x^*)$

[4]Traditionally global optimization methods are divided in *deterministic* and *stochastic*, to the second group belong all methods that use some form of randomization in their decision process.

[5]Depending on the shape of feasible region, even uniformly random sampling can be a difficult task. Some possible ideas of how to perform this tasks are reported in the following for some applications.

The probability of finding the optimum is given by $\frac{V_{S_\epsilon}}{V_S}$, where V_S is the volume of the set S and V_{S_ϵ} is the volum of set S_ϵ (see Figure 1.2).

Using additional information on the problem to modify this simple algorithm can allow to improve the convergence probability. Such information can be divided in two main large classes: global information and local information. Let us consider two examples.

Local information Any global minimum is also a local minimum, therefore we can consider to use a standard local optimizer (gradient-descent, conjugate gradient, Quasi-Newton, etc) in conjunction with the PRS, getting the so called Multistart method (see Algorithm 2).

Algorithm 2 `Multistart`

```
1: procedure MULTISTART(N)
2:     x =GloballyGenerate()
3:     x* = locmin(x)
4:     while n < N do
5:         z=GloballyGenerate()
6:         z = locmin(z)
7:         if f(z) < f(x*) then
8:             x* = z
       return x*, f(x*)
```

The probability of finding the optimum using Multistart is $\frac{A_\epsilon}{V_S}$, where A_ϵ is the volume of the "basin of attraction" of points in $S_\epsilon^\star$ (see Figure 1.4). It is not obvious how to define a basin of attraction; the "common sense" is that a point x is in the basin of attraction of a local minimum point $\bar{x}$, if we can move from x to $\bar{x}$ "going down" along the objective function, (see [O1] for a tentative formal definition and some observations). We can observe that passing from PRS to Multistart is done introducing an *intensification* step in the search (made through a standard local search), i.e. a step that, instead of global exploring the overall space, tries to improve "locally" the current solution.

If we consider an ideal local optimizer, we can define function

$$\mathcal{L}_f(x) = \begin{cases} locmin_y(x) & f(y) \\ y \in S \end{cases}$$

where function $locmin(x)$ represent the solution obtained by a local minimizer using x as starting point. We can observe that minimizing $\mathcal{L}_f$ is equivalent to minimize f, therefore the Multistart algorithm can be imagined as a PRS applied to $\mathcal{L}_f$ instead to f. Many stochastic global optimization algorithms can be modified accordingly, considering to work on the ideal function $\mathcal{L}_f$ instead of f, simply adding a standard local search step after each sampling. We observe that, from a practical point of view, $\mathcal{L}_f$ is implicitly defined and **in practice** depends on the

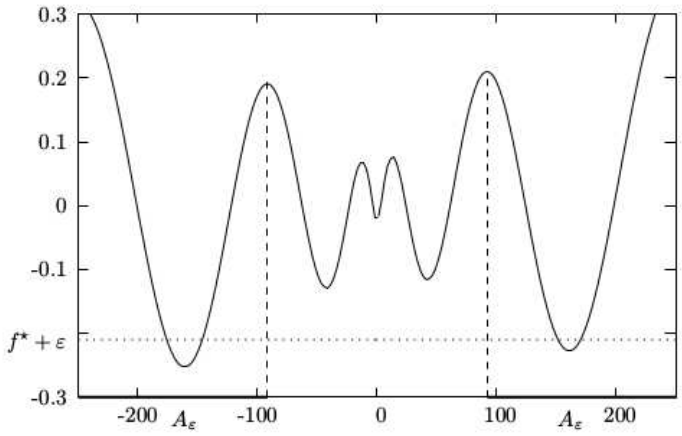


Figure 1.4: Multistart success probability

local minimizer used. In particular, if f is not locally convex, the local minimum can belong to a different basin of attraction with respect to the starting point x, or even points that are not local minima can be found, for example, any KKT point ([38]).

Global information In many global optimization problem, the local optima are not randomly displaced in the feasible space, but follow some sort of ordered structure. For molecular conformations problem (from protein folding to atomic clusters), the objective function shows the so called *funnel structure* ([29]), i.e. the objective function can be imagined as the sum of a "simple" one (with a single or few local minima) and a "noise" that introduces a high number of local minima (see Figure 1.5 for an example and [52, 31] for possible definitions).

In this case, it would be more effective, instead of sampling completely at random, to sample in the *neighborhood* of the current point, introducing a new *intensification* phase, but at a different "level", instead of considering continuous local optimization, taking into account adjacent local optima, i.e. a sort of combinatorial structure between them (see [31] for a discussion on the structure of global optimization problems and the difficulty associated to them). The resulting algorithm that we will call, for similarity with a strategy used in combinatorial optimization, Iterated Local Search (ILS) is stretched in Algorithm3: after generating an initial starting point on the overall search space, new points are generated "near" this one, using an appropriate LocallyGenerate() function. When one of such new points improves upon the current "center", it substitutes it (see lines 5-6 of Algorithm 3), and the search continues in a new neighborhood.

A common StoppingRule is waiting until a maximum number of consecutive *unsuccessful* iterations are performed, where unsuccessful means that the new generated

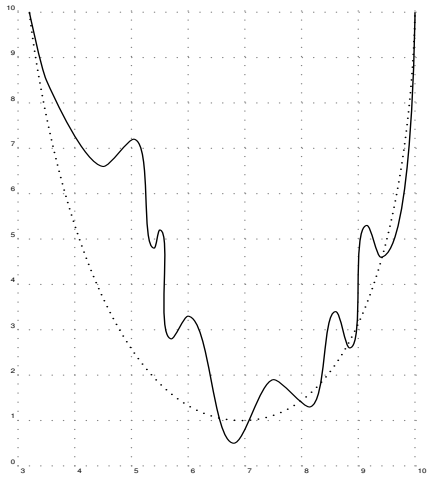


Figure 1.5: An example of funnel function

Algorithm 3 `Iterated Local Search`

```
1: procedure ITERATED LOCAL SEARCH
2:     x* =GloballyGenerate()
3:     while StoppingRule is false do
4:         z=LocallyGenerate(x*)
5:         if f(z) < f(x*) then
6:             x* = z
       return x*, f(x*)
```

point does not improve with respect to the current record. This stopping criteria allows to continue the search, as long as there is an improvement, and therefore to explore each new neighborhood.

The efficiency (and efficacy) of ILS depends strongly upon the definition of the LocallyGenerate() function. If the neighborhood is too small[6], then the algorithm stalls, if it is too large, ILS behaves similarly to a PRS, losing all the advantages obtained introducing the LocallyGenerate procedure.

If the ILS is applied to $\mathcal{L}_f$, we obtain the algorithm known as Monotonic Basin Hopping (MBH), that is illustrated in Algorithm 4 and that resulted very effective in solving many differently optimization problems ([53],[21]) and it is the basic element of the works I present in Sections 1.2- 1.3 and Chapter **??**.

Just as an illustration (a formal proof would ask more precise definitions and hypothesis), let us consider again the funnel function in Figure 1.5 and the result after applying an ideal local optimization in Figure 1.6. If the LocallyGenerate() function is defined

[6]smaller than the basin of attraction of the minimum inside the neighborhood

Algorithm 4 `Monotonic Basin Hopping`

```
1: procedure MONOTONIC BASIN HOPPING(N)
2:     x* =GloballyGenerate(), n = 0
3:     while n < N do
4:         z=LocallyGenerate(x*)
5:         z = L(z)
6:         if f(z) < f(x*) then
7:             x* = z, n = 0
8:         else
9:             n = n + 1
       return x*, f(x*)
```

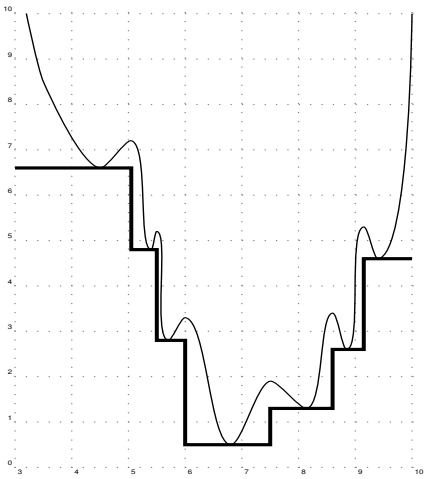


Figure 1.6: The effect of local optimization

correctly, i.e. if it defines a neighborhood such that it can contain the largest basin of attraction, and the stopping criterion allows to sample enough points in such neighborhood, then, MBH will converge to the global optimum with probability one. In fact, from any starting point in the domain there exists a descending (stair-case) path to the global optimum.

Some variants of MBH, that are quite general to be applied to different GO problems, but more flexible, are based on adapting the LocallyGenerate function to the current minimum point, similarly to Variable Neighborhood search for discrete optimization, as in Adaptive MBH methods ([31]). In [J5] and [J4], we proposed some methods that taking into account that MBH can be seen operating on the $\mathcal{L}_f$ function, try to determine search directions directly on the resulting step-function using a smoothing operator (see Figure 1.7). These methods proved to outperform the standard version of MBH and the Adaptative one on several difficult global optimization test functions (but they can result slower in terms of computational time).

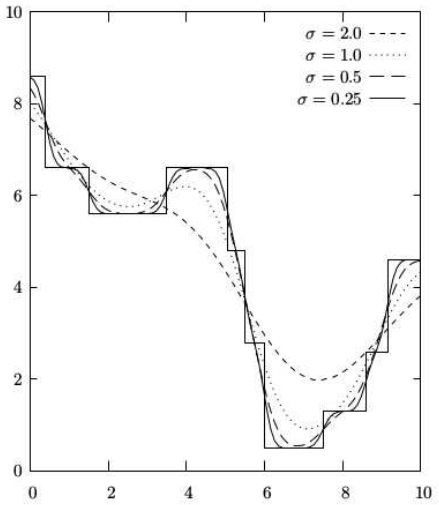


Figure 1.7: Example of different smoothing operators on $\mathcal{L}_f$

If the function does not present a single funnel, but multiples, as shown for example in Figure 1.8, the MBH strategy is not any more sufficient to have the guarantee to find the global optimum (even in probability). In fact, MBH can explore a single funnel and therefore can find only one of the different funnel bottoms (each one depending on the starting point).

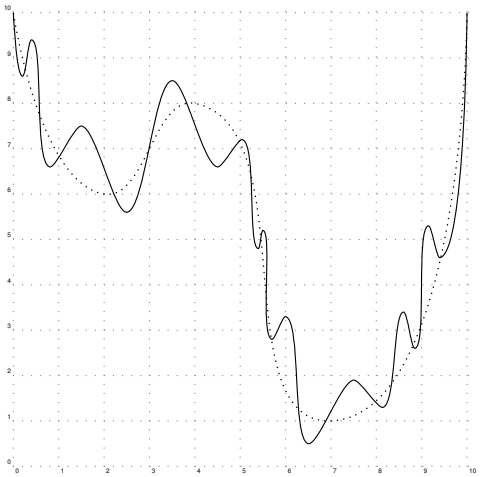


Figure 1.8: An example of a 2-"funnels" function

An *exploration* phase must be added to the method, that by construction is monotonically descending. Ideally, such phase must be able to allow exploring "completely" the search space. We mention some of the most common strategies that contain an exploration step:

- Repeating MBH from different randomly generated starting points. This strategy can be seen as the combination of a MBH and a Multistart where each "local search phase" of Multistart is performed as a MBH iteration.
- Simulated Annealing, eventually applied on $\mathcal{L}_f$. See Algorithm 5, where $P(\beta, z, x) = \min\{1, e^{-\beta(f(z)-f(x))}\}$ is a possible acceptance function, β^{-1} is commonly called temperature.

Algorithm 5 `Simulated Annealing`

```
1: procedure SIMULATED ANNEALING(β)
2:     x* =x=GloballyGenerate()
3:     while StoppingRule is false do
4:         z=LocallyGenerate(x)
5:         Extract a random probability P
6:         if P ≤ P(β, z, x) then
7:             if f(z) < f(x*) then
8:                 x* = z
9:             x = z
10:            Update β
       return x*, f(x*)
```

We can observe that if the temperature tends to zero we get exactly a PRS (or Multistart if applied on $\mathcal{L}_f$) and when it goes to ∞ we get an ILS method (or MBH).

- Using a Population-based MBH (PMBH): instead of working on one local solution at the time like in MBH, a "population" of K local solutions P (the so called "parents") is considered. Each of these solutions is perturbed and locally optimized to get a population of "children" C. This step can be seen as a single iteration of MBH using as current center each parent(see lines 4-5 of Algorithm 4). Then children are compared with parents to decide if they must be integrated in the current population. Differently from running K MBH in parallel, children are not necessarily compared with their own "direct" parent (the solution that was perturbed to get to them), but with the element in P that is closest to them. The idea behind this strategy is to try to compare solutions that are "close" and therefore allow the algorithm to locally explore each single funnel and at the same time keep a population that can represent well the overall search space (in some sense this strategy allow to counter-balance the intensification effect introduced by MBH for

the overall algorithm, and at the same time, for each sequence parent-children-parent enforces it). Therefore, it is necessary to define a "distance" function D between two solutions. For each child j, we search for the "least dissimilar parent"

$$n(j) = \arg \min_{i \in 1,\dots,K} \{D(C_j, P_i)\}$$

Let call $D(j) = D(C_j, P_{n(j)})$ the value of the minimum dissimilarity found and $\bar{D}$ the average distance in the current population P. At each iteration we decide if the new elements (children) must be inserted in the current population by one of the following criterias:

- if $D(j) < \bar{D}$, and C_j is better than $P_{n(j)}$: children C_j substitutes its closest parent $P_{n(j)}$;
- if $D(j) \geq \bar{D}$, then child C_j is sufficiently different: children C_j is added to the population P for the next iteration (or it substitutes the worst member)

We can imagine that, if function D is defined to recognize solutions belonging to the same funnel as similar, this strategy allows each element of the population to explore a single funnel. In particular, if an new element is found that is "enough" different from the current onces, it can be imagined that a new unexplored space region is detected.

In Sections 1.2-1.3, I will describe two family of problems (circle packing and space trajectory design) and how the presented ideas where adapted (and enriched) to solve some variants of them. The resulting methods' quality was proved by their capability to find new putative optima and to allow obtaining very good results in open optimization competitions.

1.2 Two circle packing problems

The problem of placing non overlapping objects belonging to $\mathbb{R}^d$ within a "smallest" container is a classical mathematical problem with important applications in manufacturing and logistics and, in particular, to problems related to cutting and packing – see e.g. [6] for a survey of applications in the fields of cutting, container loading, communication networks, facility layout.

The most studied case is the packing of *equal* circles in the unit square. Optimality proofs do not exist for the majority of cases, except of very small size problems. Therefore, to compare algorithm results it is necessary to have a common ground. For packing problem a reference is the E. Specht's web site [44] where putative optima for many different packing problems are reported, and in particular, for the packing of n equal circles in the unit square all best known solutions for up to 300 circles (at the time of our research, now it is up to 900) are reported.

To solve this problem many different approaches have been proposed as computer-aided optimality proofs [10, 40, 36], branch-and-bound approaches [34], and - due to the difficulty of solving exactly the problem - a significant number of heuristic approaches including a Multistart based on the unconstrained minimization of a properly defined energy function [39], a billiard simulation method [20], a two-phase approach with a first approximation phase based on local moves of circles along appropriately chosen directions and with an exponentially decreasing step-size and a second refinement phase based again on billiard simulation [3], an approach based on random perturbations of circles and possible acceptance of non improving moves [5].

Different variants of circle (and other 2-dimensional shapes) packing exist in the literature. Just to cite some of them in connection with our contribution, it is worth to mention the case where circles might have different sizes. In [45] the problem of placing circles with different sizes into a rectangular container with fixed width and minimum height is attacked by a combination of a branch-and-bound approach and a reduced gradient algorithm. In [23] a heuristic procedure is proposed to place circles with different sizes into a circular container with minimum radius, combining a quasi-physical approach (based on squeezing and collisions between circles) and a quasi-human approach (a circle compressed by many others, possibly including the external container, is moved to a new random position in order to get some "relief"). In [54] a build-up algorithm is proposed to sequentially place circles into a circular container. In [24] the minimization of the radius of the circular container is carried out through a combination of tabu search and simulated annealing ideas. Finally, in [6, 42] a global optimization approach has been tested.

In the following, two heuristic algorithm both based on MBH and PMBH methods will be described to solve two circle packing problems. The first has been defined for solving the packing of n equal circles in the unit square ([J3]), and the second for solving a "discrete" variant of circle packing proposed in an international competition based on unequal circles in a circular container ([J2]).

1.2.1 Packing n equal circles in the unit square

This packing problem is defined as follows: given a natural number n, find the maximum radius r that allows to place n non-overlapping circles of radius r inside the unit square. Different formulations are possible for such a problem, if we represent by x_i, y_i the coordinates of the center of the i-th circle, we can write:

$$\begin{aligned} r_n = \max\ & r & & \\ \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq r & & \forall\ i \neq j & \\ r \leq x_i \leq 1 - r & & i = 1, \ldots, n & \\ r \leq y_i \leq 1 - r & & i = 1, \ldots, n & \end{aligned} \tag{1.3}$$

where r_n denotes the objective value of the problem for n circles.

Using variable scaling the solutions of the previous problem are equivalent to the solutions of the following formulation ([12]):

$$\begin{aligned} r_n = \max\ & r & & \\ (x_i - x_j)^2 + (y_i - y_j)^2 \geq r^2 & & \forall\ i \neq j & \\ x_i, y_i \in [0, 1] & & i = 1, \ldots, n & \end{aligned} \tag{1.4}$$

Formulation 1.4 is one of the most common forms used in numerical optimization methods, and the one we used in our experiments. The objective is linear, but constraints are reverse convex[7]. Thus this problem is nonconvex and highly multimodal. However, with respect to general problems with nonconvex constraints, for which even finding feasible solutions may be an extremely hard task, here feasible solutions are easy to find[8].

Monotonic Basin Hopping

In order to define a MBH strategy, a neighborhood structure (or equivalently a perturbation) has to be defined. Let $\xi_n = \frac{\alpha}{\sqrt{n}}$ (in our computation we set $\alpha = 0.5$). For each variable z, where $z = x_i$ or y_i, $i = 1, \ldots, n$, if we denote by $\hat{z}$ the current value of this variable, then the new value of the variable is uniformly sampled over the interval

$$[\max\{0, \hat{z} - \xi_n\}, \min\{1, \hat{z} + \xi_n\}] \subseteq [0, 1].$$

Some comments are needed at this point. First of all we need to comment our choice of the value for ξ_n. It is well known (see, e.g., [8]) that

$$r_n \sim \frac{2^{1/2} 3^{-1/4}}{\sqrt{n}}. \tag{1.5}$$

Then, let us consider an "active" pair of points of an optimal solution (a pair of points is said to be active if the distance between the two points is equal to r_n, i.e., if the

[7] the resulting feasible set represented by any of these constraints is the complement of a convex set, in this specific case the "outside" of a circle

[8] Sampling n points in the interval $[0, 1]^2$ and setting $r = 0$ produces a feasible solution

associated constraint in (1.4) is active). The maximum possible step-size allowed by our perturbation method is given by $\sqrt{2}\xi_n$ which is only slightly greater than $r_n/2$. This basically means that after the perturbation, with a very high probability, both points will still lie on the same side of a maximally separating line (i.e., a perpendicular line through the midpoint of the line segment connecting the two centers). This way the structure of the previous solution is still partially preserved, as it has to be for local moves. Also note that the perturbation of a point only affects its neighbors (it is easily seen that all points at distance larger than $r_n + 2\sqrt{2}\xi_n$ from a given point will still be at a distance larger than r_n after the perturbation, and therefore they will not have an impact on the objective function).

Next we note that there is some asymmetry in the perturbation of a point when this is close to the border. Indeed, in this case the perturbed point will be generated with a higher probability "towards" the interior of the unit square rather than "towards" the border (just think about the perturbation of a point lying at a vertex of the unit square). Therefore, this perturbation mechanism somehow acts as an implicit compression tool, which might have a positive effect. Indeed, for molecular conformation problems it has been observed that the insertion of explicit compression tools has extremely positive effects in detecting the most challenging global minima (see for example [35]).

Finally, we remark that our perturbation is still incomplete. Indeed, Problem (1.4) has a further variable which is r. This variable is perturbed in a different way with respect to the others. In order to generate a perturbed point within the feasible region of problem (1.4) the value for r has to be chosen in $[0, \bar{r}]$, where $\bar{r}$ is the minimum distance between points of the perturbed solution, i.e.,

$$\bar{r} = \min_{i \neq j} \sqrt{(\bar{x}_i - \bar{x}_j)^2 + (\bar{y}_i - \bar{y}_j)^2}. \tag{1.6}$$

It has been experimentally observed that setting $r = \bar{r}$ is not an efficient choice, while the choice $r = 0$ turns out to be quite efficient. This can be explained as follows. With $d = \bar{d}$ the perturbed solution lies on the border of the feasible region, while with $d = 0$ the perturbed solution almost surely lies in the interior of the feasible region, which gives more freedom to the local search procedure started from the perturbed solution.

Population MBH

From the computational experiments, it seems that larger instances present an higher number of different configurations. Following a successful approach developed for the optimization of molecular structures [22], we implemented a Population based version of our MBH. As already mentioned, in order to implement this strategy, a measure of dissimilarity is needed. Inspired by the ideas successfully used for molecular clusters in [30], we defined a dissimilarity function based on the number of "neighbors" of each circle in a given solution. Let δ be a threshold and represent with $X = \{X_i \in \mathbb{R}^2\}_{i=1}^n$ the coordinates of the centers of the circles in a given disk packing. Given a disk i, let define the number of disks whose distance from disk i does not exceed δ:

$$N_\delta(X, i) := |\{j \in 1, \ldots, n, j \neq i : \|X_i - X_j\| \leq \delta\}|$$

We can count how many circles has k neibourghs, using histograms:

$$H_\delta(X,k) := |\{i \in 1,\ldots,n : N_\delta(X,i) = k\}|$$

Then we can compare two packings X and Y evaluating

$$D_\delta(X,Y) = \sum_{k=0}^{n}(k+1)|H_\delta(X,k) - H_\delta(Y,k)|.$$

Thanks to the weighting factor $k+1$, more weight is given to configurations in which the number of disks with many neighbors is different. As a general rule it is advisable to choose a value for δ which is somewhat larger than the expected minimum distance between pairs of points. In fact, in many configurations, even putative optimal ones, there are "free disks" (a.k.a. "rattlers"), i.e., disks that can be moved without violating any constraint. Therefore, these disks might or might not be in contact with others, in different equivalent solutions, so that through a dissimilarity measure (with a too small δ) two configurations which differ only for the location of the rattlers might erroneously be considered as different. In our computations we usually set $\delta = 1.5/\sqrt{n}$.

This measure of dissimilarity is quite easy to compute and displays a very good discriminating power; moreover it is invariant to rigid transformations of the packing configurations, for example, given any solution and the equivalent one that is obtained swapping coordinates x with coordinates y ($\pi/2$ rotation) present a dissimilarity of zero.

A short summary of the computational results

In our computational experiments, for the standard version of MBH we employed $N = 100$ (maximum number of consecutive iterations without a success), while for the population variant (PBH) we choose 40 elements in the population and $N = 30$; in PBH we used this parameter on the whole population, i.e., we choose to stop the algorithm as soon as 30 generations of children had been obtained with no improvement in the best one. For local optimization, we used SNOPT[9] [18] with feasibility tolerance parameters set to 10^{-12}.

As already mentioned, putative optimal configurations are maintained by E. Specht on his web site [44]; there, after a new finding for $n = 97$ in January 2003, Specht comments: *"It seems to be very unlikely to find still better packings for $n \leq 100$, but it is possible"*. Basically, since many different methods had been tested on these instances, it seemed to be reasonable to conjecture that packings known at that time for n up to 100 were indeed the optimal ones. Therefore, in order to check the robustness of our method the first aim was to show that we were able to efficiently reproduce at least all the best known packings up to $n = 100$. What we obtained was actually much more than that. Indeed, in this range we have been able to detect many new best known solutions, now reported in [44] and to confirm most of the remaining. Stimulated by this success we performed numerical experiments up to $n = 130$ again obtaining many improved

[9] a large scale SQP method

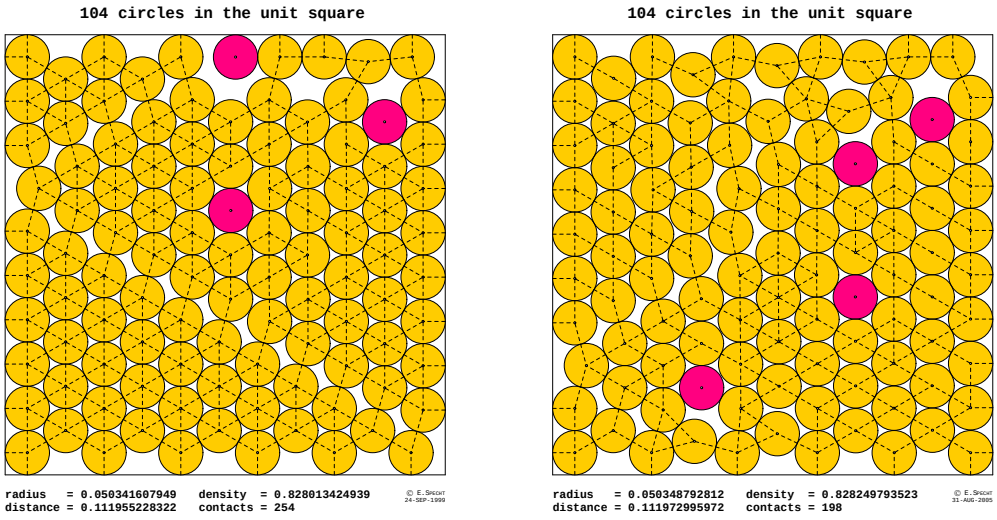


Figure 1.9: Putative optimal packings for $n = 104$: previously known record (left, $d = 0.111955228322$) and new putative optimum (right, $d = 0.111972995972$)

putative optima. In the Table 1.1, the results for $n \in [50, 130]$ at the time of writing our paper [J3] are reported: in normal typeface the values of n for which we could obtain the previously known putative optimum [10]; in **bold** the values of n corresponding to cases in which we could improve over previously known putative optima and, finally, in *italic* those instances for which we were not able to obtain, within the prescribed accuracy, the putative optimum. We remark that in this problem an apparently small improvement in the distance can, and usually does, imply a significantly different geometry. Just for illustration we present in Figure 1.9 a comparison between the previous putative optimum and our improved configuration for $n = 104$. In this figure, dark disks are "rattlers".

Many of the new records and (when no new record has been detected) of the old putative optima have been found several times by both MBH and PMBH. For what concerns failures, only two cases have occurred. $n = 112$ could not be found even after running 100 PBH independent experiments. The case $n = 75$ is particularly significant: we could indeed find a configuration whose distance is just 10^{-10} smaller than that of the putative optimum, but most of the times we reached a configuration whose distance is more than 10^{-7} smaller than that of the putative optimum. In order to check whether the optimal configuration could have been recovered by means of local perturbations, we adopted the technique described in [15] to obtain a rigid movement of disks which eventually led to an "unjamming" of our best configuration. From the resulting configuration of this procedure a single run of a local optimization method led us to the putative optimal solution for $N = 75$. The same procedure was however not successful for $N = 112$,

[10] with a maximum error on the distance of 10^{-12}

50	51	52	**53**	54	55	56	57	58	**59**
60	61	62	63	64	65	66	67	**68**	69
70	71	72	**73**	74	*75*	76	**77**	**78**	79
80	81	82	83	84	**85**	**86**	87	**88**	89
90	91	92	93	94	**95**	96	97	**98**	99
100	**101**	102	103	**104**	**105**	**106**	**107**	**108**	**109**
110	**111**	*112*	**113**	114	**115**	**116**	**117**	**118**	**119**
120	**121**	**122**	**123**	124	125	126	**127**	**128**	**129**
130									

Table 1.1: Results obtained by executing 100 independent experiments with MBH for $n \leq 100$ and 10 independent runs with PMBH for all values of n.

which thus remains the unique failure of our method for $N \leq 130$.

Some observations on the formulation and random generation

We observe that solving problem (1.4) is equivalent to solving any problem with the following form:

$$\begin{aligned} &\max \ f(r) \\ (x_i - x_j)^2 + (y_i - y_j)^2 \geq g(r^2) \qquad &\forall\ i \neq j \\ x_i, y_i \in [0,1] \qquad &i = 1, \ldots, n \end{aligned} \tag{1.7}$$

where f is a monotonic increasing function and g is some continuous function satisfying

$$g(r^2) \begin{cases} = r^2 & \text{for } r \geq r^\star \\ \leq r^2 & \text{for } r < r^\star \end{cases}$$

where $r^\star$ denotes a lower bound for the optimal value of (1.4). Although any function f and g satisfying the above conditions leads to formulations which are equivalent from the theoretical point of view, considerable differences can be observed in the practical behavior.

Furthermore, we tested the influence on performance of the addition of a simple explicit compression mechanism: after having perturbed a solution, we further modify this solution by multiplying all of the coordinates by a factor $c \in (0,1)$ (in particular, we tested $c = 0.8$); thus the resulting configuration shrinks to fit within the square $[0,c]^2$.

In order to test the impact of different choices, we selected four test environments (listed in Table 1.2) and performed for each of them 100 runs with n ranging from $n = 31$ (the first n value for which a computer-aided proof of optimality has not been given yet) up to $n = 100$. Note that choice C1 is equivalent to (1.4) and that the introduction of the *tol* value is necessary to guarantee the differentiability of the objective function at $r = 0$. Also note that at each iteration of a MBH run, the $r^\star$ value is the current record value for that run.

Choice	f	g	Compression
C1	r	r^2	No
C2	$100r$	$r^2 - [\max\{0, r^\star - r\}]^2$	No
C3	$100\sqrt{r + tol}$	$r^2 - [\max\{0, r^\star - r\}]^2$	No
C4	$100\sqrt{r + tol}$	$r^2 - [\max\{0, r^\star - r\}]^2$	Yes

Table 1.2: Different tested choices ($tol = 10^{-5}$).

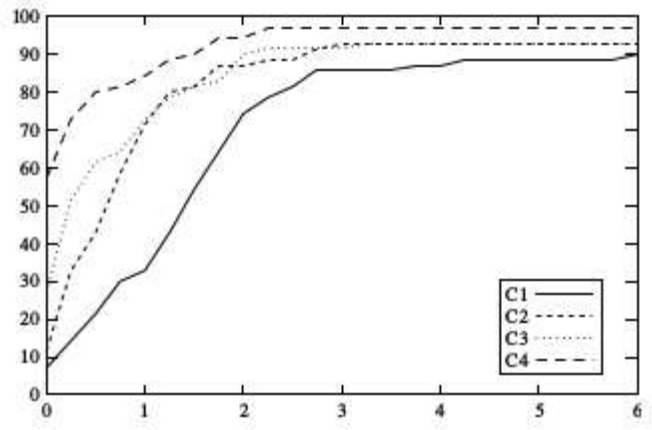


Figure 1.10: Performance profiles of the four tested choices. Test choice C4 is the best one in 57% of the problems while in 84% cases its success rate is at least 1/2 with respect to the best one.

In Figure 1.10, we report – in a very compact form – results for the different variants in the form of performance profiles, introduced in [14] with the aim of comparing different solution algorithms (let us represent it by set $\mathcal{C}$) on a given test set ($\mathcal{P}$). In what follows, we give a brief explanation of how to interpret them. Let us define a performance measure perf(c, p) for each algorithm (strategy) c and for each instance (problem) p. Examples can be: the total running time, the value of the best solution found. In our case, we consider the putative optimal radius $r_p(c)$ found by choice c on instance p and define the following "counting" function:

$$\text{Perf}_c(x) = \frac{100\|p \in \mathcal{P} : r_p(c) \leq 2^x \cdot \min_{i \in \mathcal{C}} \quad r_p(i)\|}{\|\mathcal{P}\|} \tag{1.8}$$

This curve represents the percentage of problems where choice c has a performance that is not worst than 2^x times with respect to the best choice. Therefore, on the ordinate it can be read the percentage of instances where the algorithm finds the putative optima, for $x = 1$ the number of tests were the gap with the putative optima is smaller than or equal to 100%, and so on. Very roughly speaking, up-left positioning in the figure is a sign of good performances.

We can observe that C4 strategy is the most effective, showing that an "intelligent" choice of the starting points (in this case the utilization of a compressor factor) is a key element, doubling the number of successes with respect to the same formulation without compression (C3). Also the choice of a different formulation (from C1 to C2, and from C2 to C3) has an impact, increasing not only the number of successes (twice from C2 to C3), but also the quality of the overall local solutions (not only the putative one).

1.2.2 Circle packing contest

In this section we present the solution strategy we used to tackle the circle packing competition proposed in the "Al Zimmermann's Programming Contests"[55]: given a positive integer n, place n disks of radii, respectively, equal to $1, 2, \ldots, n$ inside a circle whose radius (r_n) is minimal. The participants were asked to propose solutions for all n in $5, \ldots, 50$, and a total score was assigned to each participant considering their results for each given size. More precisely, for each submitted solution of size n, a quality measure is defined:

$$Q_n = \frac{\text{best} \quad r_n}{\text{submitted} \quad r_n} \tag{1.9}$$

where best r_n represents the value of the best found solution for size n among all participants and no submission for a given n produced a value $Q_n = 0$. The total score was given by:

$$\sum_{n=5,\ldots,50} (0.7Q_n + 0.1(Q_n^{16} + Q_n^{128} + Q_n^{1024})) \tag{1.10}$$

Therefore, the maximum attainable score was of 46 ($Q_n = 1 \forall n = 5 \ldots 50$).

Let consider a mathematical programming formulation for the problem:

$$\min R \tag{1.11}$$

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq i + j \qquad \forall\, 1 \leq i < j \leq n \tag{1.12}$$

$$\sqrt{x_i^2 + y_i^2} \leq R - i \qquad \forall\, 1 \leq i \leq n. \tag{1.13}$$

where, as in the previous problem, x_i, y_i denote the coordinates of the center of circle i. Constraints (1.12) force circles i and j not to overlap, while constraints (1.13) force each circle to be included in the disk with radius R centered at the origin. Although the objective is linear and constraints (1.13) define a convex region, constraints (1.12) are reverse-convex (as for the previous packing problem).

In comparison to the problem of placing identical circles in a smallest container, here the fact that each disk has a different radius generates new difficulties and challenges. In fact, while the problem with identical radii can be considered as a pure continuous optimization problem, here the fact that each circle has a different radius adds some sort of combinatorial structure over the original one. Moreover, although a direct application of this model to real life problems is difficult to imagine, nonetheless the ideas used in attacking this problem might find an application in methods for optimal placement of figures with the same shape but different sizes in a container.

Strategies

We developed several models and algorithms during the course of the contest. Many of them were discarded as non productive, just a few of them survived until the last weeks and gave us the possibility of winning against 154 competitor teams. In the following, we outline the main ideas underlying most of the models and methods we tried.

MBH Given our experience in the field of molecular conformation problems [16] and in equal circle packing [J3], we decided to try a similar approach also in this case.

An initial solution was found by drawing the coordinates of the center of each circle from a uniform distribution on a suitably large box[11]. After the generation of circle centers, it proved useful, sometimes, to re-scale them in order to avoid circle overlapping.

We tested different perturbation rules but ended up with the following: two circles, with similar radii (the difference between the radii could not be larger than 2), were chosen at random and their radii exchanged. This is a combinatorial neighborhood exploration and it proved to be much more efficient than, e.g., continuous random displacement of the circle centers. After swapping two circles either we restored feasibility through re-scaling or not (usually 50% of the times).

A critical issue both for the initial generation and for the perturbation was related to the choice of the initial value for the R variable. Our local optimizer, SNOPT [18], was extremely sensitive to the initialization of the R variable prior to local optimization (the same has been observed for other nonlinear solvers). Choosing to initialize R in the most natural way, i.e. the radius of the smallest circular container containing all circles, gave the initial solution too much rigidity and, often, the local optimizer did not produce any useful solution (similarly to what we obtained for the packing of equal circles in a square). We tried different other possible values for R and the best choice turned out to be setting R slightly (say 10%) larger than the radius of the smallest circular container containing all circles. This strategy can be considered similar to the compression technique we presented for the packing of equal circles in the square.

Reduction of the space of variables It was quite early observed that in putative optimal configurations of N circles, the smallest ones did not play any role and very often they were "rattlers", i.e. disks which could be displaced to a different position without violating any constraint nor worsening the objective. Thus we decided to solve a reduced optimization problem in which instead of placing disks of radii $1, \ldots, N$, we only looked for optimal configurations of $N-m$ circles of radii $m+1, \ldots, N$, thus discarding the first m. The value of m we chose was usually around $0.25N$. Removing smallest circles had three beneficial effects:

- first, the dimension of the feasible space is reduced, which make local optimization slightly easier and quite significantly faster;

[11] usually chosen as $[-N^2/2, N^2/2]$

- second it allows a greater freedom in choosing the positions of the remaining disks, which otherwise could be artificially constrained by the position of small circles which could have found a place somewhere else without any difficulty;
- finally, we observed that removing smaller disks has the effect of reducing the number of local optima – in particular, all locally optima packings which differ only in the position of the smallest circles are reduced to a single one.

Obviously, after optimization with MBH, we had to restore the eliminated circles, if possible, without enlarging the container radius. This was done by sequentially re-inserting missing disks, one at a time starting from the largest one (see [J2]).

Backward and forward moves Since optimal configurations for close N values might be quite similar, quite regularly we also tried backward and forward moves in search of improvements. In particular, while looking for an optimal configuration for N circles, we tried to start from a known solution with $N+1$, in which we eliminated the smallest circle (or the $m+1$ smallest ones in order to have only $N-m$ circles as described before) and reduced all the radii by one. Analogously, for forward moves, we started from a configuration of $N-1$ circles, removed the $m-1$ smallest circles in order to have only $N-m$ circles, augmented all the radii by one, re-scaled in order to have a feasible starting solution. The resulting configurations were inserted as the "good enough" ones within the population-based approach (see next point).

Population based approach Another strategy we implemented with success was that of working with a population of solutions. As before, let us represent with $X = \{X_i \in \mathbb{R}^2\}_{i=5}^{n}$ the coordinates of the centers of the circles in a solution, therefore, X_i represents the coordinates of the circle of radius i. Letting ρ_i^X be the distance from the origin to the center of circle i in the solution X. Then given two solutions X and Y, we defined the following pairwise dissimilarity measure:

$$D(X,Y) = \sum_{k=\lceil N/2 \rceil}^{N} k|\rho_k^X - \rho_k^Y|$$

This measure takes into account only the largest circles and is based on their distance from the center of the container. Furthermore, it gives more weight to different positions of larger circles, as this is a good indicator of significantly different structures. Note that this dissimilarity measure is invariant under rotation.

The initial population was mostly generated at random; however we chose to generate one of its elements as an already "good" one. In particular, this element might be: (a) *the current record*; (b) *the result of a MBH run*; (c) *the result of a backward or forward move* (as described previously).

Inserting a "good enough" element in the initial population turned out to be an extremely beneficial choice. Indeed, we observed that the population is very quickly filled in with neighbours of this "good enough" element and the population-based

strategy allows to explore more evenly the state space of possibly worse but still good configurations around this element, avoiding the danger of being too greedy. We observed, in several occasions, that improvements were consequences of the phenomena of backtracking (a child is worse than its father but enters the next population) and survival (a father is worse than its child but survives in the next population) as observed for molecular conformation problems (see [22] for more details).

Note that, while using as "good enough" element the current record or the result of forward and backward moves does not guarantee a full exploration of the search space, using the results of MBH with random restarts guarantees a complete exploration of this space.

In Table 1.3 we report all the best results obtained during the competition together with our results and, within parentheses, the ranking of our result with respect to all the others for each size. The list of all records as well as their coordinates can be obtained from the contest web site [55].

n	Best	Our	Rank	n	Best	Our	Rank
5	9.00139774	9.00139774	**1**	29	99.51231790	99.51231790	**1**
6	11.05704039	11.05704039	**1**	30	104.57855508	104.74779928	3
7	13.46211067	13.46211067	**1**	31	109.77194698	109.77194698	**1**
8	16.22174667	16.22174667	**1**	32	114.86543833	114.90022384	2
9	19.23319390	19.23319390	**1**	33	120.21695714	120.24932984	2
10	22.00019301	22.00019301	**1**	34	125.43350175	125.61242286	2
11	24.96063428	24.96063428	**1**	35	131.15635463	131.16650268	2
12	28.37138943	28.37138943	**1**	36	136.53490083	136.53490083	**1**
13	31.54586701	31.54586701	**1**	37	142.17498053	142.25633406	2
14	35.09564714	35.09564714	**1**	37	142.17498053	142.25633406	2
15	38.83799550	38.83799550	**1**	38	147.85769135	147.99157574	2
16	42.45811643	42.45811643	**1**	39	153.55530119	153.60382813	2
17	46.29134211	46.29134211	**1**	40	159.48902487	159.57390300	3
18	50.11976262	50.11976262	**1**	41	165.29190968	165.29190968	**1**
19	54.24029359	54.24029359	**1**	42	170.92576161	170.92576161	**1**
20	58.40056747	58.40056747	**1**	43	177.07434007	177.23962454	2
21	62.55887709	62.56005858	2	44	183.17606157	183.37007125	2
22	66.76028624	66.76028624	**1**	45	189.63543910	189.67917387	2
23	71.19946160	71.19946160	**1**	46	195.91076339	195.91076339	**1**
24	75.75270412	75.75270412	**1**	47	202.18561174	202.22801940	2
25	80.28586443	80.28586443	**1**	48	208.63594672	208.63594672	**1**
26	85.07640122	85.10628281	3	49	214.66195201	214.66195201	**1**
27	89.79218156	89.82957381	4	50	221.08975259	221.08975259	**1**
28	94.54998647	94.70586218	3				

Table 1.3: Best and our solution values, plus our ranking with respect to all submitted solutions

1.3 Space mission analysis

A very important problem in mission analysis is the planning of interplanetary trajectories, which consists in launching a spacecraft from a given astronomical body (usually the Earth) along a trajectory which leads to some other astronomical body. The aim of the mission can be to land on the body or to put the spacecraft into the planet's orbit and the objective of a model is to help trajectory planners in taking the best decision, e.g., on the starting date and other relevant parameters, in order to obtain a "low-cost" mission.

In Figure 1.11 is illustrated the real trajectory of the mission Rosetta, a probe build by ESA and launched on the 2th of March 2004. The probe collected data during several fly-bys [12] and finished its mission by hard-landing on the comet 67P/Churyumov-Gerasimenko the 10th November 2014. The full lines (in color) represents the orbits of the different astronomical bodies (planets/asteroids/comet) encountered during the trajectory, the dotted lines represent the trajectory itself.

In general, accurate models are computationally very demanding. For this reason simplified models are typically used in the first planning phase: it is assumed that the spacecraft is equipped with a chemical propulsion engine and the aim is that of minimizing the overall energy spent during the mission.

Global optimization techniques are usually applied to these simplified models. These might return a set of good solutions, that can be further refined through more accurate and costly models.

Different papers (see, e.g., [1, 17, 28, 43]) propose Genetic Algorithms (GA) for this kind of problems. Evolutionary strategies, in particular Differential Evolution (DE), turn out to be a valid alternative (see, e.g., [9, 41]). Hybrid methods have also been proposed. In these methods either the problem structure knowledge (see [27]), or the results of some optimization methods (see [50, 48, 49, 46]) are exploited to evaluate portions of the feasible region and, consequently, to discard them or, alternatively, to intensify the search within them. Particularly relevant are recent studies carried on to compare the performance of different GO algorithms on different benchmark problems. The tested algorithms in these studies (see [11, 25, 26, 37, 46, 47, 51]) include, besides GA and DE, also Particle Swarm Optimization, Adaptive Simulated Annealing, GLOBAL, COOP, Multilevel Coordinate Search, DIRECT.

It is worth to note that, during last years, people at ESA ACT (Advanced Concept Team) carried on a considerable effort to make standard models of many benchmark problems (see `http://www.esa.int/gsp/ACT/inf/op/globopt.htm`), available both in C/C++ language and in MATLAB.

The currently available studies show that DE often performs quite well. However, a recent study [47] reveals that a basic version of Monotonic Basin Hopping (MBH, see, e.g., [29, 33]) is able to outperform some other algorithms, including DE, on some benchmark problems. This fact led us to propose a method based on the MBH approach.

[12] parts of the trajectory where the probe is close enough to an astronomical body

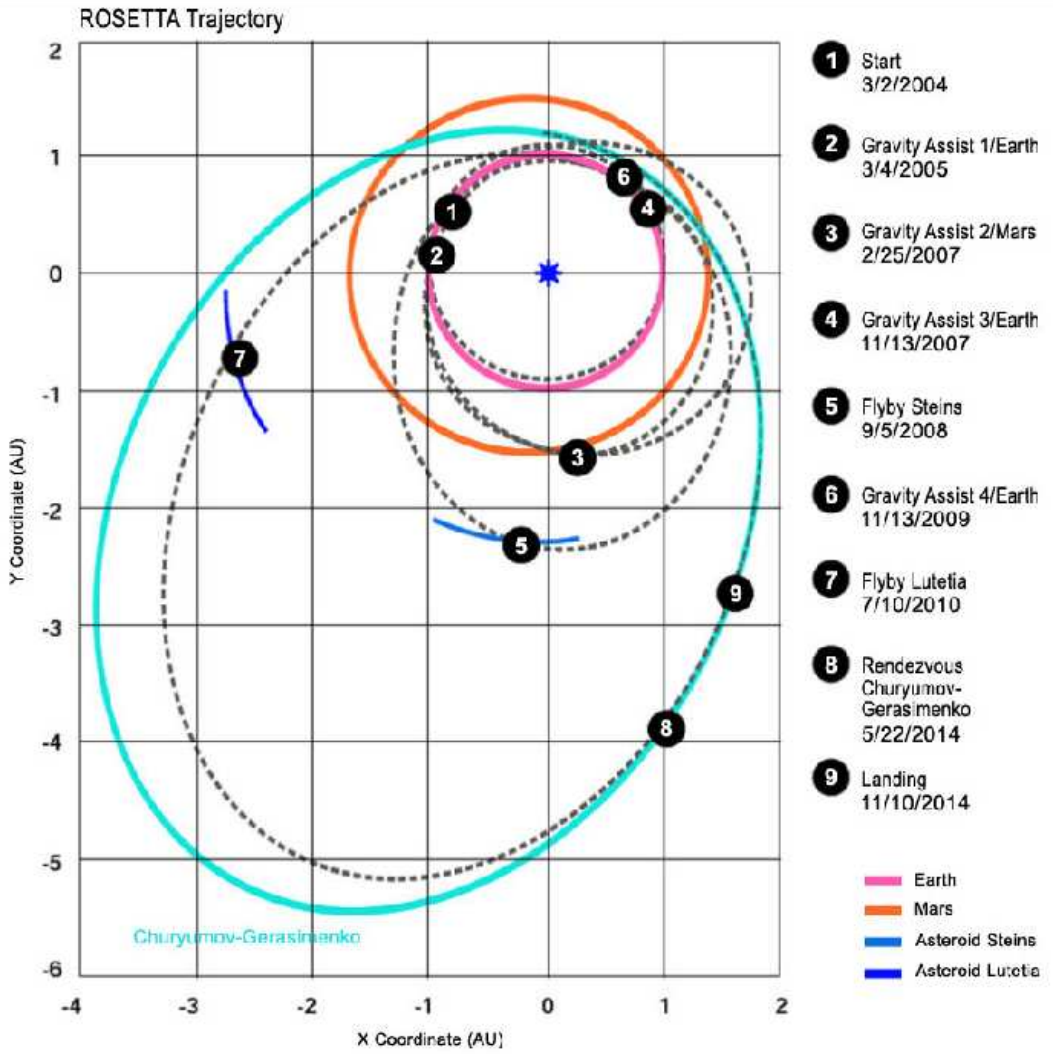


Figure 1.11: Trajectory of mission Rosetta

Before introducing the mathematical formulation and our method, we need to introduce some definitions.

Leg A *leg* is the trajectory followed by the spacecraft between two astronomical bodies (planets or asteroids). For example, the dotted line between number 2 and 3 represent a leg between the Earth and Mars.

Pericenter radius The *pericenter radius* at an astronomical body is the minimum distance between the trajectory of the spacecraft and the body (see Figure 1.12)

Swing-by A *swing-by* or *gravity assist* maneuver is the result of the gravitational interaction between the spacecraft and the astronomical body: as the spacecraft gets

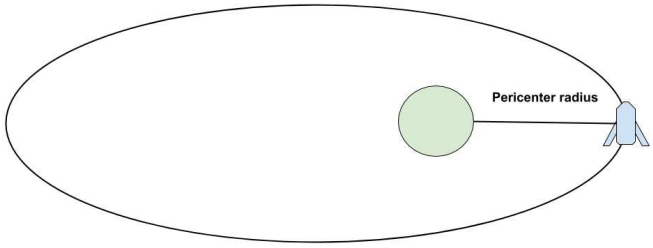


Figure 1.12: Pericenter radius

close to the body, such interaction does not change the modulus of the spacecraft velocity but changes its direction. The new direction depends both on the modulus of the velocity and on the pericenter radius. In our example, we have 4 swings-by: Earth (4/03/2005), Mars (25/02/2007), Earth (13/11/2007), Earth (11/13/2009).

Lambert's problem and arc Given two points in space and the time of flight between them, the trajectory followed by the spacecraft between the two points can be calculated by solving a *Lambert's problem*, which basically consists in the solution of a second order ordinary differential equation with boundary conditions. The resulting trajectory is called a *Lambert's arc*. Lambert's problems usually have multiple solutions but, exploiting some problem knowledge (and, actually, accepting the risk of excluding good solutions), we can introduce further assumptions which reduce the number of solutions to one (we refer, e.g., to [27] for more details).

Deep Space Maneuver A *Deep Space Maneuver* (DSM in what follows) is a change in the spacecraft velocity during a leg (the spacecraft is usually assumed to be able to thrust its engine at most once during each leg). When a DSM is added inside a leg, the leg can be considered as split in two Lambert's arc.

1.3.1 Problem definition and analysis

In the problems discussed in this paper we have a sequence of $n+1$ astronomical bodies $B_0, \ldots, B_n$ (B_0 is usually the Earth). The bodies are not necessarily distinct. We would like to visit the sequence in such a way that the overall energy consumption is minimized. Note that in what follows it is assumed that the sequence is *fixed*, but we may also think of models where the sequence of bodies is part of the decision problem, thus implying the introduction of discrete choices in the models.

MGA problem

The first model we consider is the Multiple Gravity Assist (MGA in what follows) model. In this model the variables are:

- t_0, the starting date of the mission;

- $T_i,\ i = 1, \ldots, n$, the time of flight along leg i (joining body B_{i-1} with body B_i).

Given the values of these variables, we are able to identify the positions p_{i-1} and p_i respectively of body B_{i-1} at time $t_0 + \sum_{j=1}^{i-1} T_j$, and of body B_i at time $t_0 + \sum_{j=1}^{i} T_j$. Therefore, the solution of the corresponding Lambert's problems allows us to identify the Lambert's arcs (which are conic arcs, either part of an ellipse or of an hyperbola) along all legs. It is then possible to compute the velocities at the end of each leg i and at the beginning of the following one $i + 1$, $i = 1, \ldots, n - 1$. In order to transfer from one leg to the next one, the spacecraft needs to provide a single impulse, a single change of velocity denoted by Δv_i. In the initial leg the spacecraft will have to provide a single impulse Δv_0 to leave the starting planet's orbit and reach the starting velocity at the initial leg. Similarly, in the final leg the spacecraft will have to provide a single impulse Δv_n to move from the final velocity in the last leg to the velocity of the target astronomical body B_n. Each impulse causes an energy consumption proportional to the modulus of the change of velocity. Therefore, in order to minimize the overall energy consumption, we are led to the following objective function:

$$\|\Delta v_0\| + \sum_{i=1}^{n-1} \|\Delta v_i\| + \|\Delta v_n\| . \tag{1.14}$$

Usually, MGA models also include constraints on the pericenter radius r_i at each intermediate body B_i, $i = 1, \ldots, n - 1$, which typically require r_i not fall below a given threshold r_i^{min}: the spacecraft needs to be far enough from body B_i in order to be able to leave the planet orbit and avoid a "forced" landing, due to the gravitational force. Such constraints can be kept as explicit ones or, alternatively, can be moved to the objective function, through the addition of properly defined penalization terms in (1.14). Note that the pericenter radius at each intermediate body is a dependent variable in the MGA model: once we have computed the Lambert's arcs, we can also derive it through appropriate formulae.

MGADSM problem

The second model allows for the introduction of DSMs and will be denoted by MGADSM. Such model is more flexible but also more complex. In particular, it requires the introduction of new variables besides those already discussed for the MGA model, in order to take into account the DSMs:

- the modulus and the direction (defined by two angles) of the spacecraft relative velocity at the initial body B_0 (V_∞, u, v)[13];
- the time instant in which each DSM maneuver takes place; usually these are formulated through the introduction of variables $\eta_i \in [0, 1]$, $i = 1, \ldots, n$ which define

[13]For the sake of precision, although related to angles, variables u and v are usually constrained to belong to the interval $[0, 1]$: they represent linear transformations of the polar coordinates of the spacecraft relative velocity at the initial body B_0 (see, e.g., [47, 51] for details).

at which portion of leg i the DSM maneuver occurs. More precisely, during leg i a DSM is performed at time

$$t^i_{DSM} = t_0 + \sum_{j=1}^{i-1} T_j + \eta_i T_i.$$

This way, instead of having a unique Lambert's arc during leg i, we have two of them, the first one joining the position (at time $t_0 + \sum_{j=1}^{i-1} T_j$) of body B_{i-1} with the position of the spacecraft at time t^i_{DSM}, and the second one joining the position of the spacecraft at time t^i_{DSM}, with the position of body B_i at time $t_0 + \sum_{j=1}^{i} T_j$;

- the pericenter radii r_i, $i = 1, \ldots, n-1$, at the intermediate bodies are now independent variables;
- finally, at each intermediate body B_i, $i = 1, \ldots, n-1$, we need to choose an angle b_i. The spacecraft's incoming velocity and the orbit's eccentricity of the Lambert arc joining the position of the spacecraft at time t^i_{DSM}, with the position of body B_i at time $t_0 + \sum_{j=1}^{i} T_j$, allow to define a cone, along whose surface the spacecraft's outgoing velocity lies: the value of angle b_i uniquely identifies the direction of such velocity along the cone's surface.

Each DSM requires a change of velocity and this implies an energy consumption which has to be included in the objective function.

All these models can be considered as **box-constrained black-box** ones and, like we did in this paper, no information on the problem, like, e.g., analytical derivatives, can be exploited.

1.3.2 A MBH for space trajectories planning

The core elements to define a MBH strategy are the perturbation (LocallyGenerate function) and the standard local search procedure (represented by $\mathcal{L}_f$). In theory, any standard local search method can be used to obtain $\mathcal{L}_f$, however this choice can change greatly the practical behavior of the method. We can further enhance the performance of the overall method by employing a "two-phase" local search. The first phase is aimed at driving the search towards promising portions of the feasible region (the result may not even be a local minimizer), while the second phase is a refinement one, actually leading to a local minimizer. In this sense, a two phase local optimization can be interpreted in two complementary ways: as a transformation of function $\mathcal{L}_f$, aiming at increasing the probability to reach the global optimum (introducing some global information) or, as a modification of the random generation/perturbation, i.e. the first local search can be seen as a refinement step of the random generation, moving from an "uniform" generation to one ideally concentrated near by the global optimum (or good local optima). For example, in molecular conformation problems the first phase is defined through the addition of geometric penalization terms to the objective function (see, e.g., [16],[O1]) driving the search through more compact solutions (that seem to be good candidates

for the global solution). Of course, the definition of a two-phase approach is strictly problem dependent and must be adapted consequently.

Local optimization inspired by Implicit Filtering Even if almost everywhere smooth, space trajectory problems usually display, even for small instances, a function landscape which is characterized by an enormous number of local optima: the objective function looks like a smooth one, perturbed by some sort of noise which generates many local optima. Examples can be seen for example in [37], in which bi-dimensional plots for the objective function in the case of Earth-Jupiter-Saturn, Earth-Mars and other missions, show the presence of many local optima often clustered in a periodic structure.

An interesting technique, called *implicit-filtering algorithm* (IF), has been introduced by Kelley [19, 38] for the case of box-constrained problems for which the objective function can be thought as $f(x) = f_s(x) + \phi(x)$, where $f_s(x)$ is smooth and $\phi(x)$ is not differentiable but such that $f_s(x) \gg \phi(x)$. The basic idea is to substitute the exact gradient with a finite difference estimate and use it inside a standard descent algorithm for smooth optimization. The approximated gradient is computed by forward or centered finite differences, with a step h which, starting from a relatively high value, is gradually decreased during the iterations; this way convergence properties can be derived (see [7]).

We decided not to actually implement an IF algorithm but to mimic in some sense its behavior. We chose to use the SNOPT [18], which makes available a finite difference support in which either forward or central differences are used adaptively, applying the following formulae:

Forward scheme

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(\hat{x}) - f(x)}{h(1 + |x_i|)} \quad \text{where } \hat{x}_j = \begin{cases} x_j & j \neq i \\ x_i + h(1 + |x_i|) & j = i \end{cases} \tag{1.15}$$

Central scheme

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(\hat{x}^+) - f(\hat{x}^-)}{h(1 + |x_i|)} \quad \text{where } \hat{x}_j^{\pm} = \begin{cases} x_j & j \neq i \\ x_i \pm \frac{1}{2}h(1 + |x_i|) & j = i \end{cases} \tag{1.16}$$

In SNOPT, forward differences are used by default, using a step length parameter h which can be chosen by the user; SNOPT switches from forward to central difference when the current point is close to a stationary point. Moreover, the algorithm reduces the step length to ensure feasibility with respect to the linear and box constraints.

The main advantages of this strategy are the following:

- forward differences require half function evaluations with respect to centered ones;
- step length reduction occurs only to ensure feasibility or close to the solution.

SNOPT with finite difference is not an IF-like algorithm, as the strategy used is that of performing very precise derivative estimation, trying to avoid excessive function evaluations. However, as previously suggested, we can separate the local search in two phases: during the first one we carry on an "imprecise" search by choosing a large step h; during the second phase we refine the result by switching to a much smaller h value. Such two-phase local search resembles the behavior of an IF algorithm.

On the random generation and variable scaling Some choices turned out to be quite beneficial both in terms of effectiveness and of robustness.

Variable scaling According to our experience, confirmed also by other works like, e.g., [2], it is important to scale variables in order to reduce numerical problems: each ESA ACT test problem is composed by variables with widely different scales: variables in fact include angles, velocities, departure dates, and so on. In our method we scaled each variable to the interval $[-0.5, 0.5]$.

Periodic variables Many test problems contain variables representing angles constrained to lie in $[0, 2\pi]$. Clearly we can discard such limitation w.l.o.g. and let variables be free, but, to comply with the previous issue, before any local optimization phase variables are scaled back to the original interval. This way we obtain the following advantages:

- the local optimizer can explore the solution space more freely when close to the former boundary;
- during the perturbation step we have not to deal with feasibility of the perturbed variables.

We performed a large set of numerical experiments with two objectives in mind: first we wanted to obtain good trajectories, i.e. solutions which were comparable or possibly better than those deposited at the ESA ACT web site. Second, we wanted to check which of many possible variations in our algorithm were the most successful and the most robust ones. By this last term we mean that one of our aims has been that of proposing an algorithm which was capable, in many cases, to produce a set of good solutions.

We briefly discuss here our results only for MGADSM problems characterized by the presence of box constraints only. The meaning of each variable is briefly summarized in Table 1.4, while problem characteristics (number of variables and sequence of astronomical bodies visited) are listed in Table 1.5. Note that Tandem is not a single problem but a set of 24 problems, each corresponding to a different sequence of planets from the Earth to Saturn.

Table 1.4: Variables for box constrained ESA MGADSM problems

Name	Meaning	#
t_0	departure time	1
V_∞	dep. vel. modulus	1
u	dep. vel. angle1	1
v	dep. vel. angle2	1
T_i	time of flight	n
η_i	time of DSM i	n
rp_i	pericenter radius at swing-by i	$n-1$
b_i	outc. vel. angle at swing-by i	$n-1$

Problem Name	variables	n	Planet sequence
Cassini	22	5	E V V E J S
Messenger	18	4	E E V V Me
Rosetta	22	5	E E V E E 67P
Tandem	18	4	E P1 P2 P3 S

Table 1.5: Box constrained ESA MGADSM problems. E: Earth, V: Venus J: Jupiter, S: Saturn, Me: Mercury, M: Mars, 67P: Comet 67P/Churyumov-Gerasimenko, Pi: generic planet chosen in the set $\{E, V, M, J\}$

All tests, if not otherwise stated, have been performed with the following stopping criteria:

- Multistart performed 1000 steps, with uniformly generated starting points;
- at each Multistart step, MBH was executed until no improvement was observed in the last $N = 500$ iterations

We report our results using diagrams similar to performance profiles ([14]), using the current performance function:

$$f(t) = \frac{|\{i \in 1..\texttt{NRuns} : f_i \leq f^\star \times (1 + t/100)\}|}{\texttt{NRuns}}$$

$f(t)$ represents the percentage of times (out of 1000 independent trials) in which the algorithm obtained a final value which was within $t\%$ of the currently known putative optimum.

In practice, at $t = 0$ we can read the percentage of times (if any) the method obtained the currently known global minimum, at $t = 100$ the fraction of runs giving a value which is at most twice the optimum, i.e. no more than 100% worse, and so on. This way it is quite easy to read from the figure which algorithm gave the best approximation to the optimum and which was capable of producing a larger quantity of good results. These

graphical representations are closely related to performance profiles, whereas here we compare different independent runs of some algorithms on a single problem, while usual performance profiles show the behavior of single runs of different methods on different test problems.

Results for the Tandem mission

As principal test-bed, we choose the Tandem mission, a box constrained problem with 18 variables. At the ESA ACT web site, 24 instances of box constrained missions to Saturn are proposed, differing on the particular sequence of swings-by performed. In the following figures we will represent computational profiles in particular for what concerns the mission with highest estimate of the global maximum, i.e. mission 6 (starting with Earth, with 3 swings-by at Venus, Earth and Earth again). This problem is formulated as a maximization one, as the objective is a function of the final mass of the aircraft.

The first trials we performed were devoted to understand which kind of perturbation was "optimal" during the execution of MBH. In particular, as there seemed to be some evidence that some variables in the problem like, e.g., starting times and times of flight, were in some sense easier to choose than other ones, or, at least, that, once well chosen, they were quite stable, we tried to check whether perturbations involving only a few variables at a time were more successful than perturbations in which every variable is randomly displaced. We choose the following characteristic parameters for our experiments:

- at each step of MBH, the current solution was perturbed in the following way: for algorithm labeled `MBH1PPertSome` between 1 and 4 coordinates were randomly chosen and uniformly perturbed in an interval of radius equal to 5% of the box containing the variable; for algorithm labeled `MBH1PPertAll` *every* coordinate was uniformly perturbed in an interval whose radius is 5% of the box.
- numerical derivatives were computed using a parameter $h = 10^{-5}$ in formulae (1.15)-(1.16).
- A single phase of local optimization was performed

In Figure 1.13 we report the results obtained running the two versions of this method, with the graphical representation introduced above.

It is quite evident from Figure 1.13 that perturbation of all coordinates is preferable: although both methods find similar global optima, the method based on the perturbation of all variables is significantly more robust (the corresponding curve is significantly "higher" than that of the competing algorithm).

The second set of experiments was aimed at checking the efficiency of our two-phase approach versus the single phase one. During the first-phase we let $h = 10^{-2}$ in the formula for numeric derivatives. When the local optimization method, SNOPT, called for stopping, we started a second optimization with the usual parameter $h = 10^{-5}$.

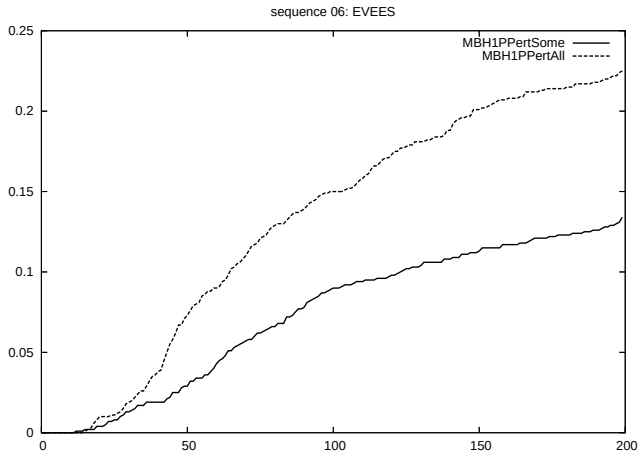


Figure 1.13: Comparison of two different perturbations: all variables (dotted line) vs. just a few ones (solid line).

Apart from this, we maintained the other parameters unchanged. In Figure 1.14 we report the comparison between one- and two-phase optimization.

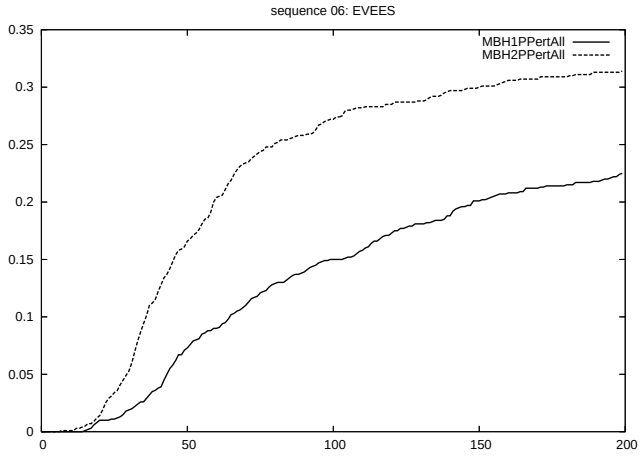


Figure 1.14: Comparison between 1– (solid line) and 2–Phase (dotted line) algorithms

Again it is evident how using two phases is extremely beneficial both in terms of precision and of robustness.

A final experiment was carried on in order to check whether MBH was indeed useful: we counted, for the 1000 experiments made, the total number of two–phase local searches performed, which resulted to be 950 046. We then ran the same number of (two-phase) Multistart iterations and checked the obtained result. In Figure 1.15 we report the comparison between MBH and Multistart; in the figure, for what concerns Multistart, we

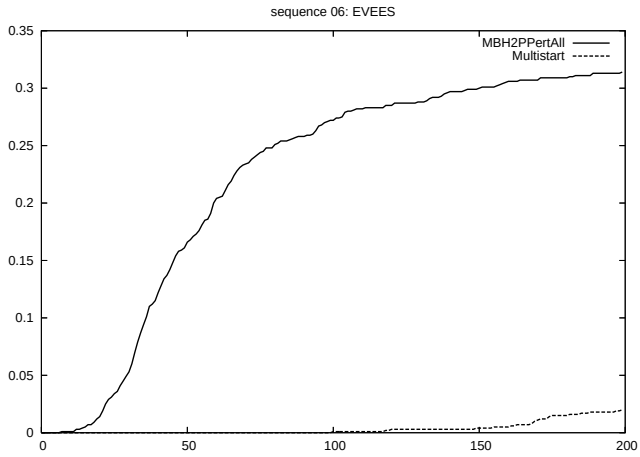


Figure 1.15: Comparison between Multistart and MBH

choose the 1000 best results and compared them with MBH – it is clear that this way the behavior of Multistart is artificially much improved; nonetheless the superiority of MBH is striking. This fact might lead us to conjecture that, similarly to problems in molecular conformation, also space trajectory optimization possesses a "funnel" structure, in which minima are clustered together. The fact that trajectory planning problems might possess a funnel structure apparently was never noticed or conjectured in the literature.

It is worth mentioning that repeating the same kind of analysis on all the 24 tests for the Tandem mission gave roughly the same behavior.

Results for other box-constrained missions

We also ran experiments on the other box-constrained tests proposed by ESA ACT, namely those related to Rosetta, Messenger and Cassini missions. Again, the same kind of algorithm was particularly efficient in all of those tests. It can be easily seen from the ESA ACT web site that, as of the time of writing, we were record holders for all of the box-constrained tests available.

In particular, we consider particularly interesting the fact of having been able, for what concerns the Messenger mission, to discover a competitive solution which corresponds to starting the mission years before the starting date of previously known solutions (as well as of the real space mission).

We would like to remark that most of the novel putative global optima we found are truly new solutions, i.e. they cannot be considered as refinement of previously known ones. As an example, we plot in the following figures a trajectory assumed to be optimal for Rosetta mission on April 2008, with objective function (representing total mission variation in velocity) equal to 1.417km/s, and the one we found (and later improved) in May 2008, with objective 1.3678. Although the variation in the objective is not

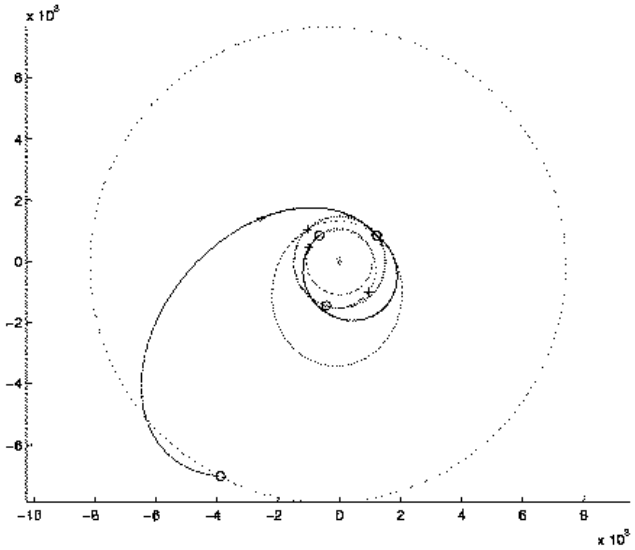


Figure 1.16: Rosetta mission, $\Delta_V = 1.417$

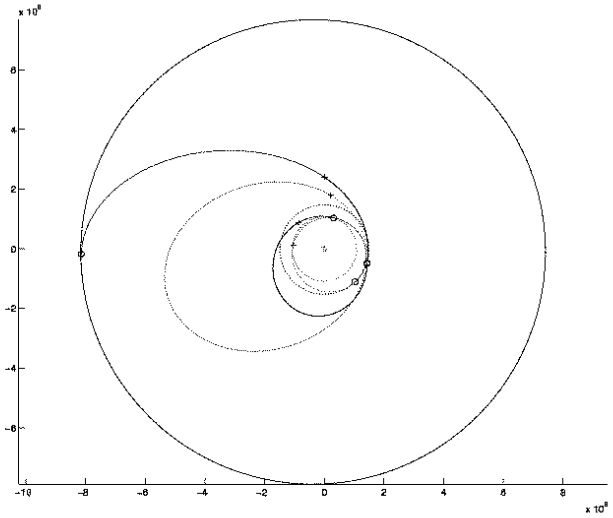


Figure 1.17: Rosetta mission, $\Delta_V = 1.3678$

particularly impressive, the trajectories widely differ.

For more details on the results, the interest reader can refer to the full paper[J1].

Bibliography

[1] O. Abdelkhalik and D. Mortari. N-impulse orbit transfer using genetic algorithms. *Journal of Spacecraft and Rockets*, 44:456–459, 2007.

[2] Astrid Batterman, Joerg M. Gablonsky, Alton Patrick, Carl T. Kelley, Katheleen R. Kavanagh, Todd Coffey, and Cass T. Miller. Solution of a groundwater control problem with implicit filtering. *Optimization and Engineering*, 3(2):189–199, June 2002.

[3] D. V. Boll, J. Donovan, R.L. Graham, and B.D. Lubachevsky. Improving dense packings of equal disks in a square. *The Electronic Journal of Combinatorics*, 7(R46):1–9, 2000.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[5] L. G. Casado, I. Garcia, P. Szabö, and T. Csendes. Equal circles packing in square II: new results for up to 100 circles using the TAMSASS-PECS algorithm. In F. Giannessi, P.M. Pardalos, and T. Rapcsak, editors, *Optimization Theory: Recent developements from Mátraháza*, pages 207–224. Kluwer Academic Publishers, 1998.

[6] Ignacio Castillo, Frank J. Kampas, and Janos D. Pinter. Solving circle packing problems by global optimization: Numerical results and industrial applications. *European Journal of Operational Research*, to appear, 2006.

[7] T. D. Choi and Carl T. Kelley. Superlinear convergence and implicit filtering. *SIAM Journal on Optimization*, 10(4):1149–1162, 2000.

[8] H.G. Croft, K.J. Falconer, and R.K. Guy. *Unsolved problems in Geometry*. Springer-VErlang, New York, 1991.

[9] B. Dachwald. Optimization of interplanetary solar sailcraft trajectories using evolutionary neurocontrol. *AIAA Journal of Guidance, Control, and Dynamics*, 27:66–72, 2004.

[10] C. de Groot, R. Peikert, D. Würtz, and M. Monagan. Packing circles in a square: a review and new results. In *System Modelling and Optimization*, pages 45–54, Zürich, 1991. Proc. 15th IFIP Conf.

[11] P. Di Lizia and G. Radice. Advanced global optimization tools for mission analysis and design. Technical Report 03-4101b, ESA, 2004.

[12] Artan Dimnaku, Rex Kincaid, and Michael W. Trosset. Approximate solutions of continuous dispersion problems. *Annals of Operations Research*, 136(1):65–80, Apr 2005.

[13] L. C. W. Dixon. Global optima without convexity. In H. Greenberg, editor, *Design and Implementation Optimization Software*, pages 449–479. Sijthoff and Noordhoff, Alphen aan den Rijn, 1978.

[14] Elizabeth D. Dolan and Jorge J. Morè. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[15] Aleksandar Donev, Salvatore Torquato, Frank H. Stillinger, and Robert Connelly. A linear programming algorithm to test for jamming in hard-sphere packings. *Journal of Computational Physics*, 197(1):139 – 166, 2004.

[16] Jonathan P. K. Doye, Robert H. Leary, Marco Locatelli, and Fabio Schoen. Global optimization of Morse clusters by potential energy transformations. *INFORMS Journal On Computing*, 16:371–379, 2004.

[17] P.J. Gage, R.D. Braun, and I.M. Kroo. Interplanetary trajectory optimization using a genetic algorithm. *Journal of the Astronautical Sciences*, 43:59–75, 1995.

[18] P. E. Gill, W. Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12:979–1006, 2002.

[19] P. C. Gilmore and Carl T. Kelley. An implicit filtering algorithm for optimization of functions with many local minima. *SIAM Journal on Optimization*, 5(2):269–285, 1995.

[20] R.L. Graham and B.D. Lubachevsky. Repeated patterns of dense packings of equal disks in a square. *The Electronic Journal of Combinatorics*, 3:1–16, 1996.

[21] A. Grosso, A. R. Jamali, M. Locatelli, and F. Schoen. Solving the problem of packing equal and unequal circles in a circular container. *J. of Global Optimization*, 47(1):63–81, May 2010.

[22] Andrea Grosso, Marco Locatelli, and Fabio Schoen. A population-based approach for hard global optimization problems based on dissimilarity measures. *Mathematical Programming*, 110(2):373–404, Jul 2007.

[23] H. Huang, W. Huang, Q. Zhang, and D. Xu. An improved algorithm for the packing of unequal circles within a larger containing circle. *European Journal of Operational Research*, 141:440–453, 2002.

[24] W. Huang, Y. Li, B. Jurkowiak, C.M. Li, and R.C. Xu. A two-level search strategy for packing unequal circles into a circular container. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming*, pages 868–872. Springer-Verlag, 2003.

[25] Dario Izzo. Advances in global optimisation for space trajectory design. In *25th International Symposium on Space Technology and Science*. Japan Society for Aeronautical and Space and ISTS, 2006.

[26] Dario Izzo. 1st ACT global trajectory optimisation competition: Problem description and summary of the results. *Acta Astronautica*, 61(9):731–734, November 2007.

[27] Dario Izzo, V. M. Becerra, D. R. Myatt, S. J. Nasuto, and J. M. Bishop. Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories. *Journal of Global Optimization*, 38:283–296, 2007.

[28] Y.H. Kim and D.B. Spencer. Optimal spacecraft rendezvous using genetic algorithms. *Journal of Spacecraft and Rockets*, 39:859–865, 2002.

[29] Robert H. Leary. Global optimization on funneling landscapes. *Journal of Global Optimization*, 18:367–383, 2000.

[30] Julian Lee, In-Ho Lee, and Jooyoung Lee. Unbiased global optimization of lennard-jones clusters for $n \leq 201$ using the conformational space annealing method. *Phys. Rev. Lett.*, 91:080201, Aug 2003.

[31] M. Locatelli. On the multilevel structure of global optimization problems. *Computational Optimization and Applications*, 30(1):5–22, Jan 2005.

[32] M. Locatelli and F. Schoen. *Global Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2013.

[33] Marco Locatelli. On the multilevel structure of global optimization problems. *Computational Optimization and Applications*, 30(1):5–22, January 2005.

[34] Marco Locatelli and Ulrich Raber. Packing equal circles in a square: a deterministic global optimization approach. *Discrete Applied Mathematics*, 122:139–166, 2002.

[35] Marco Locatelli and Fabio Schoen. Efficient algorithms for large scale global optimization: Lennard-jones clusters. *Computational Optimization and Applications*, 26(2):173–190, Nov 2003.

[36] Mihály Csaba Markót and Tibor Csendes. A new verified optimization technique for the "packing circles in a unit square" problem. *SIAM J. on Optimization*, 16:193–219, 2005.

[37] D. R. Myatt, V. M. Becerra, S. J. Nasuto, and J. M. Bishop. Advanced global optimisation for mission analysis and design. Technical Report 18138/04/NL/MV, ESA, 2004.

[38] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, USA, second edition, 2006.

[39] K.J. Nurmela and P.R.J. Oestergard. Packing up to 50 equal circles in a square. *Discrete Computational Geometry*, 18:111–120, 1997.

[40] K.J. Nurmela and P.R.J. Oestergard. More Optimal Packings of Equal Circles in a Square. *Discrete Computational Geometry*, 22:439–457, 1999.

[41] A. D. Olds, C. A. Kluever, and M.L. Cupples. Interplanetary mission design using differential evolution. *Journal of Spacecraft and Rockets*, 44:1060–1070, 2007.

[42] J.D. Pinter and F.J. Kampas. Nonlinear Optimization in Mathematica using MathOptimizer Professional. *Mathematica in Education and Research*, 10:1–18, 2005.

[43] G. Rauwolf and V. Coverstone-Carroll. Near-optimal low-thrust orbit transfers generated by a genetic algorithm. *Journal of Spacecraft and Rockets*, 33:859–862, 1996.

[44] E. Specht. Packomania. `http://www.packomania.com`. Accessed: 2018-09-06.

[45] Y.G. Stoyan and G.N. Yaskow. Mathematical model and solution method of optimization problem of placement of rectangles and circles taking into account special constraints. *International Transactions in Operational Research*, 5:45–57, 1998.

[46] M. Vasile and M. Locatelli. A hybrid multiagent approach for global trajectory optimization. *Journal of Global Optimization*, 2008. to appear.

[47] M. Vasile, E. Minisci, and M. Locatelli. On testing global optimization algorithms for space trajectory design. In *Proceedings of AIAA 2008*, 2008.

[48] Massimiliano Vasile. A global approach to optimal space trajectory design. *Advances in the Astronautical Sciences*, 114:629–647, February 2003.

[49] Massimiliano Vasile. Design of Earth-Mars transfer trajectories using evolutionary-branching technique. *Acta Astronautica*, 56:705–720, 2005.

[50] Massimiliano Vasile and Paolo De Pascale. Design of Earth-Mars transfer trajectories using evolution-branching technique. In *54th International Astronautical Congress*, October 2003.

[51] Tamás Vinkó, Dario Izzo, and Claudio Bombardelli. Benchmarking different global optimisation techniques for preliminary space trajectory design. In *58th International Astronautical Congress*. International Astronautical Federation (IAF), September 2007.

[52] D. J. Wales and J. P. K. Doye. Global optimization by Basin-Hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *Journal of Physical Chemistry A*, 101(28):5111–5116, 1997.

[53] David J. Wales. Energy landscapes and structure prediction using basin-hopping. In *Modern Methods of Crystal Structure Prediction*, chapter 2, pages 29–54. Wiley-Blackwell, 2010.

[54] D. Zhang and A. Deng. An effective hybrid algorithm for the problem of packing circles into a larger containing circle. *Computers & Operations Research*, 32:1941–1951, 2005.

[55] A. Zimmermann. Circle packing contest. `http://www.recmath.org/contest/CirclePacking/index.php`. Accessed: 2018-09-06.

International Journals

[J1] B. Addis, A. Cassioli, M. Locatelli, and F. Schoen. A global optimization method for design of space trajectories. *Computational Optimization and Applications*, 2009.

[J2] B. Addis, M. Locatelli, and F. Schoen. Efficiently packing unequal disks in a circle. *Operations Research Letters*, 36:37–42, 2008.

[J3] B. Addis, M. Locatelli, and F. Schoen. Disk packing in a square: a new global optimization approach. *INFORMS Journal on Computing*, 20:516–524, 2008.

[J4] B. Addis and S. Leyffer. A trust-region algorithm for global optimization. *Computational Optimization and Applications*, 35:287–304, 2006.

[J5] B. Addis, M. Locatelli, and F. Schoen. Local optima smoothing for global optimization. *Optimization Methods and Software*, 20:417–437, 2005.

Other publications

[O1] B. Addis. *Global optimization using local searches*. PhD thesis, Computer Science and Automation Engineering, 2005.