UNIVERSITÀ DEGLI STUDI DI FIRENZE
Dipartimento di Sistemi e Informatica

Dottorato di Ricerca in
Ingegneria Informatica e dell'Automazione
XVII Ciclo

Global optimization using local searches

Bernardetta Addis

Ph.D. Coordinator
Edoardo Mosca

Advisor
Fabio Schoen

Anno Accademico 2003–2004

Contents

Introduction

Many problems of practical interest can be posed as box constrained smooth non-linear optimization problems. In particular, well-known conformational problems such as protein folding and atomic/molecular cluster problems. In these applications we are interested in finding the minimum energy conformation in three-dimensional space.

In protein folding the primary structure (that is the amino-acids sequence) is given and we want to determine the 3-dimensional structure that minimize the energy. The shape of proteins is of fundamental importance in determining their activity in living bodies, in fact their capability to interact with other substances is strictly related to that. From the genome mapping our knowledge about proteins and their activity is widely increased, but it is even more important from the primary structure to determine the shape of proteins by means of automatic procedures. A similar problem, known as cluster conformation, is interesting in modeling different kinds of material from noble gases to fullerene (C_{60}), a basic component for nano-tubes, a material that seems to be very interesting in nano-technologies.

In a large number of conformational problems, the energy landscape is characterized by a huge number of local, not global, minima. For this reason general purpose methods are doomed to fail in finding the solution of the problem.

There is an apparent contradiction between what happens in "Nature"

and in simulations. In fact, despite of the huge number of local optima, in real processes the global optimum conformation is found quite fast (in protein folding this is called Levinthal's paradox). In some sense "Nature" is able to move along a path jumping from one minima to another and reaching easily the global optimum. It is widely believed that this depends mainly on the particular landscape of the energy. In fact, it frequently happens that local optima are not randomly displaced, but quite often most of the local optima might be seen as small amplitude perturbations of a smooth function which possesses only few, easily reachable, local optima. If this is the case the energy is said to have a funnel structure. This concept was first introduced in protein folding, and than used also in other fields. The general idea of funnel and a description of some conformational problems is presented in Chapter 2.

A way to efficiently solve these problems is to develop algorithms which take into account the special structure of the function. Methods that use local searches are considered particularly successful in many cases. Local searches reduce the oscillation of the function and get rid of barriers between one minimum and lower adjacent minima. Monotonic Basin Hopping is an example of a quite efficient strategy for conformation problems that uses local searches. A brief introduction to global optimization and some methods are reported in Chapter 1.

We focus in particular on protein docking: given two molecules, we search the structure of the complex that minimize the energy. This conformation is called the docking configuration and its knowledge is important in order to understand the interaction between the two proteins. In first approximation proteins can be considered as rigid bodies, so the method we proposed is aimed to solve rigid protein docking. To solve this problem as an optimization one, we introduce the penalized energy strategy, that has been very successful in atomic cluster conformations, in the Monotonic Basin Hopping. The method consists of a double local search, the first one is aimed to obtain a close match between the two interacting molecules by means of the inclusion of a penalty term in the original energy. The second is necessary to obtain a minimum point of the real objective function. The overall result

of the method is to increase the basin of attraction of the global minimum. The method and the preliminary results are reported in Chapter 3.

Furthermore we propose a new framework for global exploration which tries to guide random search towards the region of attraction of low-level local optima, exploiting explicitly the funnel structure of the function. The main idea originated by the use of smoothing techniques. We apply a smoothing transformation not to the objective function, but to the result of local searches. We propose a computational approximation scheme to evaluate the smoothing and a technique to optimize it. In Chapter 4 the problem is introduced, and in Chapter 5 two methods for optimizing the smoothed function are proposed. We test the methods on several test functions: they result to be very efficient and robust in solving difficult global optimization problems.

Chapter 1

Global Optimization

A global optimization problem, for our purpose, can be defined as

$$\begin{cases} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & x \in S \subset \mathbb{R}^n, \end{cases} \tag{1.1}$$

where f is sufficiently smooth and $S \subset \mathbb{R}^n$ is a compact set which (combined with the continuity or lower semicontinuity of f) guarantees the existence of the minimum value $f^\star$. For the sake of simplicity we consider the case in which all local minimima, and so the global, are attained in the interior of S. Let

$$S^\star = \{x \in S : f(x) = f^\star\}$$

be the set of the solution points of problem 1.1. Under the above hypothesis $S^\star$ is always not empty.

The global optimization problem as stated before, is inherently unsolvable in a finite number of steps. This can be verified as follows (see [16]). Consider a continuously differentiable function f. For any point $\bar{x}$ in the feasible set S and any neighborhood B of $\bar{x}$, we can construct another function

$\tilde{f}$ with domain S such that

$$\tilde{f} = f \quad \forall x \in S \setminus B$$

and

$$\bar{x} = \underset{x \in S}{\operatorname{argmin}} \tilde{f}(x)$$

Thus we cannot guarantee that the point $\bar{x}$ is not the global optimum without evaluating the function in every neighborhood of the point itself. This means that there is no algorithm able to find the solution of problem 1.1, that is finding, in a finite number of steps, a point and an "optimality certificate" (i.e. a proof that the point is a global optimum). Only a complete exploration of the feasible set can guarantee to find the global optimum. This is related to the particular structure of the global optimization problem and not to the method used for solving the problem itself. We can reduce our expectations and consider the problem solved if, for some $\epsilon > 0$, we find a point in the level set

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

or in

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

where $d(x, A)$ is a suitable distance between a point x and a set A, for example

$$d(x, A) = \min_{y \in A} \|y - x\|$$

Even finding a point in S_ϵ, or $B_\epsilon(S^\star)$ is still a difficult problem.

In many applications, such as some design problems, finding a good candidate for global optimality can be enough. In other fields, even if theoretically the global optimum cannot be found, there exist methods that can solve a large number of real instances. In such cases we can reduce our request to find "very good" local optima. That is as good as possible in a given amount of time. To this aim it is really important to guarantee a good exploration of the feasible set. It is widely believed that in order to

solve large scale global optimization problems an appropriate mixture of local approximation and global exploration is necessary. Local approximation, if first order information on the objective function is available, is efficiently performed by means of local optimization methods. Unfortunately, global exploration, in absence of some kind of global information on the problem, is a "blind" procedure, aimed at placing observations as evenly as possible in the search domain.

1.1 Local optimization

As we already observed, if the problem allows the use of a sufficiently efficient local optimization algorithm, a two-phase procedure is a good candidate for global optimization [42]. Such a procedure involves sampling coupled with local searches started from some of the sampled points. We define the local minimization operator as

$$L(x) := \begin{cases} \underset{y}{\text{locmin}}(x) & f(y) \\ \text{subject to} & y \in S. \end{cases} \tag{1.2}$$

where locmin(x) represent a local minimization using x as starting point. We note that this operator is implicitly defined and in practice depends on the local minimizer used. In general, $L(x)$ is a piecewise constant function whose pieces correspond to the basins of attraction of the local minima of $f(x)$. It is not obvious how to define basins of attraction in optimization. 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. More formally, we can say that the basin of attraction of a minimum $\bar{x}$, $\mathcal{B}_{\bar{x}}$ is the union of the set $\{\bar{x}\}$ and the set of all points x such that:

$$\exists \quad \alpha : [0,1] \to \mathbb{R}^n \quad \text{continuos}$$

s.t.

$$\begin{cases} \alpha(0) & = x \\ \alpha(1) & = \bar{x} \\ f(\alpha(t')) & < f(\alpha(t)) \quad \forall \quad t < t' \end{cases}$$

In a large number of cases this definition can be considered coherent with the common idea of basin of attraction that usually is related to the behavior of an ideal minimizer. We could expect that starting from a point $x \in \mathcal{B}_{\bar{x}}$, a local minimizer, at least ideally, could reach the corresponding minimum point $\bar{x}$. There are several exceptions and some correction that can mend the definition. First of all we can observe that maximum points can be in more than one basin of attraction. To overcome this we can distinguish between simple and proper basin of attraction. Considering the definition describing simple basin, the largest subset of $\mathcal{B}_{\bar{x}}$ without intersection with other basins, is the proper basin of $\bar{x}$. In this case our definition seems to be coherent with the behavior, for example of an ideal gradient method. In fact we can consider to move from any point in a proper basin of attraction $\mathcal{B}_{\bar{x}}$ to $\bar{x}$ decreasing the objective function.

In some cases our definition cannot be considered completely satisfactory, or at least a little bit ambiguous. For example due to the presence of no strict minima. In this case the corresponding minimum value is attained in a region and not in a single point. Using our definition only boundary points of that region would have a basin of attraction different from themselves (due to the fact that we impose the path being strictly decreasing). It is possible to correct the definition to handle this case at the cost of a more complicated notation; but maybe it is not strictly necessary. Even considering the case of no strict minima, the definition is still coherent with an ideal gradient method, in fact in this case we can only reach points on the frontier of the minimum region. In fact the method is supposed to stop when the gradient is zero.

Let us consider that L is the result of an ideal process of minimization. Clearly, the global optimization problem (1.1) has the same optimal objective value as the following problem:

$$\begin{cases} \underset{x}{\text{minimize}} & L(x) \\ \text{subject to} & x \in S. \end{cases} \tag{1.3}$$

We note that the piecewise constant nature of $L(x)$ implies that the minima

of (1.1) and (1.3) need not agree. In fact, any global minimum of (1.1) is also a global minimum of (1.3), but not vice versa. However, because $L(x)$ is implicitly defined by a local minimizer, we can simply record

$$x_{min} := \mathcal{LS}(x) := \begin{cases} \arg\underset{y}{\operatorname{locmin}}(x) & f(y) \\ \text{subject to} & y \in S. \end{cases} \tag{1.4}$$

It follows that x_{min} is also a local minimum point of $f(x)$, and we can recover a global minimum of $f(x)$ by solving (1.3) taking care of recording the minimum point resulting from the process of evaluating L.

1.2 Stochastic methods

As already observed global exploration without any form of information cannot be done efficiently. Before introducing methods aimed to overcome these difficulties, at least for some family of problems, we briefly describe a few stochastic global optimization methods. The aim of this chapter is not to give an overview of general methods, but just to present the general methods most used in conformational problems, such as protein docking or clusters conformation.

The simplest stochastic method that can be realized is Pure Random Search. At every step we generate a point x, and we record the function value at this point. The generation is made using an uniform distribution on the feasible set S. We repeat this procedure keeping the best point in the overall process. In Algorithm 1 the method is presented in a schematic way. The parameter K represents the total number of steps. The result is the point $x^\star$, the one with lowest value of the visited points.

The method is as simple as inefficient. The information collected during previous steps is not used in the current one. At each step k we perform a uniform random search on the feasible set, independently from values and positions of the previous sampled points. For this reason the performance of the method is more or less unrelated to the structure of the function. For example if we consider the problem solved when we found a point in S_ϵ,

Data : K
$k = 0$
$x^\star = x_0 =$ random uniform point in S
while $k < K$ **do**
  $x_{k+1} =$ random uniform point in S
  if $f(x_{k+1}) < f(x^\star)$ **then**
    $x^\star = x_k$
  $k = k + 1$

Algorithm 1: Pure Random Search

then the average number of steps for solving the problem is related only to the volume of S_ϵ. More precisely, calling V_S the volume of the set S, it is proportional to $\frac{V_S}{V_{S_\epsilon}}$ (see Figure 1.1).

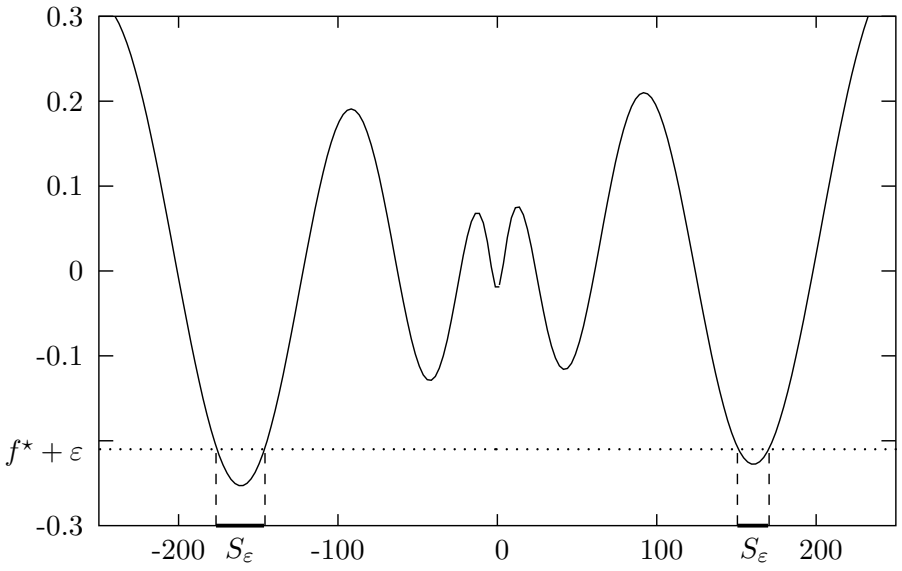

Figure 1.1. Pure Random Search

Indeed the method is interesting, as it can be considered the base of a large number of algorithms. Each of them introduces one or more improvements into this naive strategy. An example of such a procedure is the family of Simulated Annealing methods.

1.2.1 Simulated Annealing

Simulated Annealing (SA in what follows) algorithms are based on an analogy with a physical phenomenon: the behavior of molecules in a liquid metal during solidification. At high temperatures the molecules move freely; if the temperature is slowly decreased, the thermal mobility of the molecules is gradually lost and they can form a pure crystal. If the temperature is decreased too quickly a liquid metal rather ends up in a polycrystalline or amorphous state with a higher energy. In [38] a Metropolis Monte Carlo method was proposed to simulate the physical annealing process. In [49] and [26] the analogies between the configurations of a physical system and the feasible points in an optimization problem, and between the potential energy and the objective function of the optimization problem, lead to the definition of SA algorithms for the solution of combinatorial optimization problems. The approach has been later extended to continuous global optimization problems.

We briefly describe the general structure of these methods. SA algorithms randomly generate at each iteration a candidate point in a suitable neighborhood of the current point and, through a random mechanism controlled by a parameter called temperature (in view of the analogy with the physical process), they decide whether to move to the candidate point or to stay in the current one at the next iteration. There exist a large number of variant of this method for continuous global optimization, most of them can be represented as in Algorithm 2.

The parameter t_k represent the temperature at iteration k, the set Z contains all the information collected up to the current iteration, i.e. all the points observed up to this iteration. The distribution $D(\cdot)$, that depends on set Z, i.e. the past "story", and the acceptance function $A(\cdot)$, that has values in $[0, 1]$, are used to determine the new point x_{k+1}. The next candidate point is generated using D and we decide whether to accept with probability A. The temperature is changed using a function $U(\cdot)$ with nonnegative values usually called cooling schedule.

In the description some parts have been left unspecified, appropriate choices of them define a particular SA algorithm and are essential in order

Data : T
$k = 0$, $t_k = T$
$x^\star = x_0 =$ random uniform point in S
$Z = \{x_0\}$
while *stop condition not fulfilled* **do**
  $y_{k+1} =$ sample with probability distribution $D(Z)$
  $p =$ random uniform point in $[0, 1]$
  if $p \leq A(x_k, y_{k+1}, t_k)$ **then**
    $x_{k+1} = y_{k+1}$
    if $f(x_{k+1}) < f(x^\star)$ **then**
      $x^\star = x_{k+1}$
  else
    $x_{k+1} = x_k$
  Add y_{k+1} to Z
  $t_{k+1} = U(Z)$
  $k = k + 1$

Algorithm 2: General Simulated Annealing

to guarantee its efficiency. For a detailed description of the different choices and the theoretical issues of convergence the reader can refer to the survey [34].

In Algorithm 3 we describe more in detail a very basic version. The next candidate point is generated in a ball of given radius Δ with center the current point x_k. The acceptance function is the Metropolis criterion ([26]), that guarantees all improvement steps are accepted, and an acceptance for not improving steps related to the temperature and the worsening in the objective with respect to the current point. In particular, considering the same difference in the objective function $\delta = f(y) - f(x)$ the probability of acceptance is larger for higher temperature values. The method has a decreasing probability to accept ascending steps with the decrease of the temperature.

Data : Δ, T
$k = 0$, $t_k = T$
$x^\star = x_0 =$ random uniform point in S
while *stop condition not fulfilled* **do**
  $y_{k+1} =$ uniform sample in $B(x_k, \Delta)$
  $p =$ random uniform point in $[0, 1]$
  $A = \min\{1, exp(-\frac{f(y_{k+1})-f(x_k)}{t_k})\}$
  if $p \leq A$ **then**
    $x_{k+1} = y_{k+1}$
    if $f(x_{k+1}) < f(x^\star)$ **then**
      $x^\star = x_{k+1}$
  else
    $x_{k+1} = x_k$
  $t_{k+1} = U(Z)$
  $k = k + 1$

Algorithm 3: Simple Simulated Annealing

1.2.2 Multistart

Other methods improve the PRS strategy dividing explicitly the global search from the local one. A very basic example of this is the Multistart method ([28], [50]). It consists in uniform random sampling of the space and performing local search from the sample points. The method is described in Algorithm 4 .

Data : K
$k = 0$
$x =$ random uniform point in S
$x^\star = x_0 =$ local minimization starting from x
while $k < K$ **do**
  $x =$ random uniform point in S
  $x_k =$ local minimization starting from x
  if $f(x_k) < f(x^\star)$ **then**
    $x^\star = x_k$
  $k = k + 1$

Algorithm 4: Multistart

It is very similar to PRS, but here we do not compare function values of the samples, but local optima that we can reach from them. The result is that the solution is always at least a local optimum and, maybe more interesting, the average number of steps necessary to find the solution is related to the volume of the basin of attraction of the global optimum. In particular if we consider again to search a point in S_ϵ, the average number of steps necessary to solve the problem is proportional to $\frac{V_S}{V_{A_\epsilon}}$, where A_ϵ is the union of the basin of attraction of the minima in S_ϵ (see Figure 1.2).

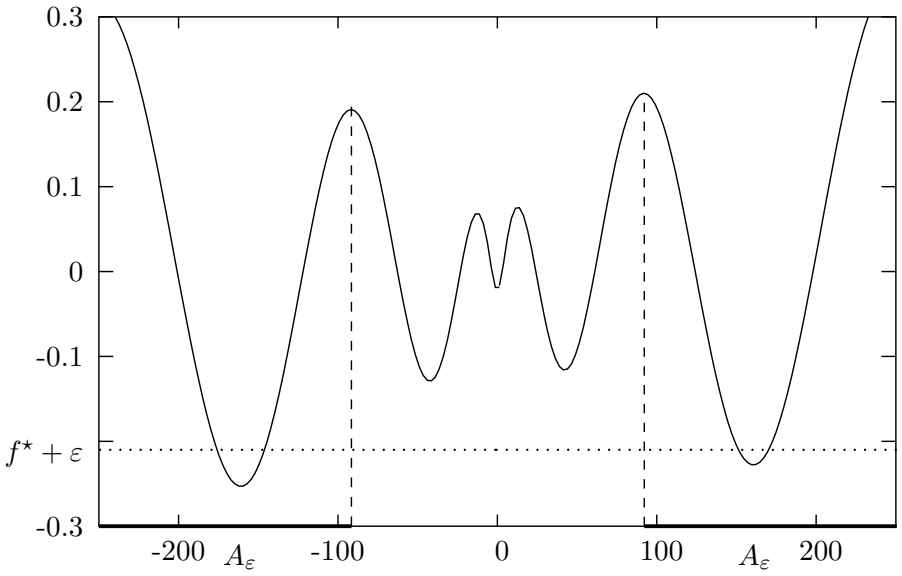

Figure 1.2. Multistart

Several optimization algorithms can be considered as different methods aimed at minimizing $L(x)$, instead of $f(x)$. In this view Multistart reduces to Pure Random Search applied to $L(x)$. In Algorithm 5 the method is stated using the notation introduced in 1.1

Another example is the Monotonic Basin Hopping algorithm.

Data : K
$k = 0$
x = random uniform point in S
$x^\star = x_0 = \mathcal{LS}(x)$
while $k < K$ **do**
  x = random uniform point in S
  if $L(x) < L(x^\star)$ **then**
    $x^\star = \mathcal{LS}(x)$
  $k = k + 1$

Algorithm 5: Pure Random Search on $L()$ - Multistart

1.2.3 Monotonic Basin Hopping

Monotonic basin hopping ([28]) consists of repeatedly performing local optimizations starting from points randomly generated in the neighborhood of the current one: as soon as a local optimization ends up in a local minimum whose value is better than the current one, this local minimum becomes the current point.

It can be considered as the result of applying to $L(x)$ the simulated annealing procedure we previously described, with temperature constantly equal to zero.

Data : Δ, N
$n = 0$, $k = 0$
y = random uniform point in S
x_0=$x^\star = \mathcal{LS}(y)$
while $n < N$ **do**
  y_k = random uniform point in $B(x_k, \Delta)$
  if $L(y_k) < L(x^\star)$ **then**
    $n = 0$
    $x^\star = x_{k+1} = \mathcal{LS}(y_k)$
  else
    $n = n + 1$
  $k = k + 1$

Algorithm 6: Monotonic Basin Hopping

At each step a better local optimum is selected in the neighborhood of the current one, if it is possible. In particular usually the stopping criterion is based on the number of unsuccessful steps. We consider a step successful, if the point y_k improves the function $L()$ with respect to the current point x_k; we stop after a given number of consecutive unsuccessful steps. The radius parameter Δ is critical for the efficiency of the method. If it is too large the method reduces to Multistart, if it is too small, no better point might be found in $B(x_k, \Delta)$. In some problems a good value can be found with a relatively small number of attempts, using some knowledge on the problem. Another possibility is to adaptively change the radius.

1.2.3.1 An adaptive version

To our knowledge the only adaptive version of MBH is proposed in [35]. We describe the strategy for updating the radius using the same notation used in the description of MBH method (see Algorithm 6). Let $\bar{N}$ be a positive integer, $l \in (0, 1)$ and set the radius $\tilde{\Delta} = \Delta$. Then, every $\bar{N}$ steps, evaluate the fraction p of iterations for which $z_k \neq x_k$ (that is the candidate point is different from the center of sampling ball), and update $\tilde{\Delta}$ using the following strategy:

$$\text{if} \quad p = 1.0 \begin{cases} \tilde{\Delta} > \Delta, & \text{then} \quad \tilde{\Delta} = \tilde{\Delta} - \Delta \\ \tilde{\Delta} \leq \Delta, & \text{then} \quad \tilde{\Delta} = \tilde{\Delta}/2 \end{cases}$$

$$\text{if} \quad p < 1.0 \begin{cases} \tilde{\Delta} \geq \Delta, & \text{then} \quad \tilde{\Delta} = \tilde{\Delta} + \Delta \\ \tilde{\Delta} < \Delta, & \text{then} \quad \tilde{\Delta} = 2\tilde{\Delta}. \end{cases}$$

We refer to the method resulting from the introduction of this radius update as AMBH (Adaptive MBH).

Chapter 2

Local searches and funnel functions

2.1 Funnel-like functions

In many molecular conformation problems [19], the number of local optima is huge. However it frequently happens that local optima are not randomly displaced, but quite often their arrangement is such that reaching the global optimum is, is some sense, easier – in biology these kind of functions are referred to as "funnel"-like structures. By this they mean that quite often most of the local optima might be seen as small amplitude perturbations of a smooth function which possesses only few, easily reachable, local optima, i.e. local optima whose basin of attraction is significantly large (see Figure 2.1, the dotted line represents the underlying funnel structure).

A formal definition of funnel does not exist. This is a relatively new concept appeared in molecular biology to describe the energy landscape of protein folding. This idea was introduced to solve the so-called Lenvinthal's paradox. To clarify the idea of funnel we shortly describe this problem. It

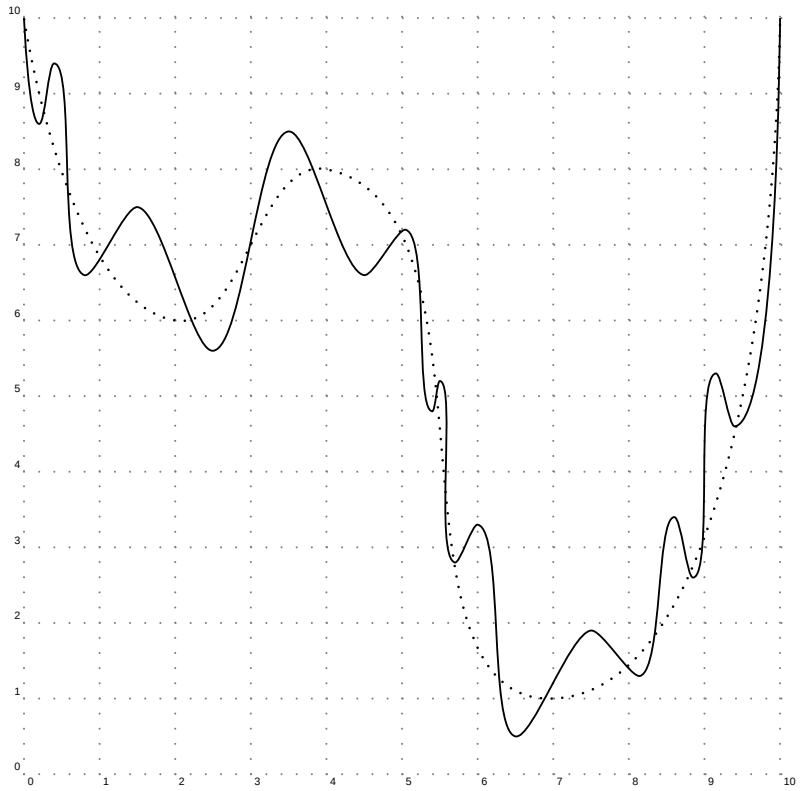

Figure 2.1. Example of funnel function

is widely believed that the 3-dimensional structure of proteins is determined uniquely by the primary structure, that is the sequence of amino-acids. Levinthal noticed (see [31]) that there seems to be a contradiction in the protein folding process. His argument starts by noticing that the number of possible conformations in a protein scales exponentially with the number of its amino-acids. Considering that this number is of the order of hundreds (in some cases even thousands), we obtain a huge number of conformations. If we consider that for exploring each conformation we need at least a picosecond (10^{-20} s), we can understand that the time necessary to fold a protein is of the same order of the universe events. That is equivalent to say that

folding cannot ever happen.

Of course there is something wrong in the argument. Proteins fold and a large part of them very quickly, the time scale is of the order of minutes, generally seconds. The weak point in Levinthal's argument (see [11]) is the assumption that all conformation are equally likely in the path from the unfolded to the folded states. On the contrary conformation with a lower free energy are more likely than those that with higher free energy. Protein folding is a dynamic process where the Boltzmann factor (probability of a conformation is proportional to its energy) has to be take into account. Furthermore a particular shape of the energy has to be supposed. It is widely believed that the surface of the energy landscape helps in guiding the protein to the native state, that is the folded one. In this case biologists talk of funnel structure energy. In particular they mean that even if the surface may have bumps and wiggles the energy generally decreases as we move to the native structure.

This concept was extended in other fields that can be considered, in some sense, related to protein folding, such as protein-protein docking and atomic cluster conformation. Also in these problem something similar to proteins folding happens. The number of possible conformation is huge, but the process happens quite fast, so something similar to Levinthal's paradox can be stated, and a similar structure for the energy can be supposed (see [19], [48]).

Let us consider how conformational problems can be solved using optimization. Usually, the potential energy of the system is considered as the objective function and the variables are the degrees of freedom of the system itself, that is the possible conformation of the amino-acids in protein in folding, the mutual position of two molecules in docking, or the coordinates of atoms in clusters. In particular the possible conformations correspond to local minimum points of the energy (the other points are just transient states). As we already pointed out, there does not exist a formal definition for funnels. The concept is related to dynamics and properties of the free energy of the system, for this reason it is strictly problem dependent. Still when funnel-like problems are resolved using optimization some common

properties can be observed, in particular in the shape of the objective function. We try to characterize (similarly to [28]) what we mean for funnel-like functions in this work, without pretence of covering all the real cases.

Suppose to construct an oriented graph, where the set of vertices corresponds to the set of local minimum points (for the sake of simplicity consider that only strict local minima exist). Two vertices x_i, x_j are connected by an arc (x_i, x_j) if the following conditions hold:

$$\begin{cases} f(x_i) & \geq f(x_j) \\ x_j & \text{is reacheable from} \quad x_i \end{cases}$$

where f is the objective function, and we use the same name for minimum points and associated vertices. The bottom of the funnel is a vertex x with only incoming arcs, that is there does not exist an ascending path from the local minimum x.

To construct the graph we need a criterion to decide whether a point is reachable from another one or it is not. In dynamics we can move from one conformation to another if we do not have to increase the energy, that is if the barrier between the two conformations can be overcome by the thermal energy. In this perspective simulated annealing, allowing increase of the energy, can be seen as a simulation of the dynamic process. The objective function does not take into account thermal energy, so the algorithm has to consider the effect of this parameter in some way.

If we consider to move in the space of conformations, we can use as objective function the original one transformed by the process of local searches (that we described in Section 1.1). Using $L(\cdot)$ the barriers between minima disappear, so a different criterion can be used to determine the possibility to reach a minimum. We already observed that MBH can be considered a simplified version of SA applied to L. In the perspective of conformational problems SA increasing steps are generally made with the purpose of moving from one minimum point to better one. Considering this, MBH is more effective than SA in jumping from a minimum to a closer one, because only moving from a minimum to a better one is allowed. Considering also that this method appears to be successful in several conformational problems, it

seems reasonable to consider two local minima connected if it is possible to move from one to the other with a MBH step. That is x_j is reachable by x_i if there exists Δ s.t. $B(x_i, \Delta)$ intersects the basin of attraction of x_j. Of course this cannot considered a formal definition because it depends on the parameter Δ, even though in conformational problems it is sensible to think that the "amount of change" for moving from a conformation to another is bounded.

If the landscape is such that there exist more than one funnel bottom, than we talk about multi-funnel functions. In this case, even more than for the single funnel, the amplitude of the basin of the funnel is relevant to evaluate the probability of reaching the bottom of the funnel itself. We can consider the basin of attraction of a funnel (with bottom x) as the union of the basin of attraction of all minima y_i such that a path from y_i to x exists. We now shortly describe two conformational problems, and a possible strategy using local searches to exploit some global information to drive the search towards the global optimum. In Chapters 4 and 5 we introduce methodologies aimed to exploit directly the funnel structure.

2.2 Cluster conformation

There are several models to describe cluster conformation (see [51]). In particular we restrict our description to cluster with only pair interactions depending on the distance. In this case the atoms can be represented as points in $\mathbb{R}^3$. Considering a cluster of n atoms, and $x \in \mathbb{R}^{3n}$ the vector containing their coordinates, these potentials have the following form:

$$E(x) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} v(r_{ij})$$

where v is the contribution to the energy of a pair of atoms and r_{ij} the distance between atoms i and j. The searched conformation corresponds to the global optimum of $E(x)$. Depending on the shape of v, the energy provides a reasonable description for different inter-atomic interactions. For

example, the Lennard-Jones potential:

$$v(r) = \frac{1}{r^{12}} - \frac{2}{r^6}$$

is suitable for rare gases (see Picture 2.2). A more flexible model is repre-

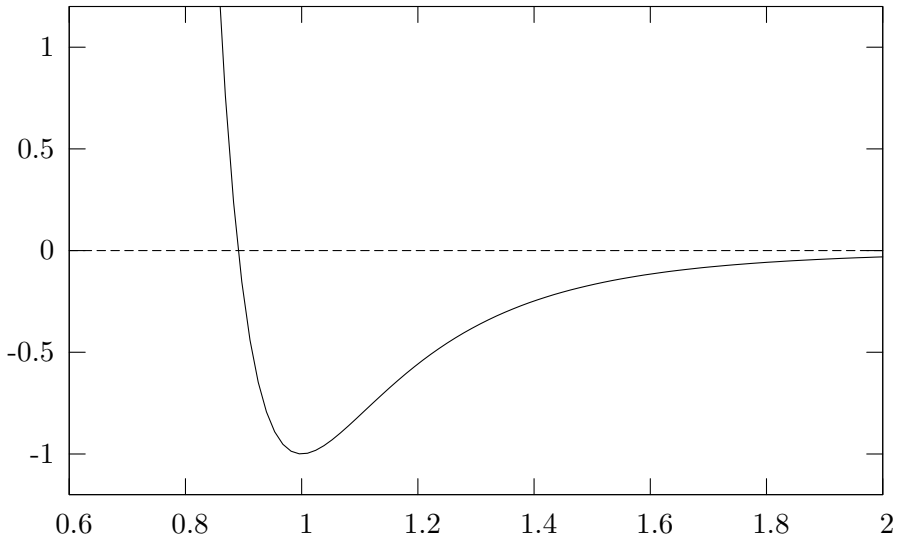

Figure 2.2. Lennard-Jones pair-wise contribution

sented by the Morse potential:

$$v(r) = e^{\rho(1-r)}(e^{\rho(1-r)} - 2).$$

Different values of ρ are appropriated to different materials. For $\rho = 6$ the Morse potential has the same curvature at the bottom of the well of the Lennard-Jones potential(see Picture 2.2). Despite a very simple model, the problem of finding the global optimum is considered very difficult, in fact the number of local minima is supposed to increase exponentially with the number of atoms([45]). For this reason these problems are considered very significant test for global optimization algorithms.

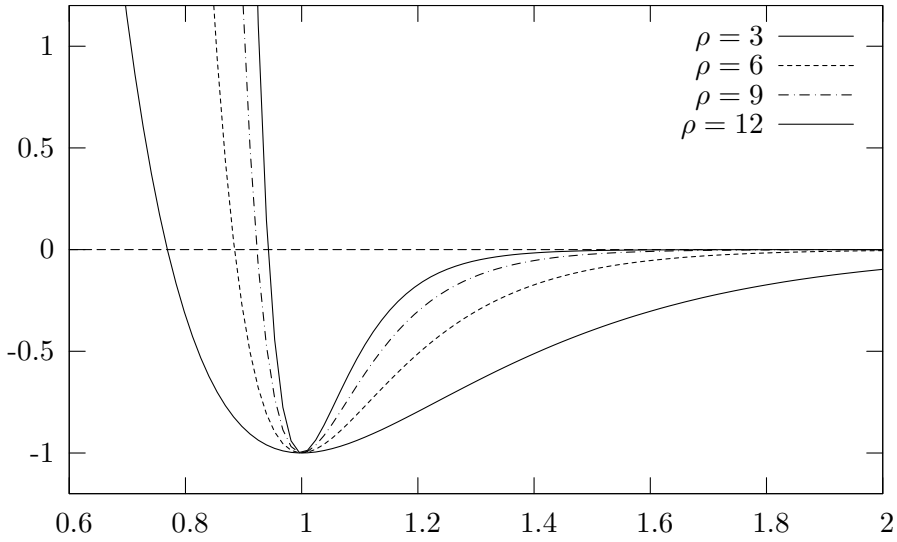

Figure 2.3. Morse pair-wise contribution for different values of ρ

2.3 Protein docking

The term docking is frequently used in the scientific literature to denote the process by which two (or more) different molecules interact in such a way as to form a single complex. In particular it is used when no chemical bond is formed between the molecules, just weak interactions are supposed. Being able to predict the structure of the complex resulting from a docking process is extremely relevant: knowing how (and where) a small ligand molecule docks to a protein might be useful in order to predict the effect that a new drug could produce in a living body.

For average problem sizes, on the order of tens of thousands of possible ligands can be docked to a large biomolecule in order to screen out those which do not appear to fit well or those whose docking site is different from the desired one, thus eliminating the highly expensive and time-consuming process of actually producing and experimenting with large sets of ligands. Most of the literature on docking methods deals with ligand-protein docking, or, in general, with the problem of docking two molecules, one of which has

a very modest size. Several software packages already exist for this purpose and it can be safely assumed that the problem of docking small ligands can in many instances be solved with relatively low computational effort.

The real challenge is now protein-protein docking or, in general, the problem of docking two (or more) large biomolecules, and in fact in the last years more literature has been devoted to this topic.

2.4 Two-phase method

For cluster conformation problems the two-phase strategy proposed in [36],[37] have been very successful, in particular in finding the global optimum in the "difficult cases" (such as $n = 38$ or $n = 75$ for Lennard-Jones clusters). In these cases the energy landscape is a multi-funnel one, and the funnel corresponding to the global minimum has a smaller basin of attraction (see [19]).

The general idea of the method is to change the standard local optimization with a two steps one. This search starts with a local optimization on a function which rewards or penalizes some meaningful characteristics for the global optimum conformation (we can refer to this function as transformed, $Tf(\cdot)$). Then from the local optimum found,a standard local optimization is started. The procedure is described in the following:

$$
\begin{aligned}
&z = \mathcal{LS}_{Mf}(x)\\
&y = \mathcal{LS}_{f}(z)\\
&\text{Set result} = y
\end{aligned}
$$

where we used the formalism introduced in Section 1.1, with $\mathcal{LS}_g$, the local procedure considering g as objective function.

The form and structure of the function used in the first phase is problem-dependent, the aim of the procedure is to increase the region of attraction of the global minimum. In other words the first phase is aimed at producing good starting points for local optimization. The two-phase local search can introduced in different global optimization methods that use standard local

searches. If we consider, for example, Multistart, we can consider the two-phase procedure as a way to change uniform sampling in such a way to take into account some knowledge of the problem. In [18] and, more in detail, in [20] the relationship between the energy landscape of clusters conformation problems and the transformed energy are described. In Chapter 3 we present a method using this strategy for the protein docking problem.

Chapter 3

Two-phase method for protein docking

In this chapter we propose a two-phase strategy (see Section 2.4) for the docking problem. Before introducing the problem and the our method it seems necessary to introduce the main difficulties involved in the use of optimization in this field.

3.1 A few remarks on the protein docking problem

Several kinds of difficulties arise in this context. First of all there are many modelling difficulties. In fact what we know of proteins is quite different from what proteins in nature really are: there are huge databases of protein structures, but these are related to analysis mostly made on crystallized proteins or they are based on NMR observations. These structures are often re-engineered and often lack some important information; for example, crystallographic data report only "heavy atoms", i.e., they do not report

coordinates for hydrogen atoms, which account for roughly one half of the total number of atoms of a protein. So, in order to use some of the most common models for the evaluation of interactions, hydrogen atoms have to be artificially added into the structure and placed in "reasonable" positions, before attempting any docking process.

Also, and more important from the point of view of being able to check the validity of results, there is the problem of modelling the interactions. Generally it is assumed that, in some sense, "Nature" optimizes the free energy, and that the actual docking conformation is the global minimum (or the best minimum "reachable" under the dynamic). The problem of finding an accurate and manageable model of energy function to be optimized remains. Many force fields have been defined in the literature from which an energy function can be evaluated and optimized. However, the best we can hope to obtain, is to find a docking conformation which minimizes the energy function used; the resulting docking should then be compared with the actual docked complex (if this has been observed and inserted in the database).

The choice of a force field produces other kinds of difficulties. As an example, the Amber [7] force field was chosen mainly because it is a sort of standard in protein energy calculations and because its definition is in the public domain. However using the data available and no deep knowledge on the field it is possible to manage only molecules composed of the standard amino-acids. While it is known that different proteins are composed of amino-acids (in the living bodies there exist only 20 different types), whose order is defined by the protein sequence, in most cases proteins which have been observed and recorded in public databases (often as a consequence of the crystallization process) contain also so-called hetero-atoms, atoms which do not strictly belong to any specific amino-acid. The presence of these atoms rules out the possibility, for a non-biologist, to use standard force fields or, seen from another point of view, restricts their applicability to a very limited subset of proteins. This makes extensive numerical testing and validation quite difficult without the support of a biologist.

Given a force field and two biomolecules which we would like to dock, a

global optimization method can in principle be used; it is however important to observe that in a naive approach in which all of the atoms in both molecules are free to change their relative positions, the number of variables becomes huge for most cases. Proteins usually possess a few thousands of atoms, and it is easy to obtain global optimization problems with tens of thousands of variables. Moreover, energy function evaluation is extremely costly, as it generally involves at least pair-wise contributions, so that if the number of atoms is in the thousands, the number of pairwise interactions to be evaluated at each energy function calculation is in the order of millions of operations. The situation is dramatically worse when we include three- and four-body interactions in the energy models, or when we wish to use gradient or higher order information.

Moreover, from the point of view of global optimization, docking problems are characterized by a huge number of local optima which are not global; there are no theoretical results available on the complexity of docking, but it seems reasonable to estimate that the number of locally optimal docking conformations grows at least exponentially in the number of atoms. This conjecture obviously rules out any method which tries to obtain a certified global optimum: the best one can hope to obtain is a good local optimum, with no guarantee of global optimality.

A final difficulty in tuning global optimization methods and in assessing their capabilities or in validating their results, is the fact that quite often the putative optimum docking conformation found by a good algorithm has a total energy which is much lower than the energy, evaluated by means of the same energy function, of the actual docked units, as observed in nature. This is clearly a proof of the inadequacy of the energy model used to perform energy minimization; sometimes, as a partial criterion, an optimization algorithm is considered successful if it could detect the actual docking, even if other conformations had better rank. This is a partial remedy which can be used to partially assess the feasibility of the method, but surely it has to be complemented with other techniques to solve the real task of discovering the docking conformation of unknown complexes.

3.2 A short overview of recent literature

Before introducing our approach to the docking problem, a short review of some relevant literature seems necessary. As a preliminary step, it is important to distinguish the two main classes of methods, namely rigid and flexible docking. In rigid docking, two molecules are considered as rigid bodies. This way the relative positions of atoms within the same molecule do not change. This is of course a crude approximation and a strong simplification; the main advantage of assuming rigid docking is that the number of degrees of freedom is drastically reduced, from the tens of thousands in the general case to just six variables. It can in fact be assumed that one of the two molecules (usually the one with a larger number of atoms) is kept fixed (this molecule is called the *host*), while the other (*guest*) is allowed to rotate around its geometric center and to translate. The six variables are thus three rotation and three translation parameters. The problem, given any kind of force field, becomes thus a low-dimensional optimization problem. However it should be recalled that, although only six free variables exist in this case, energy evaluation again requires the computation of millions of pairwise interactions; also, even in this case, numerical experience tends to support the conjecture that the number of local optima depends on the number of atoms and not on the number of degrees of freedom. Thus even rigid docking remains an extremely challenging global optimization problem. The results of rigid docking might be used as starting point in flexible docking methods.

As it can be easily understood, in flexible docking, as opposed to rigid one, the interacting units are not considered as rigid bodies and their shape is allowed to change in order to favor better couplings. Most flexible docking methods however do not allow every atom to move, as this would make the dimension of the optimization problem unmanageable. The main approaches within this context are those inspired by folding literature, in which the degrees of freedom are not the positions of atoms, but only a limited number of angles within the chain of amino-acids, or those based on the assumption that in flexible docking only those atoms of the two molecules which are

close enough to each other are allowed to move.

A quite detailed survey and numerical comparison of stochastic global optimization algorithms applied to the problem of docking small molecules is presented in [15]. The authors compare some standard global optimization methods, like random walk (in which at each step a new conformation is obtained by randomly perturbing the current one), simulated annealing, smoothing transformation methods, in which the objective function is smoothed, typically by means of a filter, in order to reduce or eliminate local minima. A fourth algorithm is considered, called Trust (Terminal Repeller Unconstrained Subenergy Tunneling), in which after a local optimization, the objective function is modified in such a way that the current local minimum becomes a maximum from which to escape towards a lower minimum point. Similarly to classical tunneling methods (see e.g. [32]) the function is modified in order to avoid visiting worse local optima.

The paper by [27] stands in a quite different position in the literature, as this is one of the few papers in which a deterministic (as opposed to randomized) global optimization method is employed. Here, the well known α-BB method [6] is employed and tested in several cases of peptide-peptide docking; the results reported are very significant, but their applicability to large molecules seems to be quite a difficult task.

In [47] a stochastic method is proposed in which random moves of a flexible ligand are performed, followed by local optimization. The model used is one in which the (small) ligand has several degrees of flexibility, while the host (or receptor) molecule possess a limited flexibility in the neighborhood of the docking site. In [8], again the problem of flexibly docking a small ligand is considered. Here however while, as in the previous paper, local optimizations are performed, the energy function is gradually changed during the docking process, in order to simulate the effective interactions which occur in nature. Again, this paper considers the problem of docking small ligands, a problem which, although far from being solved, can however be considered quite mature, with good and reliable software easily available (see, for example, one of the de-facto standard packages, AutoDock [41]).

In [29] a rigid docking method is described in which a novel scoring

function is used to measure some kind of geometric complementarity between a host and a guest molecule; based upon such a score, a systematic search of possible docking conformation is performed; the approach seems to be well suited only in the case of not too large molecules.

Methods suitable for protein-protein docking are quite recent. The use of rigid docking is significant in most of them, for some method is used as an overall procedure, for others is the first step of a more complex procedure. In the first case, geometric based approaches are quite common. In those methods shape complementarity is used to match the two surfaces (host-guest) either as the main tool ([21]) or together with other information of chemical nature ([13]) such as electrostatic or solvation contributions. In other methods the rigid docking is used as first stage, and then flexible docking in used. In [22] the flexible docking is performed allowing to move only atoms in the interface between the two docked molecules (by the rigid procedure). In [24] the rigid phase is performed using a low-resolution interaction potential, and then the solution is refined with a more accurate potential.

These approaches seem to be particularly interesting, as they decompose the problem into that of finding reasonable docking sites, which can be quite efficiently made by means of rigid docking, and than of a refining phase. The good results reported in these papers encourages us in improving our methods: having a good procedure for rigid docking might prove to be a fundamental step towards a complete and computationally reasonable package for protein-protein docking.

3.3 An energy model for protein-protein docking

We describe a global optimization procedure based upon the use of the Amber force field. It should be again observed that the use of this force field was mainly dictated by its wide availability and its convenient implementation, which are in the public domain. Different force field should perhaps be used in order to avoid some of the pitfalls which will be described in the section devoted to numerical experiments. We trust that our method is

quite robust with respect to changes in the force field and indeed we made some comparison with a different version which used the GROMOS force field and obtained almost identical results ([5],[25]).

Many force fields found in the literature can be described as in the following:

$$\begin{aligned}
\mathrm{E} &= \sum_{\mathrm{i}\in\mathrm{L}} \frac{1}{2}\mathrm{K}_i^b(\mathrm{r}_i - r_i^0)^2 && (3.1)\\
&+ \sum_{\mathrm{i}\in\mathrm{A}} \frac{1}{2}K_i^\theta(\theta_i - \theta_i^0)^2 && (3.2)\\
&+ \sum_{\mathrm{i}\in\mathrm{T}} \frac{1}{2}K_i^\phi[1+\cos(n\phi_i - \gamma)] && (3.3)\\
&+ \sum_{(\mathrm{i,j})} \sum_{\in C} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}\right) && (3.4)\\
&+ \frac{1}{4\varepsilon_0\varepsilon} \sum_{(\mathrm{i,j})} \sum_{\in C} \frac{q_i q_j}{r_{ij}} && (3.5)
\end{aligned}$$

Here, atoms are considered as balls, and chemical bonds as springs. The first three terms in the above expression consist of so called bonded interactions, as they refer to groups of atoms linked two by two by chemical bonds. They are modeled as oscillations around some equilibrium values. In particular, letting L denote the set of all pairs of atoms linked by a covalent bond and r_i the distance of atoms within these pairs (bond length), term (3.1) represents the energy due to the oscillation around an equilibrium value r_i^0. Symbol A denotes the set of all groups of three consecutive atoms linked by chemical bonds and θ_i the angle formed by these three atoms; term (3.2) takes into account the energy due the oscillation of the angle around an equilibrium value θ_i^0. The angle considered in term (3.3) is the dihedral (or torsion) one formed by the two planes identified by a group of four consecutive bonded atoms (these groups form set T).

The last two terms refer to all possible pairs of non bonded atoms (set C), and r_{ij} is the distance between a pair of atoms (i, j) in C. Term (3.4) is the Van der Waals interaction. The minimum of the pairwise interaction (3.4) is reached when the distance of two atoms is equal to a constant which is

defined as the sum of the van der Waals radii of the two atoms. A_{ij} and B_{ij} are constants which depend on the types of atoms i and j. Finally, the term in (3.5) is the electrostatic interaction, which depends on the electrostatic charges of atoms, respectively denoted by q_i and q_j; ε is a constant and ε_0 depends on the medium.

While the above description can be used as a basis for molecular shape optimization (like, e.g., the well known problem of protein folding), when dealing with docking some important modifications are possible. In particular, when rigid docking is considered, two interacting molecules become rigid bodies; thus no modification of internal structure is allowed. This has the consequence that, when evaluating the energy of a complex, all of the contributions due to bonded interactions account for a constant term. In fact in standard docking no covalent interaction between host and guest is allowed. All of the Van der Waals and Electrostatic contributions caused by non bonded pairs within the same molecule, again contribute a constant term to the energy. So, in evaluating docking positions, the following interaction energy can be considered:

$$E(v) = \sum_{i \in \text{guest}} \sum_{j \in \text{host}} \left(\frac{A_{ij}}{r_{ij}^{12}(v)} - \frac{B_{ij}}{r_{ij}^{6}(v)} \right) + \frac{1}{4\varepsilon_0 \varepsilon} \frac{q_i q_j}{r_{ij}(v)} \qquad (3.6)$$

Here host and guest represent the sets of atoms of two different molecules, r_{ij} is the Euclidean distance between two atoms (one belonging to each molecule), and v is a roto-translation vector, representing the 6 degrees of freedom of the guest molecule with respect to the host. Although this energy function depends explicitly on six variables only, its evaluation requires the computation of all the contributions from pairs of atoms between the host and the guest molecules.

3.4 Two phase Monotonic Basin Hopping method

The method we used to perform rigid docking is the two phase procedure introduced in Section 2.4 embedded in the Monotonic Basin Hopping algorithm. In some more details, the combined method consisting of MBH with

our two phase local searches is presented in Algorithm 7.

Data : Δ, N
$n = 0$, $k = 0$
$y =$ random uniform point in S
$x_0{=}x^\star = \mathcal{LS}(y)$
while $n < N$ **do**
  $z_k =$ random uniform point in $B(x_k, \Delta)$
  $y_k = \mathcal{LS}_{Tf}(z_k)$
  if $L_f(y_k) < L_f(^\star)$ **then**
    $n = 0$
    $x^\star = x_{k+1} = \mathcal{LS}_f(y_k)$
  else
    $n = n + 1$
  $k = k + 1$

Algorithm 7: MBH Two-phase approach

As the first phase function is the result of a transformation of the original objective function we refer to it as the transformed function (Tf). As we already underlined, the transformed function used in this method is problem dependent, and this choice influences strongly the possibility of improving on MBH alone. In [5] some preliminary experiments were performed in a simulated docking problem in which a Lennard-Jones cluster of identical atoms was cut into two parts which were then recombined by means of a docking procedure. The experience gathered in that experiments lead us to the definition of a suitable function:

$$Tf(v;\alpha,\beta,\gamma) = \sum_{i\in\text{guest}}\sum_{j\in\text{host}}\left(\frac{A_{ij}}{r_{ij}^{12}(v)} - \frac{B_{ij}}{(r_{ij}/\alpha)^6(v)}\right) \tag{3.7}$$

$$+ \beta\frac{1}{4\varepsilon_0\varepsilon}\frac{q_iq_j}{r_{ij}(v)} \tag{3.8}$$

$$+ \gamma\sqrt{r_{ij}(v)} \tag{3.9}$$

We use three parameters in order to consider different penalties. In the first term of (3.7), the van der Waals radius is changed through a rescaling of the distance term which appears in the attractive contribution (the term

in which the 12-th power of the distance appears avoids an atom collapsing against another one). Choosing a value of α strictly greater than one has a effect which is similar to a reduction in the van der Waals radius: it is trivial to see that in a generic van der Waals term

$$\frac{A}{r^{12}} - \frac{B}{r^6}$$

the minimum occurs at

$$r^\star = \left(\frac{2A}{B}\right)^{1/6}$$

so that, as a function of α, the transformed van der Waals potential has its minimum in

$$r^\star(\alpha) = \frac{1}{\alpha}\left(\frac{2A}{B}\right)^{1/6}$$

Thus, choosing $\alpha > 1$ has the effect of allowing deeper compenetration of the guest inside the host, thanks to the fact that the pairwise minimum moves towards shorter distances. The second parameter is used in order to weight the importance, during the first phase, of coulomb contribution: much of the literature on docking reports a scarce importance of the electrostatic terms during docking, probably as a consequence of the fact that protein docking in nature occurs inside water, where the electrostatic interactions are somewhat smoothed out. Accordingly, in most of our experiments we just put $\beta = 0$. Another possible way to explain the secondary importance of coulomb interaction with respect of the other contributions of the force field, is to consider the structure of the problem. The contributions to the energy, such as van der Waals and electrostatic interactions, are significant only when the distances involved are small, and it is needed a consistent amount of them to have a sufficiently low energy to guarantee the stability of the conformation. Obtaining a large amount of short distances between atoms of host and guest is possible only if part of the surface of one molecule faces the other one. That is, shape complementarity of the two molecules is fundamental to obtain a "good" docking conformation. Evidence of this are also good results reported by purely geometric approaches. From this point of view using van der Waals alone can be sufficient.

The third term in the penalized energy function is an artificial penalty used in order to strongly attract the guest molecule to the host. This term has no physical meaning (it can be thought as a kind of elastic force between the two molecules) but its use in the two phase method is extremely beneficial. The inclusion of a penalty term of this kind was already proven to be of exceptional computational value in the cited papers on Lennard-Jones cluster conformation problems (see Section 2.4). It is included in order to drive the guest protein quickly towards the host one, even when starting from a very far initial point, and also in order to favor a strong complementarity between the two molecules. In some sense we could think that the guest is strongly pushed against the host; this strongly attractive term is balanced, at short distances, by the steep barrier represented by the repulsive component of the van der Walls term.

After a local minimum of the transformed energy function has been found, a standard local optimization on the original energy function is started. In the case of flexible docking (which is not the subject of this paper) some authors report good results in using softer potentials (e.g., potential in which the van der Waals exponents 6-12 are lowered). It would be extremely easy to include such softer potentials either in the first or in the second phase of our method.

3.5 Numerical results

One of the most difficult parts of the numerical experiments was finding examples and testing the results. As it has been already observed, the choice of a particular force field implies the difficulty (or the impossibility) of working with some sets of proteins; using Amber we were forced to consider only those complexes in which no hetero-atoms (possibly except water) were contained. We scanned the Brookhaven Protein Database in order to find proteins made only of standard amino-acids; moreover we restricted our search to complexes made of two molecules of comparable sizes. Finally, in the resulting set, we choose those complexes which were not too large. After

this initial screening, we considered the following examples:

`1ciq` (proteinase inhibitor)
`1egp` (proteinase inhibitor)
`1bxp` (peptide/toxin complex)
`1l6e` (protein kinase and dimerization domain)
`1a03` (calcium-binding protein)
`1kl0` (transferase)

For each of these complexes, the following procedure was applied in order to prepare the molecules:

1. Each file was scanned for hetero-atoms; in particular, given the pre-screening made on the pdb database, no hetero-atom is present in these structure except water. If water was present, it has been deleted;

2. Each complex was scanned to check the presence of hydrogen atoms; depending on the procedure used to obtain atom coordinates, some pdb files report hydrogen positions, while some others do not. If hydrogens were absent, they were added to the pdb file by means of the `add_hydrogens` method of the BALL [10] library. Among these examples, only `1ciq, 1aap, 1kl0` needed the insertion of hydrogen atoms.

3. the resulting PDB was locally optimized using the Amber force field and letting every atom in the complex move; this all-atom local optimization was performed in order to obtain a target protein-protein complex which is indeed a local minimum with respect to the force field used. The Amber force field was implemented without any cut-off and the local optimization used was the `minimize` method in the `ConjugateGradientMinimizer` class of the BALL library. The resulting structure was then saved as a pdb file.

4. for each molecule, composed of two amino-acid chains, one of the chain was labeled host and the other guest. The criterion used has always been that of attaching the guest label to the smaller chain.

5. the interaction energy between the host and the guest was recorded; this information was obtained by computing the energy of the host, that of the guest and that of the complex separately and then defining

$$E_{interaction} = E_{Complex} - (E_{Host} + E_{Guest})$$

As it was expected, only van der Waals and Electrostatic terms were significantly different from zero, while all the other components, known as stretch, bond and torsion, were null.

The following table summarizes some characteristics of the prepared complexes; in particular we report the name of the pdb file, the number of atoms in the guest and host molecules, the interaction energy between the host and the guest in the optimized complex, the root mean squared distance (RMSD) between the original and the optimized complex (for complexes for which we had to add hydrogen atoms, the RMSD was computed only with respect to heavy, i.e. non hydrogen, atoms).

pdb	# Atoms Host	# Atoms Guest	Interaction Energy (kJ/mol)	RMSD Å
1ciq	593	415	-1427.13	0.487542
1egp	637	353	-1394.93	0.412625
1bxp	1038	267	-1709.91	1.04548
1l6e	763	763	-1412.08	0.842386
1a03	1448	1448	-1405.42	0.277063
1kl0	2011	267	-475.210	0.539769

The numerical experiments were performed by running 150 independent instances of our two-phase monotonic basin hopping method as described in detail in section 3.6; in order to have more reliable and comparable results, we used the same parameters in all the tests and also used the same sequences of random numbers in generating the 150 starting conformations. Each starting conformation was obtained by translating the geometric centers of both the guest and host to the origin of the three-dimensional space; then the guest was given a uniform random rotation around its center and successively was translated to a random position at distance 100Å from the

origin. Inside the two-phase monotonic basin hopping method the perturbation vector was chosen randomly; in particular, the perturbation of the three rotation variables was chosen uniformly on $[-0.5\pi, 0.5\pi]$, while the perturbation on the three translation variables was generated uniformly in $[-5\text{Å}, 5\text{Å}]$. These parameters were chosen after some experimentation with molecule `1ciq` and kept constant for all subsequent experiments.

For the definition of the transformed potential energy function (3.7–3.9) we used the following parameters: $\alpha = 1.6$, $\beta = 0$ (no use of Coulomb interactions in the first phase) and $\gamma = 5$. The run was stopped as soon as for the last 30 iterations no improvement was made.

3.6 Discussion on Numerical results

The proposed procedure was coded in C++ and run on an Intel-based Linux PC at 1600MHz. As a local optimization routine, both for the first and for the second phase, we used a C++ version of the limited memory BFGS quasi newton method described in [33]. The numerical results obtained can be more easily analyzed one by one:

1ciq : this is a relatively small example which was used as a test for the tuning of parameters. For this complex several different combination of parameters were used in order to find a single set of constants to be used in all subsequent experiments. With the set of parameters which was subsequently used for all other complexes, we obtained 2% success rate and the best docking found was indeed the correct one. Changing the γ penalty term in (3.9) from 5 to 1 lowered the success rate (in 150 trials)to 1%; augmenting the same parameter to 10 made the success rate go to 1.33% – it has to be remarked that we used common random numbers, so that in each experiment the same starting conformations were used. Among many other trials, here, following what many authors suggest, we tried also a first phase potential in which the van der Waals term is softer, an effect which was obtained by lowering the 12-6 exponents in (3.7) to 8-4, but no success was obtained. Also we

obtained no success when eliminating the γ penalty term.

1egp : for this complex, which consists of two relatively small-sized molecules, we could find the correct docking in 4.7% of the runs performed (i.e. we obtained 7 successes in 150 random trials). For this complex we compared the performance of our two-phase method with a single phase one, i.e. with a standard monotonic basin hopping method, initialized from the same starting points and with all the relevant parameters kept constant. In 150 trials no success was obtained: the best conformation, which was found in 56% of the runs, had an interaction energy -942.04 kJ/mol, much worse than the correct one; the computed RMSD of the best result obtained without the use of the first phase was 10.23 Å, an unacceptable result in this context. As a confirm of this, Figures 3.1 and 3.2 display the optimal docking found and the one obtained with just the single phase basin hopping (in these as well as in the following Figures, we colored in light blue the host and in orange the guest molecule).

This result shows that the compression term introduced in the penalty, as well as the reduction in the van der Waals radii, is extremely beneficial.

1bxp : in this case the optimum docking is found 16 times out of 150 trials (10.7% success rate)

1l6e : here the success rate was even higher, 13.3%, but there were false positives, that is the correct docking was not the one with minimum interaction energy. We found 5 different conformations which, according to the interaction energy, ranked best than the correct one. This is clearly not a failure of the algorithm, but of the force field: if the docking observed in nature is indeed a global minimum of the energy function, this is a clear proof that the Amber force field is not an appropriate model for the interaction energy. Finding a more appropriate model of the energy is a difficult problem which several authors are dealing with. We are however quite confident that our algorithm,

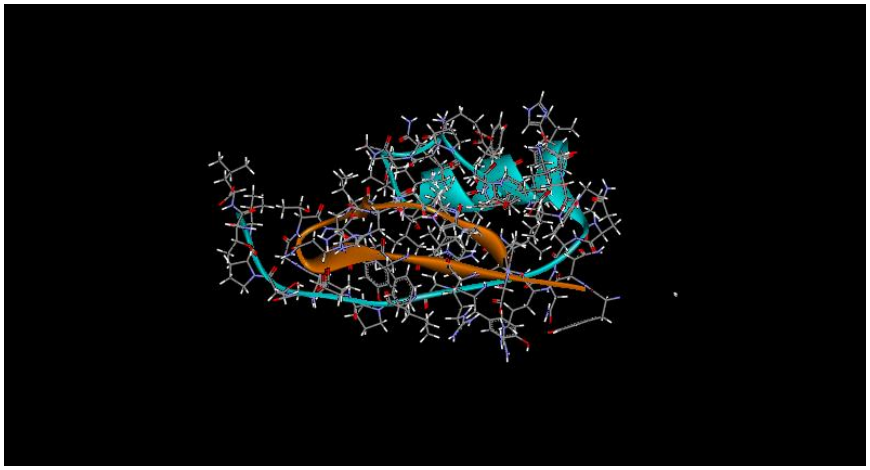

Figure 3.1. Optimal docking of 1egp found with our two-phase method (energy: -1183.6 kJ/mol)

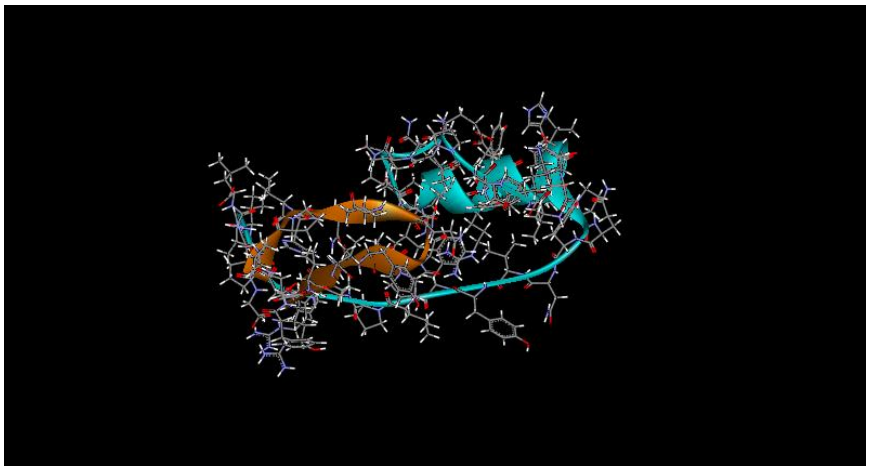

Figure 3.2. Optimal docking of 1egp found with a standard basin-hopping method (energy: -942.04 kJ/mol)

given a different and more reliable force field, will certainly improve its performance on these examples.

1a03 : the correct docking was found as the best conformation in just one trial (success rate: 0.67%). It might be the case that with some parameter tuning this docking could have been discovered more easily, but in order to obtain a validation of our method, the experiments were all performed with exactly the same parameter sets.

1kl0 : here we could find the correct docking (with success rate 1.3%), but most of the other conformations had a better ranking - i.e. almost all of the results were false positives.This is a consequence of the force field used; in this case, looking at the picture of the molecular structure in figure 3.3, it is easy to notice that there are many "good" docking sites.

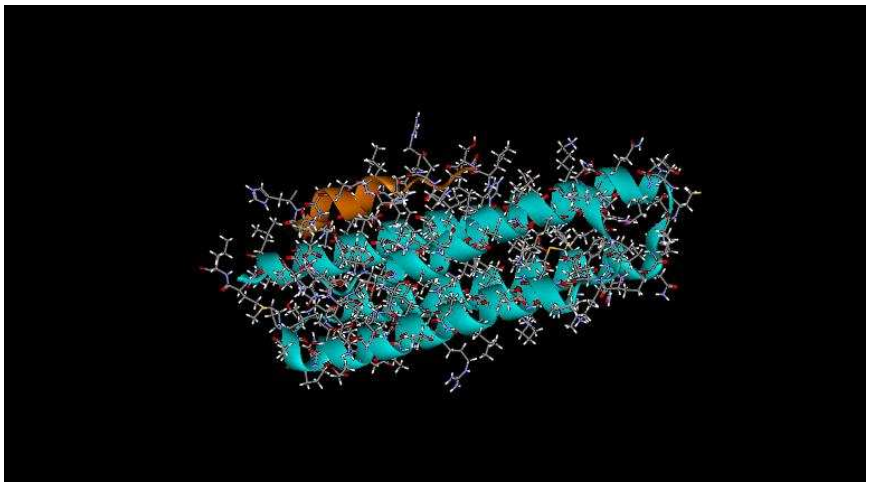

Figure 3.3. Optimal docking in 1kl0

We actually performed a large number of different experiments on different structures. Here we reported some of those results which we consider as most significant. There is a great difficulty, in this field, in performing sensible numerical experiments, as data on complexes, although easily available, are very difficult to use, and no test set was available in the literature

at the moment of our experiment. At the moment of our first tests, some authors ([12]) have been working on a suitable benchmark for testing docking algorithms. We are now working on some problems from that test set (it is available on the web at [9]). It is still difficult to apply the model to these examples. For this reason we were forced to choose only a subset of the original test set. The preliminary results are encouraging. For example on `2kai` the rate of success is 2.7% and on `1ugh` is 39%, in both cases more than double than the rate of success using the original MBH procedure. On `1udi` we get the same results of MBH, but anyway a very high number of successes (62%).

To overcome some of the difficulties of the docking procedure, in the future we plan to work in two different directions. First of all investigating the use of simpler models to reduce computational time and to possibly increase the number of tests that we can handle. The second step is further improving the global optimization method to take into account special properties of the energy landscape. As we observed in Chapter 2, the docking problem is supposed to have a funnel-like structure. Hence we have been working on methods to exploit this property (see Chapters 4-5) and we plan to adapt them to the docking problem.

Chapter 4

Smoothing

A funnel function can be roughly described as a simple function, disturbed by some sort of noise, that produce a large number of local minima. Motivated by this, quite naturally, some authors [39, 40, 44] proposed filtering approaches: if we could filter the high frequencies which perturb the funnel structure, then it will be possible to recover the underlying funnel structure and use a standard global optimization method on the filtered function (which is much easier to globally optimize) in order to reach the global optimum.

One of the main drawbacks of smoothing methods is the fact that they are based on the desire of directly smoothing $f(x)$, and, as it frequently happens, if this function is extremely oscillating, it may happen that either the smoothed function is very different from the original one, or, if the smoothing factor is small, the smoothed function is again multimodal, and optimizing it is as difficult and error-prone as optimizing the original one.

4.1 Smoothing local searches

We will try to exploit the interesting characteristics of smoothing methods for the optimization of $L(x)$. We believe that it is better to filter the piecewise linear function $L(x)$ because it is less oscillatory than $f(x)$; Figure 4.1 shows $L(x)$ for the function presented when we introduced funnels (see Figure 2.1). We apply smoothing to $L(x)$ for two reasons. First, $L(x)$ is

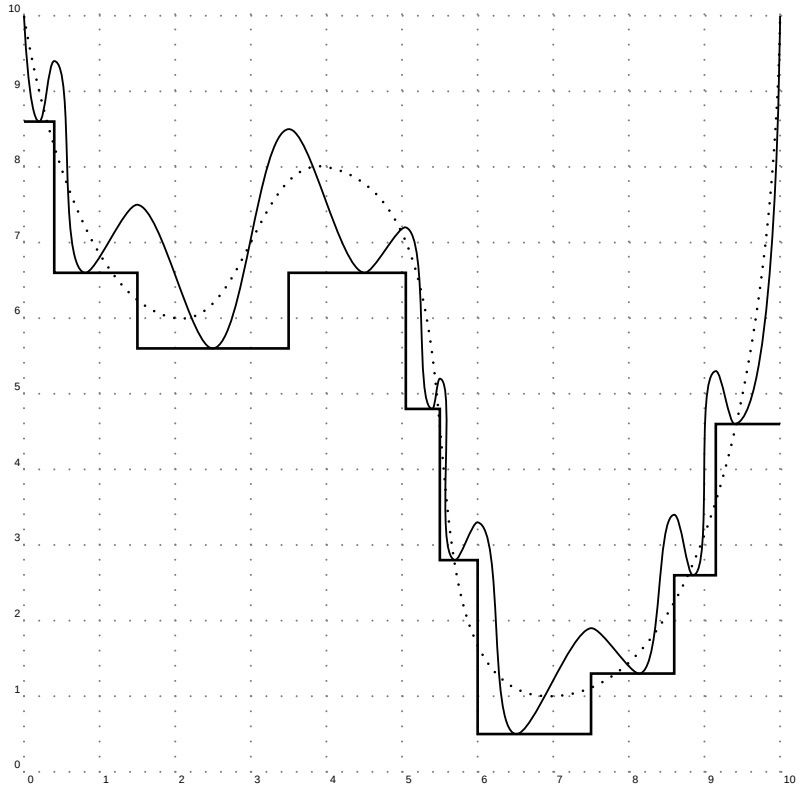

Figure 4.1. Example of the effect of local minimization

a piecewise constant function for which descent directions are difficult to define (the gradient, when defined, is always zero). Second, we expect the smoothing to provide a more global view of the function. In order to build a more reliable method for large-scale, funnel-type global optimization prob-

lems, sampling L coupled with smoothing L might prove a good strategy. If we look at Figure 4.1, it can be immediately noticed that a very good smoothing effect has already been achieved by simply observing L, the results of local searches, in place of f. It can be noticed that the oscillation is reduced to a zero volume part of the domain, the set of the discontinuity points. However, in order to fully exploit the funnel structure, a smoothing method should be applied to L: this way the piecewise constant structure of L will be replaced by a smooth function which contains information of descent directions. It is in fact evident that in a piecewise constant function no local information is able to provide guidelines on descent steps; however, if smoothing were possible, the smoothed function could help in finding appropriate descent directions and thus guide the search towards the global optimum.

4.1.1 Theoretical framework

Given a real valued function $f : \mathbf{R^n} \to \mathbf{R}$ and a smoothing kernel $g : \mathbf{R} \to \mathbf{R}$, which is a continuous, bounded, non-negative, symmetric function whose integral is one, the g–transform of f is defined as

$$\langle f \rangle_g(x) = \int_{\mathbf{R^n}} f(y) g(\|x - y\|)\, dy.$$

The most widely used kernel in the literature is the Gaussian kernel

$$g(z) \propto \exp\left(-z^2/(2\sigma^2)\right)$$

but by no means is this the only one which has been used in practice (we used the symbol $\propto$ to avoid writing a multiplicative constant which plays no rôle in the methods we are presenting here).

We propose to apply a smoothing transformation to the piecewise constant function L and, thus, we would like to estimate the transformed function

$$\langle L \rangle_g(x) \quad = \quad \int_{\mathbf{R^n}} L(y) g(\|x - y\|)\, dy. \tag{4.1}$$

In Figure 4.2 we plot the piecewise constant result of local searches of Figure 4.1 with a few examples of Gaussian transforms for different values of σ.

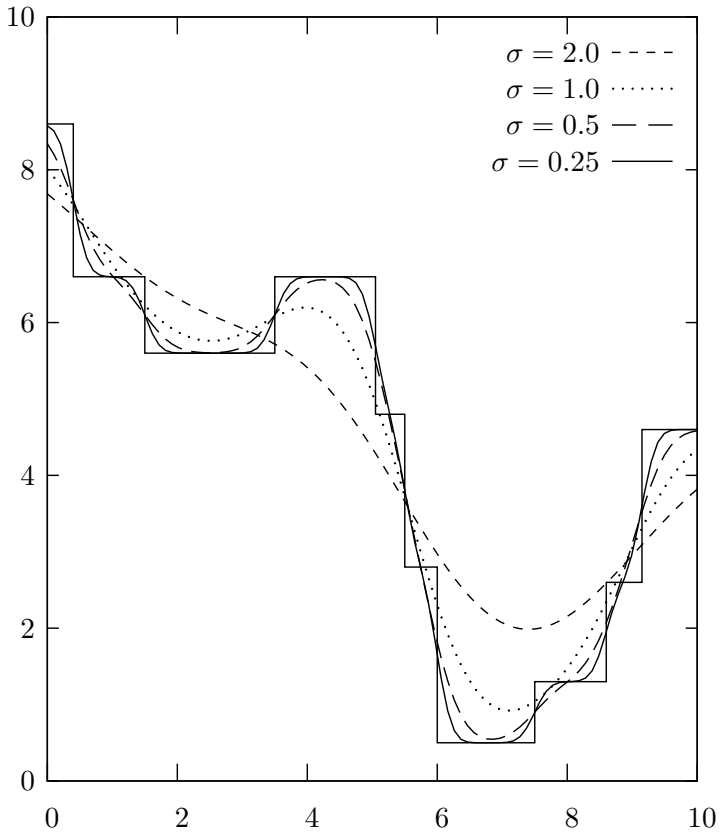

Figure 4.2. Gaussian filtering of $L(x)$

This task, apart from trivial examples, is usually an impossible one, as the analytical expression of L is not available. Nevertheless, it seems worthwhile to explore some of the main characteristics of this transform. Even if it is perfectly clear that an explicit computation of (4.1) is impossible, in this section we will analyze its behavior in the simplest possible case, i.e. that of a function of a single variable which possesses a single funnel.

4.2 Theoretical analysis of the one-dimensional case

In the one-dimensional case it may be assumed without loss of generality that the regions of attraction of different local optima are contiguous segments and thus

$$L(x) = \sum_{i=1}^{N} V_i \, \mathbf{1}_{[a_{i-1}, a_i)} \tag{4.2}$$

where $a_0 = -\infty < a_1 < \cdots < a_{N-1} < a_N = +\infty$ and $V_i \in \mathbf{R}, \mathbf{i} = \mathbf{1}, \ldots, \mathbf{N}$. We observe that in the definition of $L(x)$ some care has to be taken over the value this function attains at points which are on the boundary between two different regions of attraction. Usually these are stationary points and a typical local algorithm will not make any step if initialized at such a point. In general it is not clear to which region of attraction these points should be assigned, if any. If these uncertainty regions have zero measure they are not relevant to our aims, in fact the smoothing function is defined by an integral.

A single-funnel function is characterized by the fact that there exists an index $\ell \in \{1, \ldots, N\}$ such that

$$\begin{aligned} V_{i+1} &< V_i \quad i = 1, \ldots, \ell - 1 \\ V_{i+1} &> V_i \quad i = \ell, \ldots, N - 1. \end{aligned} \tag{4.3}$$

Notice that the global minimum value is V_ℓ. Observe also that (4.3) states the property that a descending (more precisely, not ascending) path down to the global minimum exists from any starting point. We can also assume, without loss of generality, that the origin $x = 0$ is the midpoint of the bottom step, i.e.

$$a_{\ell-1} = -a_\ell \tag{4.4}$$

(this can always be obtained through a translation).

Let $g(x)$ be a kernel (which, in particular, is a probability density function) and let $F(x) = \int_{-\infty}^{x} g(y)\, dy$ be the corresponding probability distribution function. In the following theorem we prove that if g satisfies quite general conditions (enjoyed, among others, by Gaussian densities), then the transform L of a single–funnel step function is unimodal.

Theorem 1. *Let $g(x)$ be a continuously differentiable probability density function whose support is* $\mathbf{R}$*; then, if g is logarithmically concave, i.e. if $\log g(x)$ is concave, and if the step function L defined in (4.2) satisfies (4.3), then either $\langle L\rangle_g(x)$ is monotonic or it is unimodal.*

Proof. By definition of the transform, the value of $\langle L\rangle_g(x)$ is given by:

$$\langle L\rangle_g(x) = \sum_{i=1}^{N} V_i \int_{a_{i-1}}^{a_i} g(x-y)dy.$$

After the change of variable

$$t = y - x$$

we get

$$\langle L\rangle_g(x) = \sum_{i=1}^{N} V_i \int_{x-a_i}^{x-a_{i-1}} g(t)\,dt$$

and, after easy computations,

$$\langle L\rangle_g(x) = V_1 + \sum_{i=1}^{N-1} (V_{i+1} - V_i) F(x - a_i).$$

It is immediate to prove that:

$$\lim_{x\to-\infty} \langle L\rangle_g(x) = V_1, \lim_{x\to+\infty} \langle L\rangle_g(x) = V_N$$

and that

$$\min_{i=1,N} V_i \le \langle L\rangle_g(x) \le \max_{i=1,N} V_i = \max\{V_1; V_N\}$$

so that the transform might be monotonic (increasing from V_1 to V_N if $V_1 < V_N$, or decreasing if $V_1 > V_N$), otherwise it must have at least a minimum point. We would like to show that stationary points are minima.

Taking the first derivative we obtain:

$$L'(x) := \frac{d}{dx}\langle L\rangle_g(x) = \sum_{i=1}^{N-1} (V_{i+1} - V_i) g(x - a_i). \tag{4.5}$$

We wish now to prove that $\langle L\rangle_g(x)$ is monotonic or unimodal by looking at the zeros of $L'(x)$. In fact

$$L'(x) = g(x) \sum_{i=1}^{N-1} (V_{i+1} - V_i) \frac{g(x - a_i)}{g(x)}$$

and its sign, as $g(x) > 0 \, \forall \, x \in \mathbf{R}$, is the same as that of

$$\sum_{i=1}^{N-1} (V_{i+1} - V_i) \frac{g(x - a_i)}{g(x)}.$$

We have:

$$\frac{d}{dx} \frac{g(x - a_i)}{g(x)} = \frac{g'(x - a_i)g(x) - g'(x)g(x - a_i)}{g^2(x)} \tag{4.6}$$

The denominator is always positive. The numerator is nonnegative if and only if

$$\frac{g'(x - a_i)}{g(x - a_i)} \geq \frac{g'(x)}{g(x)}.$$

If, as it has been assumed, g is logarithmically concave, then the first derivative of its logarithm

$$\frac{d}{dx} \log g(x) = \frac{g'(x)}{g(x)}$$

is non-increasing; thus the numerator in (4.6) is nonnegative if and only if $a_i > 0$ and so

$$\frac{g(x - a_i)}{g(x)}$$

is nondecreasing if and only if $a_i > 0$; however, also $V_{i+1} - V_i$ is positive if and only if $a_i > 0$, so $L'(x)$ is the product of a nondecreasing function and a positive one. Thus either it is non zero for all x, or it can be zero either at a single point or it might be zero in a segment. In any case, as the transform either has no stationary points or it must have a minimum, the theorem is proven. □

With slightly stronger assumptions we can obtain a transform which has one and only one minimum point.

Theorem 2. *If in addition to the assumptions of Theorem 1 $g(x)$ is* strictly *log-concave, then the transform has at most a single minimum point.*

Proof. In fact strict log-concavity for a continuously differentiable function means that

$$\frac{d}{dx} \log g(x)$$

is decreasing; thus

$$\frac{g'(x)}{g(x)}$$

is decreasing and, as a consequence, $L'(x)$ is the product of a positive function and of an increasing one. Thus it can have at most one zero. □

Theorem 3. *If, in addition to the assumptions of Theorem 1, g is such that*

$$\lim_{x\to-\infty}\frac{g'(x)}{g(x)}=+\infty,\ \lim_{x\to+\infty}\frac{g'(x)}{g(x)}=-\infty \tag{4.7}$$

then $L(x)$ has a minimum.

Proof. We simply need to show that $L'(x)$ changes sign. Given the assumptions, thanks to the log-concavity of g, then

$$\log g(x-a)-\log g(x)\leq -a\frac{g'(x)}{g(x)}$$

Thus, if $a>0$ the right hand side tends to $-\infty$ and we obtain that

$$\lim_{x\to-\infty}\frac{g(x-a)}{g(x)}=0.$$

Now,

$$\begin{aligned}
\frac{L'(x)}{g(x)} &= \sum_{i=1}^{N}(V_{i+1}-V_i)\frac{g(x-a_i)}{g(x)}\\
&= \sum_{i:a_i<0}(V_{i+1}-V_i)\frac{g(x-a_i)}{g(x)}+\sum_{i:a_i>0}(V_{i+1}-V_i)\frac{g(x-a_i)}{g(x)}\\
&\leq \sum_{i:a_i<0}(V_{i+1}-V_i)\frac{g(a_N-a_i)}{g(a_N)}+\sum_{i:a_i>0}(V_{i+1}-V_i)\frac{g(x-a_i)}{g(x)}\\
&\to \sum_{i:a_i<0}(V_{i+1}-V_i)\frac{g(a_N-a_i)}{g(a_N)} \quad \text{for } x\to-\infty\\
&< 0
\end{aligned}$$

A similar reasoning can be followed to show that if $x\to+\infty$ then $L'(x)/g(x)$ tends to a strictly positive quantity. □

From the above theorems, it immediately follows that, for example, if g is a Gaussian kernel, then the transform has always one and only one minimum point. It is also easy to show that if the variance of the density function g is sufficiently small, then the minimum point occurs, as expected, inside the interval corresponding to the bottom step of the objective function (we use here the fact that $E(g) = 0$, g being a symmetric density):

Theorem 4. *There exists a $\bar{\sigma} > 0$ such that if*

$$Var(g) = \int_{-\infty}^{\infty} x^2 g(x)\, dx \leq \bar{\sigma}$$

then the minimum of $\langle L \rangle_g(x)$ belongs to $[a_{\ell-1}, a_\ell]$

Proof. As the variance goes to zero, by Tchebychev's inequality, the probability tends to be concentrated at zero. So $\lim_{\bar{\sigma}\to 0} g(x) = 0$ for all $x \neq 0$. Thus

$$\lim_{\bar{\sigma}\to 0} L'(a_{\ell-1}) = (V_\ell - V_{\ell-1})g(0) < 0$$

and

$$\lim_{\bar{\sigma}\to 0} L'(a_\ell) = (V_{\ell+1} - V_\ell)g(0) > 0.$$

□

Being restricted to the one-dimensional case, all these results are of limited usefulness. However, they at least suggest the existence also for the multidimensional case of some notion of a "path of descending steps down to the global minimum", which leads to results similar to those obtained for the one-dimensional case. Even if we could prove similar results on n-dimensional space (under some hypothesis on the function $L(x)$), they would be of no practical use. In fact as we noticed when we introduced $\langle L \rangle_g(x)$, it is not possible to obtain an analytical expression of this function. This is usually not possible also because we do not even have an analytic expression for L. However, $\langle L \rangle_g(x)$ can be approximated and the search over $\langle L \rangle_g(x)$ can be substituted by search over its approximation. Chapter 5 will be dedicated to the definition of an approximation and to the resulting global optimization algorithms.

Chapter 5

Algorithms based on smoothing

5.1 An approximation scheme

Of course it is impossible to obtain an analytical expression of the transformed function $\langle L \rangle_g$; it is even impossible to obtain a numerical estimate, as the transform depends on values of L in all the domain: of course, apart from the impossibility of sampling L everywhere, even if this would be possible there would be no point in using a smoothing filter, as, at that point, the global optimum of the original would have been already discovered. So a complete description, or even a numerical estimate of $\langle L \rangle_g(x)$ at a single point is out of the question. Considering the fact that only finite samples of L will be available, the approach we followed was that of sampling in a spherical neighborhood of prefixed radius Δ centered at the current point x_0, i.e. in $B(x_0, \Delta) = \{x : \|x - x_0\| \leq \Delta\}$; after sampling, we build an approximation to the restriction of $\langle L \rangle_g(x)$ on $B(x_0; \Delta)$ which should capture some local information of the function L. In other words, in some sense similar to what is done in trust-region methods, we use information and

models in a neighborhood of the current point in order to find a descent direction. Assuming the ball is contained in the feasible set, the restriction of the transform on B can be defined as follows:

$$\langle L \rangle_g^B(x) = \frac{\int_{B(x_0,\Delta)} L(y) g(\|x-y\|)\, dy}{\int_{B(x_0,\Delta)} g(\|x-y\|)\, dy}. \tag{5.1}$$

We notice that the normalization factor is necessary in order to satisfy the requirements of a kernel function. In order to roughly estimate $\langle L \rangle_g^B(x)$ we choose to draw a uniform sample on $B(x_0, \Delta)$ of a prefixed cardinality K. Let $y_1, \ldots, y_K$ denote the points sampled in $B(x_0, \Delta)$. Then a local estimate of the restricted transform is given by

$$\hat{L}_g^B(x) = \frac{\sum_{i=1}^K L(y_i) g(\|x-y_i\|)}{\sum_{i=1}^K g(\|x-y_i\|)}. \tag{5.2}$$

This function shares with (5.1) some important properties:

1. it is continuous
2. for each x it is a convex combination of the values obtained by applying local searches from points inside $B(x_0, \Delta)$ (or, equivalently, obtained through observations of L)
3. the nearer x is to a sampled point y, the stronger the influence of $L(y)$ is on $\hat{L}_g^B(x)$, as in the original smoothing where this relationship holds between each point y in $B(x_0, \Delta)$ and the smoothing $\langle L \rangle_g^B(x)$.

5.2 A numerical algorithm

We already observed that one of the disadvantages of working with $L(x)$ was the fact that, due to its piecewise constant shape, no information could be obtained on descent directions. Here we propose to use the local approximation of the smoothed transform of L given in (5.2) as a local model useful in guiding the search towards the global optimum. In other words, the local approximation is used as a model to predict a descent direction of $L(x)$.

The proposed algorithm consists of an approximation phase and a displacement phase: in the approximation phase we are given a current point x_h and the value of $L(x_h)$; a sample of cardinality K of points in a neighborhood $B(x_h, \Delta)$ of x_h is drawn and L is observed in each point in the sample. The values thus obtained are used to build an approximate filtered function (5.2) which is numerically minimized in $B(x_h, \Delta)$. The global minimum of $\hat{L}_g^B(x)$ in $B(x_h, \Delta)$ is taken as the next current point and the procedure is iterated (displacement phase). In practice, in order to reduce the computational effort, only an approximation of the global minimum of $\hat{L}_g^B(x)$ will be used, which will be obtained by means of some very simple global optimization algorithm like, for example, Multistart. In our numerical experiments we even used just a single local optimization started from x_h. The whole procedure is stopped when no improvement of the record $x^\star$ (that is the best point found so far) has been observed since a prefixed number of steps. A scheme of ALSO (acronym for an Algorithm based on Local Smoothing for Optimization) is presented in Algorithm 8.

Data : Δ, K, N
Initialization
while $n < N$ **do**
  Samples Collection
  $n = n + K$
  Model Construction
  Solve $x_+ = \arg\min_{x \in B(x_k, \Delta)} \hat{L}_g^B(x)$
  Center Update
  $k = k + 1$

Algorithm 8: ALSO

Some details must be added to describe the complete algorithm. Local searches might produce very good local minima and, possibly, local minima which are very far from the current point. In order to speed up the procedure each time a record (i.e., the best local optimum observed so far) is obtained, the procedure interrupts and sets the current point to the record. So the phase "Samples Collection" is realized as:

repeat
  Sample a point in $B(x_k, \Delta)$ and add it to $\mathcal{K}$
until *a new record is found or* $|\mathcal{K}| < K$

For this reason after sampling we check for improvements of the record, and if it is the case we move in the new point and restart the sampling procedure.

When a candidate point x_+ is found, an unconstrained local optimization of the original objective function $f(x)$ is performed because, in general, x_+ is not a local minimum of $f(x)$. This procedure is equivalent to evaluating $L(x_+)$. If we obtain an improvement, the local minimum is taken as a new point, that is the center of the new region B. Otherwise the new center is x_+, the "Center Update" phase resulting is described as follow:

if $L(x) < L(x^\star)$ **then**
  $n = 0$
  $x^\star = x_{k+1} = \mathcal{LS}(x_+)$
else
  $x_{k+1} = x_+$

The whole procedure is described in Algorithm 9. In contrast to other procedures such as Monotonic Basin Hopping, this model allows to move the current center even if no better point is sampled. When the probability of sampling a better point is low or even zero, this can be very important. The model is able to identify the "trend" of the function also using only worse value points. In this way we can move to a new center, likely nearer to the bottom of the funnel (see Picture 5.1).

5.3 Numerical results

In order to test the numerical performance of the proposed algorithm both a set of hard test problems and an alternative algorithm have to be chosen. We decided to compare our method with Monotonic Basin Hopping, because

Data : Δ, K, N
$n = 0$, $k = 0$
$y =$ random uniform point in S
$x^\star = x_0 = \mathcal{LS}(y)$
while $n < N$ **do**
  repeat
    Collect a sample in $B(x_k, \Delta)$ and add it to $\mathcal{K}$
  until *a new record is found or* $\mid \mathcal{K} \mid < K$
  if $\min_{y \in \mathcal{K}} L(y) < L(x^\star)$ **then**
    $n = 0$
    $x^\star = x_{k+1} = \arg\min_{y \in \mathcal{K}} L(y)$
  else
    $n = n + K$
    Set $M = \mathcal{K}$
    Construct the model (using samples in M)
    Solve $x_+ = \arg\min_{x \in B(x_k, \Delta)} \hat{L}_g^B(x)$
    if $L(x_+) < L(x^\star)$ **then**
      $n = 0$
      $x^\star = x_{k+1} = \mathcal{LS}(x_+)$
    else
      $x_{k+1} = x_+$
  $k = k + 1$

Algorithm 9: ALSO

it is extremely efficient when applied to problems possessing a funnel-like structure.

To evaluate the computational performance of our algorithm, we tested it on several functions from the literature that have a funnel structure (see Appendix A).

For each test, that is represented by a function and a choice of parameters, we performed 1 000 independent runs of each algorithm; for each test solved either by our method or by MBH, we used the same seeds in random number generation, so that each of the 1 000 runs was started in the same point for all the methods tested. In both methods we used the value 1 000

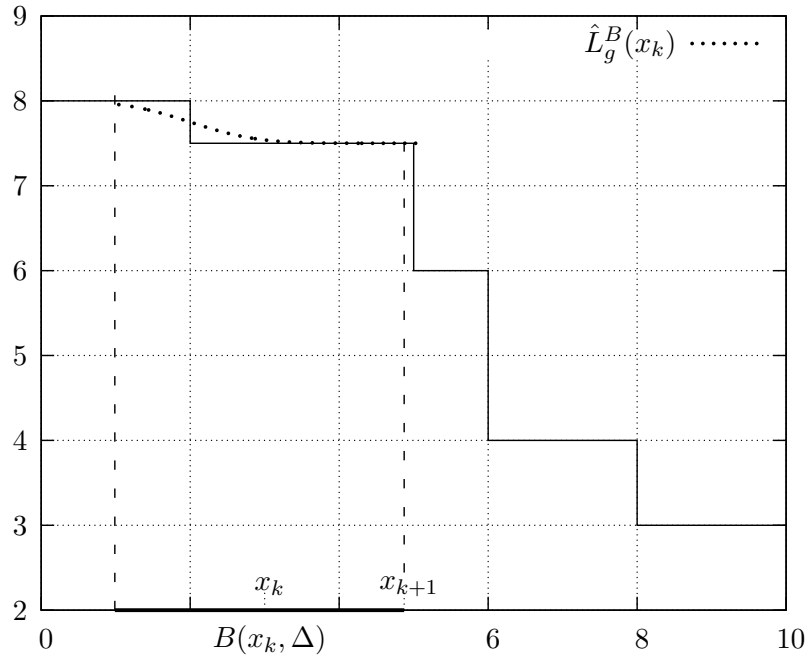

Figure 5.1. An example of a step of ALSO

for the N parameter. We chose a Gaussian kernel for the smoothing and we let its standard deviation be defined as

$$\sigma = \Delta K^{-1/n}$$

This choice for σ originates from the desire, in some sense, to have enough information from the observations in all the neighborhood $B(x; \Delta)$. In other words, assuming that, using a Gaussian kernel, enough weight is placed in a ball of radius σ centered in each observation, the above formula is obtained from the requirement that the volume covered by K balls of radius σ should suffice to cover the ball with radius Δ.

This way the only parameters which remain to be chosen are Δ, the radius of the sphere used both for generating new observations of L and for the local approximation based upon smoothing, and K, the number of observation used in order to build an approximation to the smoothed function.

We choose to employ L-BFGS [33] as a local search algorithm for both methods, with standard parameter settings; however it is felt that the results are quite robust with respect to the local search algorithm used. What in fact might radically change when passing from one type of local search method to a different one, or when changing or tuning the internal parameters of a single local search algorithm, are the number of function and gradient evaluations needed to approximate local optima to a desired accuracy. This is also the reason why we choose not to report the number of function and/or gradient evaluations: in fact these numbers change widely when we change the local optimization method employed or, simply, when a different accuracy is sought in approximating a local minimum. So, statistics on function evaluations, while surely important, does not lend itself very well to answer our fundamental question about how easy is it for a global optimization algorithm to identify the funnel leading to the global optimum. This is the aim of our research and we choose to report the number of local searches as this figure is quite robust: we experimented the effect of changing the accuracy parameter (i.e., the stopping rule in the local search algorithm) and, while obviously we observed significant variations in the total number of function evaluations, we did not observe any statistically significant change in the number of local searches.

For what concerns MBH, we tested several values of Δ in order to find the best value for that algorithm. Our main aim in these experiments has been that of showing that even when a very good algorithm, like MBH, is placed in the best possible conditions, our method still outperforms it.

Finally we stress that the minimization of $\hat{L}_g^B(x)$ employed in the experiments consisted just in a single run of local optimization started from the center of the current neighborhood. We did not extend our experiments to include a true global optimization method in this phase as the results obtained so far had already been so positive that we felt it was not worthwhile to add overhead to the whole procedure. However we plan to analyze the possible benefits of performing a few Multistart steps on function $\hat{L}_g^B(x)$. The tables with numerical results can be found in Appendix B;

The results obtained with the Rastrigin function (see Table B.1) show

a clear improvement in our method with respect to MBH. Even when the best parameter Δ is chosen for MBH, our method generally outperforms it; however it is important to notice that not only does our algorithm require less local searches than MBH, but also it has a significantly greater success rate and, what is particularly important, keeps its good behavior for a quite large set of choices of the parameter Δ. In other words not only is the method better than MBH, but it is also much more robust, so that the choice of parameters is far less critical. This greater robustness of our algorithm is shared by most of our tests, as can be seen from the tables.

Analyzing the behavior of the two methods with Levy's and Ackley's functions (Tables B.2 and B.3), again it is immediate to see that our method has a greater success rate, requires less local searches and is more robust than MBH; differently from what happened with Rastrigin test functions, here increasing the cardinality K of the sample did not improve the performance of our method. This difference may be justified observing that increasing K has two opposite effects: from one side it gives us a better approximation of the smoothed function; from another side, however, as we choose to stop the algorithm after 1 000 local searches with no improvement, a greater value of K gives our algorithm less freedom to move. For example, with $K = 50$ the algorithm might stop after 20 "major" iterations (i.e. those iterations during which a new approximate smoothing is built and used) with no improvement; increasing K to 100, leaves to our method only 10 unsuccessful iterations before stopping. For easy functions, like Rastrigin, the advantages of better approximation compensate the small number of moves; for more challenging tests this is no more true. However, analyzing the output of our algorithm, we noticed that in any case the algorithm almost always stops with a record value which is significantly lower than that found by MBH even in the cases where failure occurs. Another observation should be made for these two classes of test functions: looking at the tables it is seen that when both MBH and our method display the best performance, their behavior is almost indistinguishable. In these cases our method is actually following the same decisions as MBH: in practice before a sample of K observations is taken a new record is almost always observed. In these

cases, no approximation is performed and the method becomes the same as MBH (with no overhead). However, again, we observe that when the parameters of MBH are not optimal, our method still outperforms it. In other words, our method is equal to MBH when it is easy to follow descent steps towards the global optimum; instead, when MBH gets stuck in a local optimum, our algorithm often finds a descent path which improves the record of MBH and, quite often, ends up in the global optimum.

The Schwefel test function is particularly interesting as, differently from the previous ones, it is not a single-funnel one. So both methods might, and in fact do, end up in different optima. Again, however, it can be easily seen that our method is significantly more successful and efficient than MBH, the more so when the dimension increases. When the radius Δ is sufficiently large, the identification of the funnel containing the global optimum becomes quite easy.

The modification introduced on the Amplified Rastrigin function (see the results in Table B.5), while not altering significantly the regions of attraction of local optima, changes their relative value; thus it is expected that global optimization methods will get into more trouble in finding a descent path towards the global optimum. In fact the results, both for MBH and for our method, are worse than those found on the original Rastrigin function. However, again, our method, for all choices of the parameters, with a single exception, outperforms MBH both in the number of successes and in the number of local searches required to observe each success.

Finally we analyze the scaled variant of the Rastrigin function (table B.6). The motivation for this modification is that we wished to check the ability of our method of finding descent paths even when the level sets near local optima are not spherical. Changing the scale of some of the variables has the effect of generating ellipsoidal level sets. It is thus quite evident that a method like MBH, which samples in spheres, will have great difficulties: in fact if the radius of the sphere is too small, MBH will not find an improvement and will get stuck in a local minimum. On the other hand, if the radius is too large, sampling will be performed in a region which is so large that the probability of finding a record will usually be very low.

Quite surprisingly, however, our method is able to correctly identify descent directions, even when sampling in small spheres, as is clearly displayed by the results in the tables. Of course the higher the dimension, the worse is the behavior of the algorithm; however the number of successes is quite high. We also performed a final test to check whether the number of successes could be improved by letting the algorithm run longer; we thus increased the N parameter from 1 000 to 2 000 (see Table B.6) and in fact observed a significant improvement: this reinforces our suspicion that our algorithm successfully extracts information of good descent direction which enables it, if not stopped prematurely, to end up in the global minimum with high probability.

We also performed some preliminary computations comparing the behavior of MBH and our approach with respect to a few rigid protein-protein docking problems. We briefly comment here on some of these results; with respect to complex `1bxp` our approach was significantly superior to MBH for any parameter choice (in some cases the rate of success was even doubled). With respect to complex `1ciq` our approach was overall better than MBH but the improvements were not as large as for complex `1bxp`. With respect to complex `1a03`, though superior for some parameter choice, overall our approach appeared to be inferior to MBH. We do not comment any further these results, as they are preliminary. Docking problems may have more than one funnel: this structure might hide the performance of our smoothing approach, as it is clear that any effort spent in smoothing the function in a funnel which does not lead to the global optimum is generally wasted. Further research in this area is planned, in particular using the test set we introduced in Chapter 3.

5.4 A trust-region framework

We observed from the results obtained with ALSO that the use of a model can produce good results on funnel function, and in particular can improve with respect of MBH in terms of performances and robustness. Still the radius parameter is critic in these kind of methods. The use of a fixed value

for Δ is restrictive for the length of the steps we are allowed to take. In addition, it is not clear a priori what value Δ should take. We do know that a small radius results in small steps and, hence, in slow convergence, but large values of Δ can result in poor agreement between the model and the function and, hence, in useless candidate points. A possible approach is construct a strategy to adaptively change the radius Δ. This parameter represents the region of validity of the model and at the same time limits the length of the step. An analogy with trust-region methods in local optimization exists. Motivated by this we propose to embed ALSO within a trust-region framework.

Trust-region methods are traditionally used in local optimization to force convergence to local minima from remote starting points. To extend the trust-region framework to global optimization, we introduce the concept of global quality, and we use a model improvement step.

The use of trust-region methods for local optimization dates back to [30]; see the comprehensive book [14]. We review the basic idea of a trust-region method for the unconstrained local optimization of a smooth function $f(x)$ (a simplified scheme is in Algorithm 10). At a given iterate x_k, we construct a (second order) Taylor series model $m_k(x)$. This model is then minimized in the trust region, usually a ball of radius Δ_k around x_k. If the new point, x_+, improves the objective, we move to it and construct a new model. Otherwise, we improve the agreement between $f(x)$ and the model $m_k(x)$ by reducing the trust-region radius Δ_k. Convergence follows from the fact that $f(x)$ and $m_k(x)$ agree up to first order. To extend the trust-region framework to global optimization of $L(x)$, we need to modify the smooth local trust-region algorithm. Specifically, we have to account for the fact that $L(x)$ is non-smooth and that $\hat{L}_g^B(x)$ does not agree with $L(x)$ to first order. More important, we wish to avoid getting trapped in local minima.

As in ALSO, at each iteration we construct the model $\hat{L}_g^B(x)$ around our current iterate x_k. Specifically, we choose K samples (uniform) inside the current trust-region $B(x_k, \Delta_k)$ and perform a local minimization of $f(x)$ from each sample. If we find a new record during this phase, we simply move to the new point and construct a new model. Otherwise, we apply a

Data : Δ, $\bar{x}$, N
$k = 0$, $\Delta_k = \Delta$
$x_0 = \bar{x}$
while *stop condition not fulfilled* **do**
  Construct the model $m_k(x)$
  Solve $x_+ = \text{argmin}_{x \in B(x_k, \Delta_k)} \hat{L}_g^{B_k(x)}$
  if $f(x_+) < f(x_k)$ **then**
    $x_{k+1} = x_+$
  else
    Improve the model ($\Delta_{k+1} = \frac{\Delta_k}{\alpha}$)
  $k = k + 1$

Algorithm 10: Basic trust-region

local minimization to the model inside the trust region and obtain

$$x_+ = \underset{x \in B(x_k, \Delta_k)}{\text{argmin}} \hat{L}_g^{B_k(x)}.$$

At this point the method diverges from ALSO through the introduction of the trust-region ideas. To decide whether to accept a step, we compute the ratio of the actual to the predicted reduction, namely,

$$\rho = \frac{L(x_k) - L(x_+)}{\hat{L}_g^{B_k}(x_k) - \hat{L}_g^{B_k}(x_+)},$$

noting that the predicted reduction $\hat{L}_g^{B_k}(x_k) - \hat{L}_g^{B_k}(x_+)$ is always nonnegative. We accept the new point x_+ if we observe sufficient decrease, that is $\rho \geq \eta_1 > 0$. If the step is very successful, $\rho \geq \eta_2 > \eta_1$, and the trust-region is active, $\|x_k - x_+\| \geq \Delta$, then we increase the trust region radius for the next iteration. As long as $\rho \geq \eta_1$, we refer to the iteration as successful, otherwise ($\rho < \eta_1$) the iteration is referred to as unsuccessful. Unsuccessful trust-region iterations require special attention in the global optimization setting. In smooth local optimization, reducing Δ is guaranteed to improve the agreement between the model and the objective function. The same is not true in the global optimization context. This is consequence of the piecewise nature of $L(x)$. If the trust-region is completely contained in a basin of attraction, all the samples would have the same value, and $\hat{L}$ would

be constant, and so flat. Our model is not anymore able to produce any information on descent directions. In other words we loose the property to model global information, that is the funnel structure of the problem. Hence, we introduce a measure for the global quality of our model $\hat{L}_g^{B_k(x)}$, based on M, the set of collected samples

$$q(\hat{L}_g^{B_k(x)}) = \frac{\max_{i \in M} \mid \{y_j : L(y_j) = L(y_i)\} \mid}{M}, \tag{5.3}$$

that is the largest number of samples with the same objective value, divided by the total number of samples. Clearly, $0 \leq q(\hat{L}_g^{B_k(x)}) \leq 1$, and a value close to 1 means that a large number of samples have the same function value and stem from the same "flat region" of $L(x)$. A smaller value of $q(\hat{L}_g^{B_k(x)})$ implies that the samples represent the global nature of the function $L(x)$ better.

In our algorithm, we compute $q(\hat{L}_g^{B_k(x)})$ at every unsuccessful iteration. If it is larger than a fixed value $\bar{q}$, we remove all but one sample from the largest set, increase the trust-region radius, and obtain new uniform samples in $B(x_k, \Delta_{k+1}) \backslash B(x_k, \Delta_k)$. The motivation for this step is twofold: it improves the global nature of the model $\hat{L}_g^{B_k(x)}$, and it increases σ, thus smoothing the model.

The increase of σ arises because we have adopted, as in ALSO, the following formula for calculating the smoothing parameter, depending on the trust-region radius Δ and the number of samples K:

$$\sigma = \frac{\Delta}{K^{1/n}}, \tag{5.4}$$

where n is the dimension of the problem. Since both the number of samples and the trust-region vary, the parameter σ is updated during the algorithm.

We can now state the complete global optimization trust-region algorithm. Let $0 \leq \bar{q} \leq 1$ be a bound on global quality. Let the trust-region parameters $0 < \eta_1 < \eta_2 \leq 1$ be fixed. Let N be a given upper bound for the number of samples, $\beta_1, \beta_2 > 1$ and $0 < m \leq 1$. In Algorithm 11 a scheme is presented.

We conclude this section with a few remarks on our trust region. We have a pool of samples M that is used to collect measures. If we do not achieve

Data : Δ, K, N
$n = 0$, $k = 0$, $M = \emptyset$, $\Delta_k = \Delta$
$y =$ random uniform point in S
$x^\star = x_0 = \mathcal{LS}(x)$
while $n < N$ **do**
  repeat
    Sample a point in $B(x_k, \Delta_k)$ and add it to $\mathcal{K}$
  until *a new record is found or* $\mid \mathcal{K} \mid < K$
  if $\min_{y \in \mathcal{K}} L(y) < record$ **then**
    $n = 0$, $M = \emptyset$
    $x^\star = x_{k+1} = \arg\min_{y \in \mathcal{K}} L(y)$
  else
    $n = n + K$
    Add samples in $\mathcal{K}$ to M
    Construct the model (using samples in M)
    Solve $x_+ = \text{argmin}_{x \in B(x_k, \Delta_k)} \hat{L}_g^{B_k(x)}$
    if $\rho \geq \eta_1$ **then**
      $n = 0$ and set $M = \emptyset$
      $x^\star = x_{k+1} = \mathcal{LS}(x_+)$
      if $\rho > \eta_2$ *and StepLenght* $\geq \Delta$ **then**
        $\Delta_{k+1} = \Delta_k \beta_1$
    else
      $x_{k+1} = x_k$
      if $q(\hat{L}_g^{B_k(x)}) \leq \bar{q}$ **then**
        Δ_{k+1}=Δ_k / β_2 (only every two consecutive steps)
      else
        Throw away all but one sample with same value
        $\Delta_{k+1} = \Delta_k \beta_1$
  $k = k + 1$

Algorithm 11: TRF

good agreement, we collect new samples, and we add them to the set M (with the old ones). We use the following strategy for improving model agreement. If the global quality is low, we increase the trust-region radius and throw away part of the samples; in particular, we delete all but one of the

samples with the same value that determined the poor global quality. The aim of this strategy is to increase the diversity between samples and avoid the region around the center x_k from becoming flat. We decrease the radius only if the agreement is poor and the global quality is sufficient, let call this condition of shrink. We note that reducing the trust-region radius can be considered dangerous for a global optimization strategy because it restricts the method to a local search. Hence, we take a conservative approach: we reduce the trust region only in the second of two consecutive shrink steps ($\rho < \eta_1$ and $q \leq \bar{q}$). That is, the above condition must be verified twice in a row in order to reduce the radius. We note that, in case of poor agreement, we update the trust-region parameters, but we can keep the old points, in fact we do not move the center, so the old samples are contained in the actual trust-region region. If we do move, the new point can be far away from the old point, and the information on the latter (in particular, collected samples) may be useless for the former.

5.5 Numerical results of the adaptive version

The same test functions as for ALSO are used (see Appendix A). As in the previous tests, for each test function (different dimensions are considered as different tests) 1 000 trials from randomly generated points inside the domain are performed. The stopping criterion is the same for all the methods presented. We consider a local search unsuccessful if there is no global improvement for the objective function. After 1 000 consecutive unsuccessful local searches we stop the algorithm. Every time a global improvement is obtained, we restart the count of unsuccessful local searches. Local searches are performed with the L-BFGS algorithm [33].

The trust-region algorithm is run with the following parameter values. The initial trust-region radius Δ is taken as in the previous tests (for ALSO) and is different for every run. The decrease factor for the radius is $\beta_1 = 1.11$ and the increase factor is $\beta_2 = 1.2$. The global quality threshold is $\overline{q} = 0.6$. The parameters for step acceptance, and trust-region increase are $\eta_1 = 0.001$, $\eta_2 = 0.75$, respectively. We compare our results with the ones

obtained with ALSO. We report as reference the results of the tests using MBH, even if in the majority of the cases ALSO outperforms it. We test all algorithms for a range of initial trust-region radii. The initial radii were chosen to optimize the performance of MBH. In other words, our new algorithm competes against "optimal Δ values" of the other methods. In practice, it is unlikely that a user knows a good value for Δ, and in our examples the "optimal" Δ is the result of numerous tests. Both reference methods, ALSO and MBH, have no strategy to adapt the radius Δ. We compare our method also with the adaptive version of MBH, presented in Section 1.2.3.1. The parameters $\bar{N}$ and l are chosen as in [35]; that is, $\bar{N} = 10$ and $l = 0.8$. We test AMBH for all the radius values used for the other methods. The results are given in Appendix B. Every test function is presented in a separate table. For each method we report the percentage of successes and the average number of local searches for each success. For trials without success the number of local searches for success is set to ∞. The average number of local searches for success is a measure of the computational effort needed for finding the global optimum.

The most notable difference between the trust-region scheme and ALSO is the number of local searches for success. For ALSO, this number is very sensitive to the initial trust-region choice and typically varies by one or two orders of magnitude. For instance, for Ackley ($n = 50$) it varies from 288 to 48 433; for Levy ($n = 50$) from 47 to 1 412; and for Schwefel ($n = 5$) from 2 235 to 4 056. The variance of the number of local searches for our method is much more modest. Clearly, the new trust-region method is far less sensitive to the choice of the initial radius.

We also observe that the new trust-region algorithm uses far fewer local searches than does ALSO. In some cases, ALSO uses up to 20 times the number of local searches used by our trust-region framework. This is an important improvement: it relates directly to the amount of time a user needs to wait for the global minimum.

At the same time, the total number of successes does not deteriorate with the new trust-region algorithm compared to ALSO. We suspect the reason is that ALSO is a non-monotone method that continues to move the

center of the trust-region even during unsuccessful iterations. In this way it appears to be more suitable for global exploration. We are experimenting with a new non-monotone version of our method.

Comparing our results with AMBH, we conclude that, in general, we are more successful, but our algorithm seems to be less effective on Ackley and Levy. In particular, the number of local searches is in general larger. One reason for the better performances of AMBH is that the trust-region radius is chosen in such a way as to optimize the performances of MBH. Another reason is probably related to the different way of updating the radius: in AMBH the radius is updated by adding/subtracting a constant, whereas our algorithm multiplies/divides the trust-region radius. AMBH can find the optimal value of the trust-region radius for MBH more easily.

The detailed results of Tables B.7–B.16 are summarized in the performance profiles (see [17]) in Figures 5.2 and 5.3. The plots are generated by regarding every initial trust-region radius and every problem as a separate run. We form the ratio of the performance measure $\mathrm{perf}(s, r)$ of solver s on run r divided by the best performance for problem s, and take its base 2 logarithm:

$$\log_2 \left(\frac{\mathrm{perf}(s, r)}{\min\limits_s \mathrm{perf}(s, r)} \right).$$

By sorting these ratios in ascending order for every solver, the resulting plot can be interpreted as the probability distribution that solver s performs within a given multiple of the best solver.

Figure 5.2 compares the success rates of the four solvers. To apply the performance profile, we take the reciprocal of the success rates in Tables B.7–B.16 and set $1/0$ to 10^8. In this way, the data is consistent with the performance profile idea that smaller values are better. We can interpret the plots in Figure 5.2 as a probability distribution that a solver has a success rate at worst a factor 2^{-x} of the best solver. We observe that ALSO and TRF are more robust and have better success rates than do AMBH and MBH. TRF is slightly more robust than ALSO, a result that shows that embedding ALSO within a trust-region framework did not deteriorate robustness of the solver.

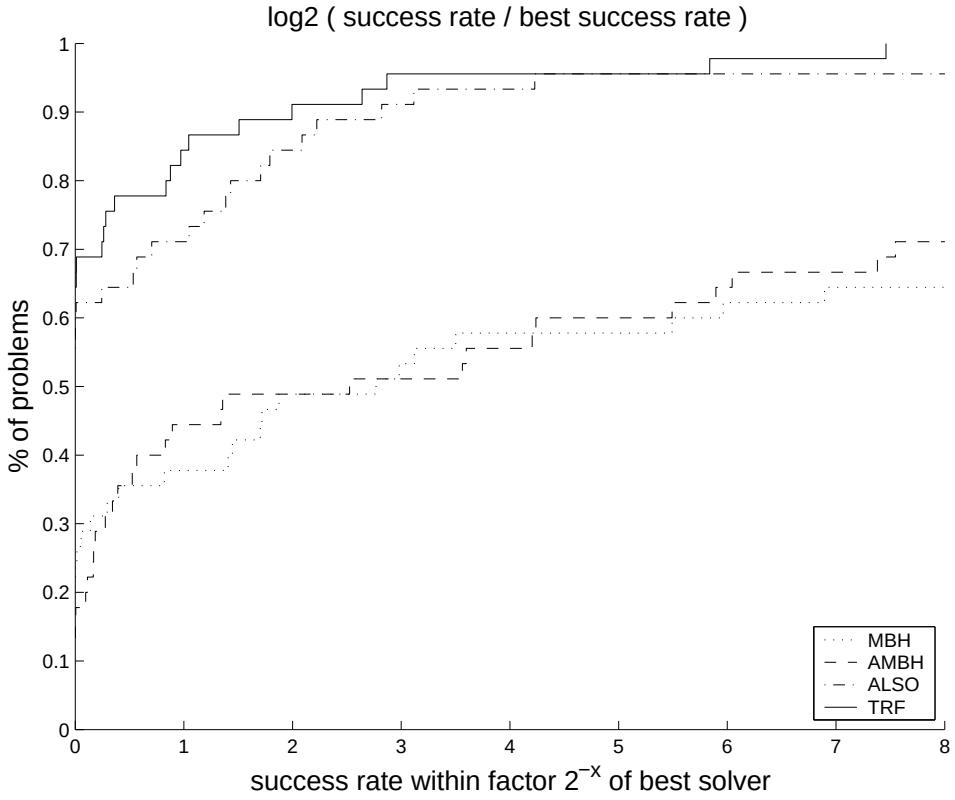

Figure 5.2. Performance profile of the success rate

In Figure 5.3 we compare the average number of local searches per successful solve. The plot represents a probability distribution that a solver solves a problem within a factor of at most 2^x of the fastest solve. The plot clearly shows that the new TRF outperforms all other solvers. We also observe that ALSO is faster than AMBH, which in turn is faster than MBH.

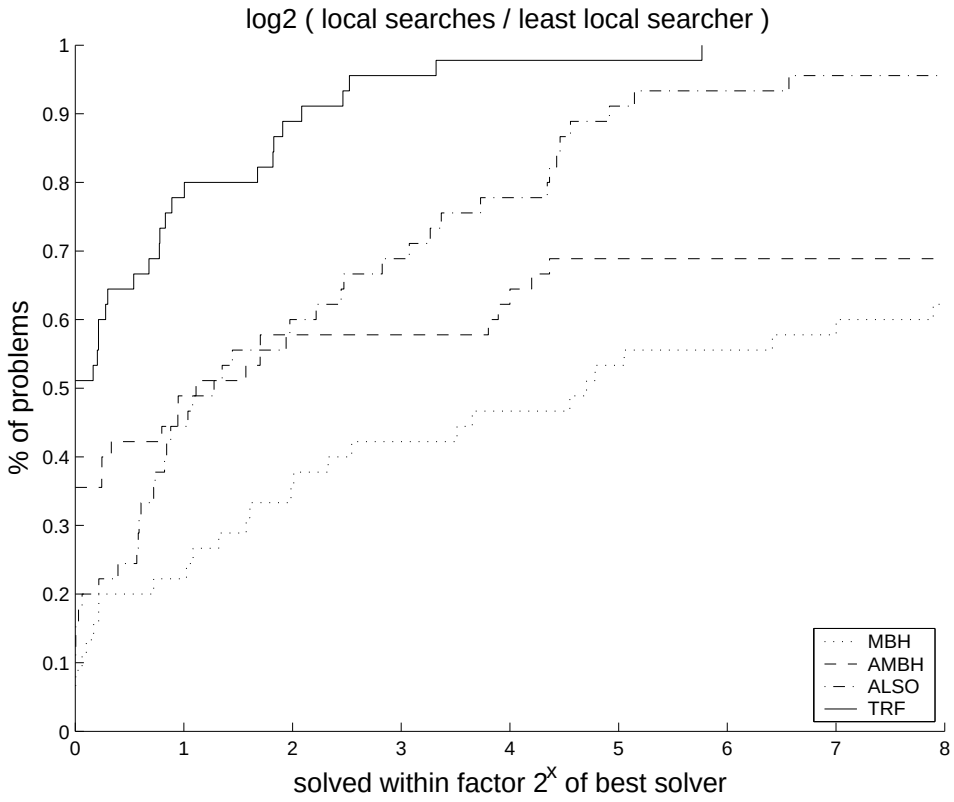

Figure 5.3. Performance profile of the average number of local searches

Conclusion

We proposed different algorithms aimed to solve difficult, large scale, global optimization problems in which the objective function has a funnel structure. In all of them the use of local search is a key element. In particular the use of local minimum values instead of the function values is very effective.

We proposed to embed a penalized energy function in Monotonic Basin Hopping for solving rigid protein docking. Although this is a very difficult problem for optimization and also from a modeling point of view, the preliminary results are very encouraging. To develop new methods we plan to explore different kind of models, that can give more freedom in choosing the test set, and in a parallel way to improve the methods using more information on the problem. To this purpose we developed strategies for working on the funnel structure. We proposed methods using smoothing techniques on the result of the local search instead of the original objective function. In this way with a soft smoothing is possible to reveal the funnel structure. The methods proposed were tested on several functions from the literature, and resulted more efficient and (maybe more important) extremely robust in solving large scale global optimization problems with huge numbers of local optima.

We plan to test the developed smoothing methods on the protein docking problem, both with and without the penalized energy function strategy.

Appendix A

Test functions

Unfortunately a reliable and stable set of test problems is still lacking for unconstrained large scale global optimization. Even in [23] the chapter on continuously differentiable unconstrained problems contains very few (and quite easy) general tests. For this reason the number of test we can use is not very large. We choose a few objective functions which share the following characteristics:

- they are essentially unconstrained (defined in a box)
- the dimension of the problems can be chosen
- the global optimum is known
- they possess a very high number of local optima

As a reference, we give the test problems with their relative boxes and the value of the global minimum:

Ackley [1]

$$Ack(x) = -20 \cdot exp(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi x_i)) \quad \text{(A.1)}$$

with $x_i \in [-32.768, 32.768]$, $f^\star = -20.0 - e$

Levy [32]

$$Levy(x) = 10\sin^2(\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2(1 + 10\sin^2(\pi x_{i+1})) + (x_n - 1)^2 \quad \text{(A.2)}$$

with $x_i \in [-10, 10]$, $f^\star = 0.0$

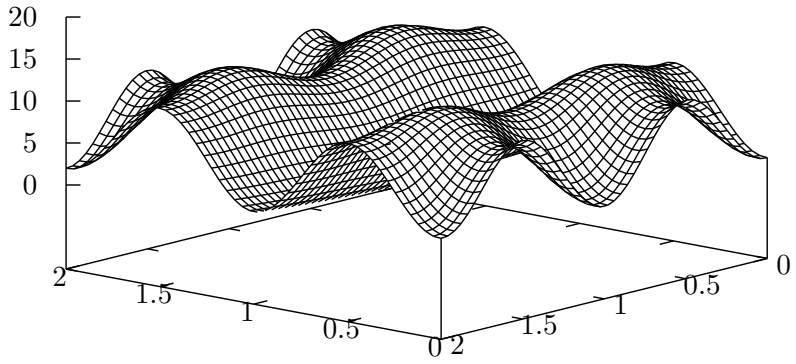

Figure A.1. Levy in two-dimensions with $x_i \in [0, 2]$

Rastrigin [46]

$$Ras(x) = 10n + \sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i) \tag{A.3}$$

with $x_i \in [-5.12, 5.12]$, $f^\star = 0.0$

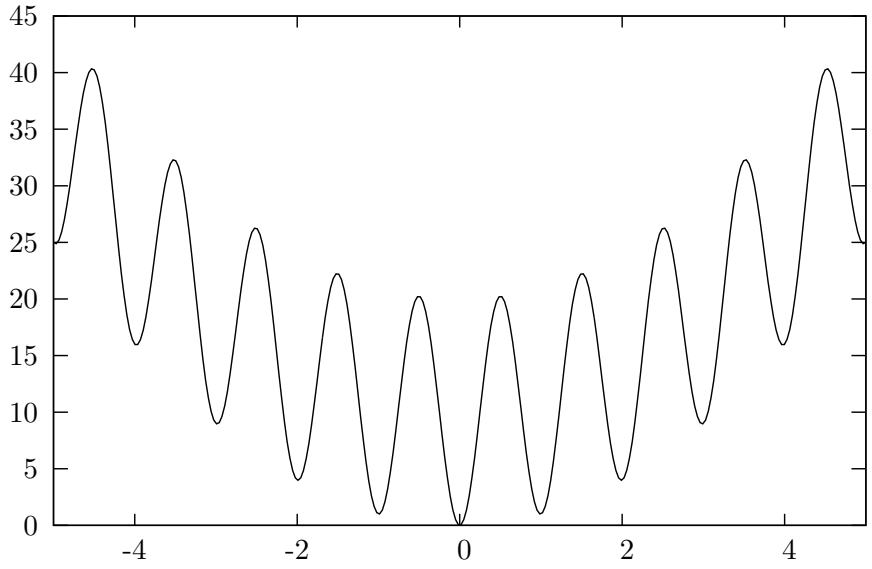

Figure A.2. Rastrigin in one-dimension

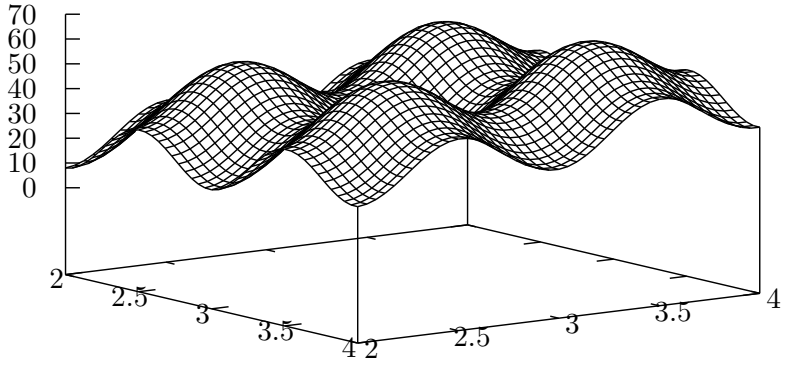

Figure A.3. Rastrigin in two-dimensions with $x_i \in [2, 4]$

Schwefel [43]

$$Sch(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{|x_i|}) \tag{A.4}$$

with $x_i \in [-500, 500]$, $f^\star = -418.9829n$

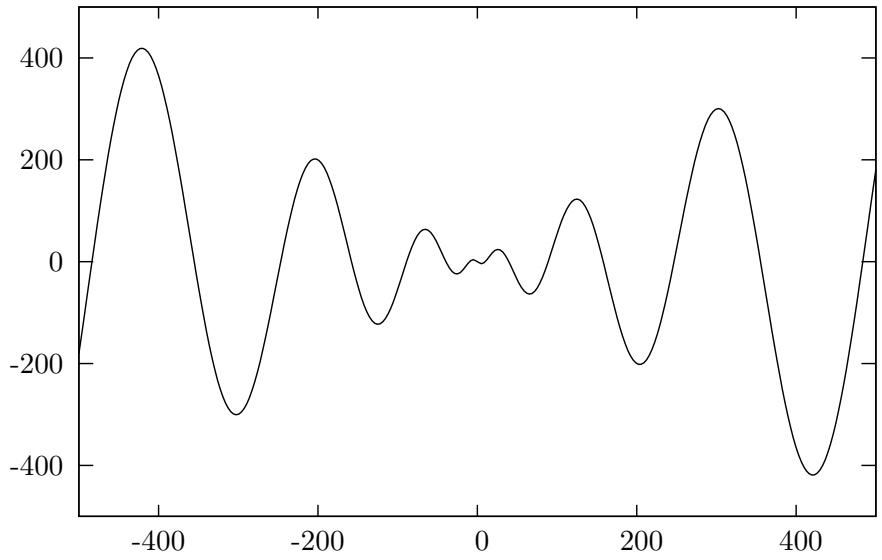

Figure A.4. Schwefel in one-dimension

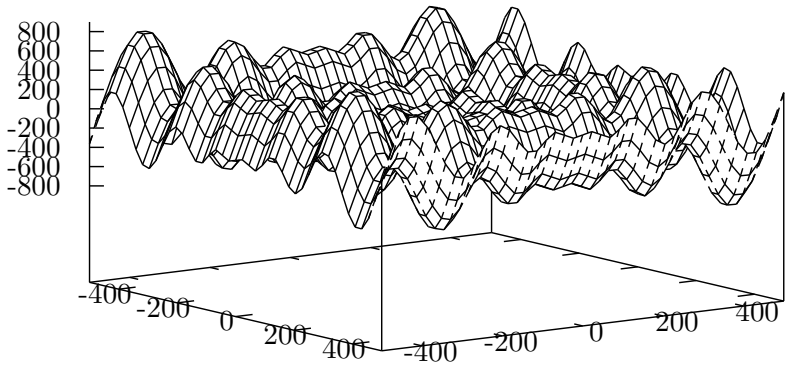

Figure A.5. Schwefel in two-dimensions

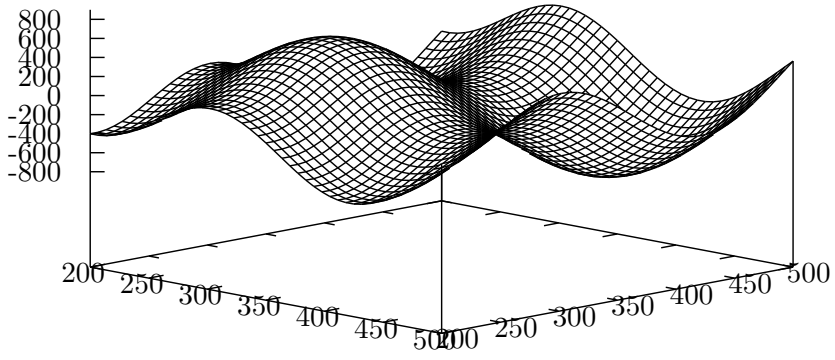

Figure A.6. Schwefel in two-dimensions with $x_i \in [200, 500]$

We also tested our method on an amplified version of the Rastrigin function:

$$AmplRas(x) = An + \sum_{i=1}^{n} x_i^2 - A\cos(2\pi x_i) \tag{A.5}$$

with $A = 100$ and with $A = 1000$ ($f^* = 0$); this function was chosen in order to test the sensitivity of the method to larger oscillations. And on a scaled version:

$$ScaledRas(x) = 10n + \sum_{i=1}^{n} (\alpha_i x_i)^2 - 10\cos(2\pi(\alpha_i x_i)) \tag{A.6}$$

where $\alpha_i = 1$ if $\lfloor i/10 \rfloor \equiv 0 (\mathrm{mod}\ 2)$, otherwise $\alpha_i = 2$ (i.e., $\alpha_i = 1$ for the first ten variables, then it is 2 for the next 10 and so on). This test function (still with $f^* = 0$) was introduced in order to check the behavior of the method in presence of asymmetric level sets. As a reference in Figure A.7 is represented the projection on $z - y$ plane of the scaled version in two dimensions with $\alpha_1 = 1$ and $\alpha_2 = 2$.

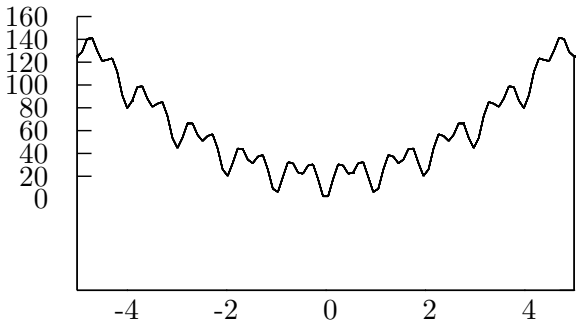

Figure A.7. Rastrigin Scaled in two-dimensions

Appendix B

Tables of numerical results

In this section we report some of the numerical results obtained for the methods developed for funnel functions described in Chapter 5. They are divided in two sections. The first for the results of the first version of the method (ALSO) compared with MBH. The second for the results of the adaptive version TRF, with the other methods tested. Results of ALSO, and MBH are reported also in the second section to help comparisons.

In each table we report the following columns: the radius Δ, the percentage of successes (by success we mean that the algorithm observed the global optimum at least once) and the average number of local searches per success, obtained by dividing the total number of local searches (except the last ones used for stopping) by the number of successes, if this number is positive. The symbol ∞ is used when the number of successes is zero. We choose not to count the last N local searches as these are constant for all the methods and, in some sense, they are just a waste, as they are used only to stop the algorithm. Results relative to each method are reported in the table with an heading with the corresponding name; results obtained

with our proposed methods appear also with the number of sample K, if is different from the dimension of the problem n.

B.1 ALSO

Results of ALSO with $K = n$ and $K = 2n$ compared with MBH.

Table B.1. Rastrigin

	Percentage of Successes			Average Local Searches		
Δ	MBH	ALSO $K = n$	ALSO $K = 2n$	MBH	ALSO $K = n$	ALSO $K = 2n$
$n = 20$						
1.0	0.0	100.0	100.0	∞	1777	3053
1.2	75.8	100.0	100.0	3020	1080	1433
1.4	99.8	100.0	100.0	510	475	468
1.6	98.2	98.2	100.0	596	514	423
1.8	32.1	73.9	98.6	3027	865	573
$n = 30$						
1.2	0.0	100.0	90.7	∞	3515	6866
1.4	8.9	100.0	100.0	66565	2470	3779
1.6	97.2	100.0	100.0	1084	961	984
1.8	98.3	93.2	100.0	760	889	698
2.0	28.9	52.0	95.8	4984	1839	988
$n = 50$						
1.8	0.0	99.0	96.6	∞	6060	10224
2.0	83.0	91.4	99.4	2444	2174	2011
2.2	94.5	58.0	96.7	1223	2249	1280
2.4	18.7	15.2	76.8	12010	10059	2257
2.6	0.0	2.7	37.7	∞	59198	4785

Table B.2. Levy

	Percentage of Successes			Average Local Searches		
Δ	MBH	ALSO $K = n$	ALSO $K = 2n$	MBH	ALSO $K = n$	ALSO $K = 2n$
$n = 20$						
0.8	30.4	100.0	95.7	2900	451	2025
1.0	98.9	100.0	100.0	1028	320	586
1.2	100.0	100.0	100.0	99	96	99
1.4	100.0	100.0	100.0	33	33	33
$n = 30$						
0.8	22.4	88.5	33.0	1032	2852	2179
1.0	34.6	98.7	60.3	2316	1606	1977
1.2	86.8	100.0	93.4	1789	716	1330
1.4	100.0	100.0	100.0	163	158	162
$n = 50$						
1.0	26.8	36.5	34.6	755	1412	831
1.2	30.4	47.9	45.1	1330	1539	1270
1.4	49.9	69.4	65.5	1887	1439	1459
1.6	95.5	98.6	96.9	1240	970	1103
1.8	100.0	100.0	100.0	146	144	146
2.0	100.0	100.0	100.0	47	47	47

Table B.3. Ackley

	Percentage of Successes			Average Local Searches		
Δ	MBH	ALSO $K = n$	ALSO $K = 2n$	MBH	ALSO $K = n$	ALSO $K = 2n$
$n = 20$						
1.0	0.0	100.0	100.0	∞	7844	13044
1.4	100.0	100.0	100.0	793	791	793
1.8	100.0	100.0	100.0	293	293	293
2.2	100.0	100.0	100.0	274	275	274
3.5	11.5	69.1	89.0	7926	1329	915
$n = 30$						
1.0	0.0	100.0	99.9	∞	20708	46390
1.4	99.9	100.0	100.0	22481	8385	13323
1.8	100.0	100.0	100.0	505	505	505
2.2	100.0	100.0	100.0	303	303	303
3.5	9.2	64.3	91.2	11328	1614	1009
$n = 50$						
1.4	0.0	99.8	0.0	∞	48433	∞
1.8	56.6	100.0	100.0	68965	19035	36007
2.2	100.0	100.0	100.0	601	601	601
3.5	100.0	100.0	100.0	282	288	283
3.9	81.5	99.9	99.8	1013	654	688

Table B.4. Schwefel

	Percentage of Successes			Average Local Searches		
Δ	MBH	ALSO $K = n$	ALSO $K = 2n$	MBH	ALSO $K = n$	ALSO $K = 2n$
$n = 5$						
80	0.1	2.8	1.4	305582	23714	35333
100	0.4	4.6	3.9	104415	6607	8062
120	3.0	14.4	10.3	13825	1924	2005
140	4.8	32.7	21.3	2688	1179	1186
160	4.4	49.8	31.5	1953	981	1077
180	6.4	64.3	39.3	1439	930	994
200	7.8	71.6	47.6	1577	893	900
220	9.8	77.4	59.1	1342	807	869
$n = 10$						
160	0.1	1.3	0.4	751440	22359	64460
180	0.2	2.5	0.7	154694	11493	31720
200	0.0	4.6	1.8	∞	7610	10503
220	0.0	6.6	1.1	∞	5767	16808
240	0.0	10.6	3.6	∞	4622	6419
260	0.5	14.9	4.4	19033	4364	6102
280	0.3	18.7	6.3	30444	4056	5425

Table B.5. AmpRast

	Percentage of Successes			Average Local Searches		
Δ	MBH	ALSO $K = n$	ALSO $K = 2n$	MBH	ALSO $K = n$	ALSO $K = 2n$
$n = 20$, A=100						
1.2	60.9	100.0	100.0	6450	1366	1994
1.4	99.7	100.0	100.0	758	645	663
1.6	93.0	98.4	100.0	834	614	508
1.8	21.0	73.6	97.4	5212	963	645
$n = 50, A = 100$						
2.0	44.9	87.8	99.7	9383	4002	3838
2.2	64.4	50.6	93.5	3315	3804	2098
$n = 50, A = 1000$						
1.8	0.0	98.1	98.0	∞	8168	15871
2.0	40.7	86.9	99.4	5698	4375	4211
2.2	58.7	53.0	92.9	7998	3852	2229
2.4	1.9	13.1	66.8	146646	14790	3239

Table B.6. ScaledRas

	Percentage of Successes			Average Local Searches		
Δ	MBH	ALSO $K = n$	ALSO $K = 2n$	MBH	ALSO $K = n$	ALSO $K = 2n$
$n = 20$						
0.6	0.0	57.2	17.9	∞	7644	35521
0.8	0.0	52.8	57.4	∞	4872	7079
1.0	0.0	0.3	0.1	∞	444273	1747470
1.2	0.0	0.6	0.1	∞	202571	1013720
1.4	0.0	3.1	0.1	∞	34111	481011
1.6	0.0	5.8	0.3	∞	20014	185277
1.8	0.0	10.4	0.6	∞	12741	123331
$n = 50$						
1.6	0.0	8.1	17.4	∞	87545	71151
1.8	0.0	1.6	7.0	∞	346525	126914
2.0	0.0	0.0	1.5	∞	∞	305760
2.2	0.0	0.0	0.0	∞	∞	∞
2.4	0.0	0.0	0.1	∞	∞	2392600
2.6	0.0	0.0	0.0	∞	∞	∞
$n = 50, \text{MaxNoImprove} = 2\,000$						
1.6	0.0	19.6	41.6	∞	39146	31626
1.8	0.0	4.2	15.4	∞	146978	61723
2.0	0.0	0.3	3.4	∞	1315497	153640
2.2	0.0	0.0	0.1	∞	∞	3600460
2.4	0.0	0.0	0.1	∞	∞	3023360
2.6	0.0	0.0	0.0	∞	∞	∞

B.2 TRF

Results of TRF compared with ALSO, MBH and AMBH. The number of samples K is equal to the dimension of the problem for both our methods.

Table B.7. Rastrigin, $n = 20$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
1.0	0.0	0.6	100.0	77.8	∞	188833	1776	652
1.2	75.8	89.1	100.0	83.4	3020	1960	1079	602
1.4	99.8	92.5	100.0	84.3	510	916	475	577
1.6	98.2	87.3	98.2	80.8	596	987	513	591
1.8	32.1	15.2	73.9	87.5	3027	7711	864	519

Table B.8. Rastrigin, $n = 50$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
1.8	0.0	0.0	99.0	48.0	∞	∞	6060	2952
2.0	83.0	49.2	91.4	46.6	2444	6459	2174	4363
2.2	94.5	74.5	58.0	51.5	1223	2644	2249	2173
2.4	18.7	4.2	15.2	49.6	12010	63048	10059	3940
2.6	0.0	0.0	2.7	50.6	∞	∞	59198	7026

Table B.9. Levy, $n = 20$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
0.8	30.4	76.2	100.0	99.2	2900	34	451	120
1.0	98.9	82.5	100.0	99.3	1028	31	320	110
1.2	100.0	93.5	100.0	100.0	99	25	96	80
1.4	100.0	99.8	100.0	100.0	33	20	33	32

Table B.10. Levy, $n = 50$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
1.0	26.8	38.4	36.5	98.3	755	60	1412	331
1.2	30.4	39.1	47.9	99.0	1330	51	1539	293
1.6	95.5	55.8	98.6	99.2	1240	45	970	169
2.0	100.0	97.1	100.0	100.0	47	22	47	45

Table B.11. Ackley, $n = 20$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
1.0	0.0	88.0	100.0	100.0	∞	381	7844	654
1.4	100.0	99.6	100.0	100.0	793	374	791	693
1.8	100.0	100.0	100.0	100.0	293	346	293	292
2.2	100.0	100.0	100.0	100.0	274	345	275	274
3.5	11.5	100.0	69.1	100.0	7926	338	1329	578

Table B.12. Ackley, $n = 50$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
1.4	0.0	67.5	99.8	100.0	∞	511	48433	2167
1.8	56.6	69.5	100.0	100.0	68965	538	19035	781
2.2	100.0	100.0	100.0	100.0	601	517	601	600
3.5	100.0	100.0	100.0	100.0	282	490	288	282
3.9	81.5	100.0	99.9	100.0	1013	498	654	613

Table B.13. Schwefel, n=5

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
80	0.1	0.2	2.8	11.9	305582	3500	23714	1076
140	4.8	2.7	32.7	11.5	2688	667	1179	748
160	4.4	2.7	49.8	12.5	1953	778	981	656
220	9.8	4.1	77.4	10.6	1342	537	807	623

Table B.14. Schwefel, n=10

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
160	0.1	0.1	1.3	4.5	751440	44000	22359	3156
220	0.0	0.1	6.6	3.7	∞	49000	5767	2378
280	0.3	0.1	18.7	3.0	30444	49000	4056	2667

Table B.15. Scaled Rastrigin, $n = 20$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
0.6	0.0	0.0	57.2	1.0	∞	∞	7644	76400
0.8	0.0	0.0	52.8	0.3	∞	∞	4873	265333
1.0	0.0	0.0	0.3	1.4	∞	∞	444273	46143
1.2	0.0	0.0	0.6	5.2	∞	∞	202572	9981
1.4	0.0	0.0	3.1	8.1	∞	∞	34112	6148
1.6	0.0	0.0	5.8	13.2	∞	∞	20014	4303

Table B.16. Scaled Rastrigin, $n = 50$

	Percentage of Successes				Average Local Searches			
Δ	MBH	AMBH	ALSO	TRF	MBH	AMBH	ALSO	TRF
1.6	0.0	0.0	8.1	12.0	∞	∞	87546	59117
1.8	0.0	0.0	1.6	11.3	∞	∞	346525	63522
2.2	0.0	0.0	0.0	12.3	∞	∞	∞	44325
2.6	0.0	0.0	0.0	11.8	∞	∞	∞	49415

Bibliography

[1] D. H. Ackley. *A Connectionist Machine for Genetic Hillclimbing*. Kluwer Acadmeic Publishers, Boston, 1987.

[2] B. Addis and S. Leyffer. A trust-region algorithm for global optimization. *Preprint ANL/MCS-P1190-0804, MCS Division, Argonne National Laboratory*, 2004.

[3] B. Addis, M. Locatelli, and F. Schoen. Local optima smoothing for global optimization. *to appear in Optimization Methods and Software*, 2003.

[4] B. Addis and F. Schoen. A randomized global optimization method for protein-protein docking. Technical Report DSI 4-2003, Dip. Sistemi e Informatica - Univ. Firenze, 2003.

[5] B. Addis and F. Schoen. Docking of atomic clusters through nonlinear optimization. *J. of Global Optimization*, 30:1–21, 2004.

[6] C.S. Adjiman, I.P. Andorulakis, and C. Floudas. A global optimization method, αBB, for general twice-differentiable constrained NLPs - II. Implementation and computational results. *Computers and Chemical Engineering*, 22:1159–1179, 1998.

[7] AMBER. Amber homepage. `http://www.amber.ucsf.edu/amber/amber.html`.

[8] J. Apostolakis, A. Pluckthun, and A. Caflisch. Docking small ligands in flexible binding sites. *Journal of Computational Chemistry*, 19:21–37, 1998.

[9] Boston University BioInformatics. Benchmark. `http://zlab.bu.edu/zdock/benchmark.shtml`.

[10] N.P. Boghossian, O. Kohlbacher, and H.-P. Lenhof. BALL - biochemical algorithms library. Technical Report MPI-I-99-1-002, Max-Planck-Institut für Informatik, 1999.

[11] J. D. Bryngelson, J. N. Onuchic, N. D. Socci, and P. G. Wolynes. Funnels, pathways and the energy landscape of protein folding: A synthesis. *Proteins: Structure, Function and Genetics*, 21, 1995.

[12] R. Chen, J. Mintseris, J. Janin, and Zhiping Weng. A protein-protein docking benchmar. *Proteins: Structure, Function, and Genetics*, 52:88–91, 2003.

[13] R. Chen and Z. Weng. Docking unbound proteins using shape complementarity, desolvation, and electrostatics. *Proteins: Structure, Function, and Genetic*, 47:281 – 294, 2002.

[14] A. R. Conn, N. Gould, and Ph. L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, 2000.

[15] D. J. Diller and C. L.M.J. Verlinde. A critical evaluation of several global optimization algorithms for the purpose of molecular docking. *Journal of Computational Chemistry*, 20:1740–1751, 1999.

[16] L. C. W. Dixon. Global optima without convexity. *Technical Report, Numerical Optimization*, 1978.

[17] E. D. Dolan and J. Moré. Benchmarking optimization software with cops. *Technical Report MCS-TM-246, Mathematics and Computer Science Division, Argonne National Laboratory*, November 2000.

[18] J P. K. Doye. The effect of compression on the global optimization of atomic clusters. *Phys. Rev. E*, pages 8753–8761, 2000.

[19] J. P. K. Doye. Physical perspectives on the global optimization of atomic clusters. In J. D. Pinter, editor, *Selected Case Studies in Global Optimization*, page in press. Kluwer, Dordrecht, 2002.

[20] J P. K. Doye, R. H. Leary, M. Locatelli, and F. Schoen. The global optimization of morse clusters by potential energy transformations. *to appear on Journal On Computing*, 2004.

[21] D. Duhovny, R. Nussinov, and H. J. Wolfson. Efficient unbound docking of rigid molecules. *Algorithms in Bioinformatics: Second International Workshop, WABI 2002*, 2452:185 – 200, 2003.

[22] J. Fernandez-Recio, M. Totrov, and R. Abagyan. Soft protein–protein docking in internal coordinates. *Protein Science*, 11:280–291, 2002.

[23] C. A. Floudas, P. M. Pardalos, C. Adjiman, W. Esposito, Z. pGümüs, S. Harding, J. Klepeis, C. Meyer, and C. Schweiger. *Handbook of Test Problems in Local and Global Optimization*, volume 33 of *Nonconvex Optimization and its Applications*. Kluwer Academic Publishers, Dordrecht, 1999.

[24] J. J. Gray, S. Moughon, C. Wang, O. Schueler-Furman, C. A. Rohl B. Kuhlman, and D. Baker. Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. *Journal of Molecular Biology*, 331:281–299, 2003.

[25] GROMOS. Gromos homepage. `http://igc.ethz.ch/gromos/welcome.html`.

[26] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[27] J. L. Klepeis, M. G. Ierapetritou, and C. A. Floudas. Protein folding and peptide docking: A molecular modeling and global optimization

approach. *Computers and Chemical Engineering*, 22 Suppl. 1:S3–S10, 1998.

[28] R. H. Leary. Global optimization on funneling landscapes. *Journal of Global Optimization*, 18(4):367–383, 2000.

[29] H.P. Lenhof. An algorithm for the protein docking problem. Technical report, MaxPlanckInstitut fur Informatik, 1995.

[30] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[31] C. Levinthal. How to fold graciously. *Mssbauer Spectroscopy in Biological Systems, Proceedings of a Meeting Held at Allerton House, Monticello, Illinois*, pages 22–24, 1969.

[32] A.V. Levy and A. Montalvo. The tunneling method for global optimization. *SIAM J. of Sci. and Stat. Comp.*, 1:15–29, 1985.

[33] D. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, B 45:503–528, 1989.

[34] M. Locatelli. Simulated annealing algorithms for continuous global optimization. *Handbook of Global Optimization II*, pages 179–230, 2002.

[35] M. Locatelli. On the multilevel structure of global optimization problems. *To appear in* Computational Optimization and Applications, 2003.

[36] M. Locatelli and F. Schoen. Fast global optimization of difficult Lennard-Jones clusters. *Computational Optimization and Applications*, 21(1):55–70, 2002.

[37] M. Locatelli and F. Schoen. Efficient algorithms for large scale global optimization: Lennard-Jones clusters. *Computational Optimization and Applications*, 26:173–190, 2003.

[38] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, and A.H. Teller. Equation of state calculations by fast computer machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[39] J. J. Moré and Z. Wu. Smoothing techniques for macromolecular global optimization. In G. Di Pillo and F. Gianessi, editors, *Nonlinear Optimization and Applications*, pages 297–312. Plenum Press, 1996.

[40] J. J. Moré and Z. Wu. Global continuation for distance geometry problems. *SIAM J. Optim.*, 7:814–836, 1997.

[41] G.M. Morris, D.S. Goodsell, R. Huey, W.E. Hart, S. Halliday, R. Belew, and A.J. Olson. *Autodock 3.05 - Automated Docking of Flexible Ligands to Receptors - User's Guide*. The Scripps Research Institute, 2001.

[42] F. Schoen. Two-phase methods for global optimization. In P.M. Pardalos and E. H. Romeijn, editors, *Handbook of Global Optimization Volume 2*, pages 151–178. Kluwer, Dordrecht, 2002.

[43] H. P. Schwefel. *Numerical Optimization of Computer Models*. J. Wiley & Sons, Chichester, 1981.

[44] C. S. Shao, R. H. Byrd, E. Eskow, and R. B. Schnabel. Global optimization for molecular clusters using a new smoothing approach. In Lorenz T. Biegler, Thomas F. Coleman, Andrew R. Conn, and Fadil N. Santosa, editors, *Large Scale Optimization with Applications: Part III: Molecular Structure and Optimization*, pages 163–199. Springer, New York, 1997.

[45] F. H. Stillinger. Exponential multiplicity of inherent structures. *Phys. Rev. E*, 59:48–51, 1999.

[46] A.A. Törn and A. Žilinskas. *Global Optimization*. Lecture Notes in Computer Sciences. Springer-Verlag, Berlin, 1989.

[47] M. Totrov and R. Abagyan. Flexible protein–ligand docking by global energy optimization in internal coordinates. *PROTEINS: Structure, Function, and Genetics*, Suppl. 1:215–220, 1997.

[48] A. Tovchigrechko and I. A. Vakser. How common is funnel-like energy landscape in protein-protein interactions? *Protein science*, 10:1572–1583, 2001.

[49] V. Černy. Thermodynamical approach to the travelling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45:41–51, 1985.

[50] 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*, 101:5111–5116, 1997.

[51] D. J. Wales, J. P. K. Doye, A. Dullweber, M. P. Hodges, F. Y. Naumkin, F. Calvo, J. Hernndez-Rojas, and T. F. Middleton. The cambridge cluster database. `http://brian.ch.cam.ac.uk/CCD.html`.