

A New Algorithm for Box-Constrained Global Optimization

S. Fanelli

Published online: 6 January 2011
© Springer Science+Business Media, LLC 2011

Abstract An important class of deterministic methods for global optimization is based on the theory of terminal attractors and repellers. Unfortunately, the utilization of scalar repellers is unsuitable, when the dimension n of the problem assumes values of operational interest. In previous papers the author et al. showed that BFSG-type methods, approximating the Hessian of twice continuously differentiable functions with a structured matrix, are very efficient to compute local minima, particularly in the secant case. On the other hand, the algorithms founded on the classical αBB technique are often ineffective for computational reasons. In order to increase the power of repellers in the tunneling phases, the utilization of repeller matrices with a proper structure is certainly promising and deserves investigation. In this work, it is shown that a BFGS-type method of low complexity, implemented in the local optimizations, can be effectively matched with proper repeller matrices in the tunneling phases. The novel algorithm $FB\alpha BB$, which can be applied in the frame of the αBB computational scheme, is very efficient in terms of Number of Functions Generations (NFG), Success Rates (SR) in the evaluation of the global minimum and Number of Local Searches (NLS).

Keywords Global optimization · BFGS methods · Tunneling techniques · αBB method · Box-constrained problems

Communicated by F. Zirilli.

The author wishes to thank the students Valeria Tozzi and Francesco Tudisco for their kind support in the implementation of Algorithm $FB\alpha BB$. The author is also indebted to an anonymous referee for some useful suggestions.

S. Fanelli (✉)
Università di Roma “Tor Vergata”, viale della Ricerca Scientifica, 00133 Roma, Italy
e-mail: fanelli@mat.uniroma2.it

1 Introduction

A class of remarkable methods for deterministic unconstrained global optimization is based upon the utilization of the *tunneling techniques*. By this approach the algorithm is composed of a sequence of cycles, where each cycle has two phases, i.e. a local optimization phase and a tunneling one. The main aim of these procedures is to build a monotone sequence of local minima (maxima), by determining a set of possible candidates for the global minimum (maximum). Classical methods of this type were described in [1–3].

In particular, the algorithm in [3], named *TRUST*, makes use of the theory of (non-Lipschitz) terminal repellers, by introducing a special tunneling function, which is based on two parts, i.e. a sub-energy function and a repeller one.

In the one-dimensional case, the convergence of *TRUST* to the global optimum is theoretically guaranteed. In the general case, *TRUST* is not able to compute the optimal solution [4], even if it can be fruitfully applied to some classes of functions (see e.g. [5]). Further experiments concerning the optimal learning in MLP-networks were presented in [6].

In [7, 8], it was shown that a proper random version of *TRUST*, called *RTA*, can have satisfactory performances for problems involving up to 20 variables and hundreds of local minima. In order to increase the effectiveness of the global optimization algorithm, one can improve both the efficiency of the local optimization phases and the power of the tunneling criteria. The former aim was deeply studied by the author et al. in the papers [9–12], while the latter scope was investigated at a preliminary level in [6, 13].

From one hand, in fact, the BFGS-type methods introduced in [9, 12] are able to obtain satisfactory results in the determination of local minima (maxima) for problems involving thousands of variables. On the other hand, by assuming a growth condition on the objective function, the αBB approach [14, 15] can be suitably extended to unconstrained global optimization, thereby obtaining the following advantages in the computational procedure:

1. superimposing the αBB approach in the tunneling phases.
2. extending BFGS-type methods to a global optimization scheme.

In this paper, it is shown that, for twice continuously differentiable functions, the utilization of structured matrices can play an important role both in the local optimizations, by approximating the Hessian of the objective function, and in the tunneling phases, by determining a set of suitable *repeller matrices*. As a matter of fact, in the classical and cited *TRUST* algorithm, the performances of the repeller function utilized in the tunneling phase are strictly dependent on a *scalar term* $k > 0$, named the *power of the repeller*. When the dimension n of the problem assumes operational values, the use of a scalar repeller is clearly inadequate.

In the present work, we suggest the use of a repeller structured matrix, based on the sum of a diagonal matrix and a low rank one. In this way, the application of a BFGS-type method can be easily extended to the tunneling phases and hence to the whole global optimization scheme, by maintaining the well known efficiency of the fast transform-based algorithms utilized in the local searches. More precisely, the preliminary numerical experiences shown in this paper, which are related to the simple

case of a rank 2 correction (algorithm $FB\alpha BB$) are encouraging from a computational complexity point of view, particularly in terms of NFG and/or NLS, thereby showing the usefulness of repeller matrices.

It is important to point out that the utilization of tight convex underestimators (see e.g. [16–19]) can furtherly improve the performances of the method presented in the present work. This will be shown in a future paper.

2 Some Preliminary Extensions of Local BFGS-Type Algorithms

Let us consider the following unconstrained problem:

$$\min F(\mathbf{x}), \quad \mathbf{x} \in R^n. \tag{1}$$

Given an approximation $B^{(k)}$ of $\nabla^2 F(\mathbf{x}^{(k)})$, let us define the matrix $\mathcal{L}_{B^{(k)}}$:

$$\begin{aligned} \|\mathcal{L}_{B^{(k)}} - B^{(k)}\|_F &= \min_{X \in \mathcal{L}} \|X - B^{(k)}\|_F, \\ \|\cdot\|_F &= \text{Frob. norm}, \quad \mathcal{L} = \text{algebra} \subset \mathbb{C}^{n \times n}. \end{aligned}$$

One can define descent methods \mathcal{LQN} [9]:

$$\mathbf{x}_0 \in R^n, \quad \mathbf{d}^{(0)} = -\nabla F(\mathbf{x}^{(0)}).$$

For $k = 0, 1, \dots$

$$\begin{cases} \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \lambda_k \mathbf{d}^{(k)}, & \lambda_k > 0, \\ B^{(k+1)} = \varphi(\mathcal{L}_{B^{(k)}}, \underbrace{\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}}_{\mathbf{s}_k}, \underbrace{\nabla F(\mathbf{x}^{(k+1)}) - \nabla F(\mathbf{x}^{(k)})}_{\mathbf{y}_k}), \\ \mathbf{d}^{(k+1)} = -B^{(k+1)^{-1}} \nabla F(\mathbf{x}^{(k+1)}). \end{cases}$$

The classical BFGS method [20] and the more recent minimization methods introduced in [9, 10, 12] are examples of \mathcal{LQN} algorithms, being $\mathcal{L} = \mathbb{C}^{n \times n}$, $\mathcal{L} = \{\alpha I\}$, $\{\text{Hartley-type}\}$, \mathcal{L}^k . The step λ_k is determined, such that:

$$\mathbf{s}_k^T \mathbf{y}_k > 0, \quad F(\mathbf{x}^{(k+1)}) = F(\mathbf{x}^{(k)} + \lambda_k \mathbf{d}^{(k)}) < F(\mathbf{x}^{(k)}).$$

The updating function φ in $B^{(k+1)} = \varphi(\mathcal{L}_{B^{(k)}}, \mathbf{s}_k, \mathbf{y}_k)$ is:

$$\varphi(\square, \mathbf{s}, \mathbf{y}) = \square + \frac{1}{\mathbf{y}^T \mathbf{s}} \mathbf{y} \mathbf{y}^T - \frac{1}{\mathbf{s}^T \square \mathbf{s}} \square \mathbf{s} \mathbf{s}^T \square.$$

The choice of λ_k and the properties of φ and $\mathcal{L}_{B^{(k)}}$ imply:

$$\begin{cases} B^{(k+1)} \text{ inherits positive definiteness from } B^{(k)}. \\ B^{(k+1)} \mathbf{s}_k = \mathbf{y}_k \implies \mathcal{LQN} \text{ are secant algorithms.} \\ \text{The structured space } \mathcal{L} \implies \mathcal{LQN} \text{ has a low complexity.} \\ \text{Every iteration of } \mathcal{LQN} \text{ has a cost } O(n \log n). \end{cases}$$

One can prove the following global convergence theorem [13]:

Theorem 2.1 Consider Problem (1), where $F(\mathbf{x}) \in C^2(\mathbb{R}^n)$.

Let F_{min} be the value of the optimal solution. Assume that:

$$\begin{aligned} \forall \epsilon_a \in \mathfrak{R}^+, \exists \epsilon_s \in \mathfrak{R}^+ : \quad & \|\nabla F(\mathbf{x}^{(k)})\| > \epsilon_s \\ \text{except for } k : \quad & F(\mathbf{x}^{(k)}) - F_{min} < \epsilon_a. \end{aligned} \tag{2}$$

If in an iterative scheme of BFGS-type:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \lambda_k B^{(k)-1} \nabla F(\mathbf{x}^{(k)}) \quad \left(B^{(k)} = \varphi(\tilde{B}^{(k-1)}, \dots), \forall k \right),$$

the following conditions are satisfied:

$$\frac{\|\nabla F(\mathbf{x}^{(k+1)}) - \nabla F(\mathbf{x}^{(k)})\|^2}{(\nabla F(\mathbf{x}^{(k+1)}) - \nabla F(\mathbf{x}^{(k)}))^T \lambda_k \mathbf{d}^{(k)}} = \frac{\|\mathbf{y}_k\|^2}{\mathbf{y}_k^T \mathbf{s}_k} \leq M, \tag{3}$$

$$\text{cond}(B^{(k)}) \leq N, \tag{4}$$

then:

$$\forall \epsilon_a \in \mathfrak{R}^+, \exists k^{**} : \forall k > k^{**} \quad F(\mathbf{x}^{(k)}) - F_{min} < \epsilon_a.$$

Theorem 2.1 points out three conditions for a global optimization BFGS-type method, i.e.:

Condition (2) assumes an *optimal matching* between the BFGS-type algorithm and the function F [6].

Condition (3) is a sort of *weak discrete convexity assumption* introduced in [21].

Condition (4) can be easily satisfied, by modifying the matrices $B^{(k)}$ during the optimization process, because *every descent direction is associated to a positive definite (p.d.) matrix with a well defined spectral structure*.

3 A Global Optimization Theorem for a BFGS-Type Method

Let us now consider the classical “box-constrained” problem:

$$\min F(\mathbf{x}), \quad \mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U. \tag{5}$$

Let $\mathbf{x}_{c(m)}^L \leq \mathbf{x}_{c(m)} \leq \mathbf{x}_{c(m)}^U$ denote the *current box* at iteration m .

Set:

$$\alpha_{\mathbf{x}_{c(m)}} = \max \left\{ 0, -\frac{1}{2} \min_i \lambda_i (\nabla^2 F(\mathbf{x}_{c(m)})) \right\}, \tag{6}$$

$$L_{c(m)}(\mathbf{x}_{c(m)}) = F(\mathbf{x}_{c(m)}) + \alpha_{\mathbf{x}_{c(m)}} (\mathbf{x}_{c(m)}^L - \mathbf{x}_{c(m)}) (\mathbf{x}_{c(m)}^U - \mathbf{x}_{c(m)}). \tag{7}$$

The following global convergence theorem holds ([14, 15]):

Theorem 3.1 Consider Problem (5) and assume $F(\mathbf{x}) \in C^2$. These hypotheses imply:

$$\text{cond}(\nabla^2 F(\mathbf{x})) \leq c, \quad \forall m \exists \alpha_m^* = \max_{\mathbf{x}_{c(m)}} \alpha_{\mathbf{x}_{c(m)}}.$$

Set:

$$F_{c(m)}^L = \inf_{\mathbf{x}_{c(m)}} L_{c(m)}(\mathbf{x}_{c(m)}), \quad F_{c(m)}^U = F((\mathbf{x}_{c(m)}^L + \mathbf{x}_{c(m)}^U)/2).$$

then, it follows $\forall m$:

$$F_{c(m)}^L \leq F_{c(m+1)}^L \leq \min_{\mathbf{x}_{c(m+1)}} F(\mathbf{x}_{c(m+1)}) \equiv \min_{\mathbf{x}} F(\mathbf{x}), \tag{8}$$

$$F_{c(m)}^U \geq F_{c(m+1)}^U \geq \min_{\mathbf{x}} F(\mathbf{x}) \geq F_{c(m)}^L. \tag{9}$$

Moreover, $\forall \epsilon_a > 0, \exists m^* : \forall m \geq m^*$:

$$\begin{cases} F_{c(m)}^U - F_{c(m)}^L < \epsilon_a, \\ \|\mathbf{x}_{c(m)}^U - \mathbf{x}_{c(m)}^L\|_2 \leq \sqrt{4\epsilon_a/c_2}. \end{cases} \tag{10}$$

Theorem 3.1 can be immediately extended by using a growth condition on the function $F(\mathbf{x})$. It follows in fact:

Corollary 3.1 Given $F(\mathbf{x}) \in C^2(R^n)$:

$$\lim_{\|\mathbf{x}\| \rightarrow \infty} F(\mathbf{x}) = +\infty. \tag{11}$$

Equation (11) implies:

$$\exists K_0 : \min_{\mathbf{x} \in R^n} F(\mathbf{x}) \equiv \min_{\|\mathbf{x}\| \leq K_0} F(\mathbf{x}),$$

$$\|\nabla^2 F(\mathbf{x})\| \leq c_1, \quad \|\mathbf{x}\| \leq K_0.$$

Assume:

$$\|\nabla^2 F(\mathbf{x})^{-1}\| \leq c_2 \quad \text{and hence} \quad \text{cond}(\nabla^2 F(\mathbf{x})) \leq c_1 c_2. \tag{12}$$

Then, the convergence results proved for (5) can be applied to (1)

We can fruitfully combine the results of Theorem 2.1 and Theorem 3.1, by proving the following:

Theorem 3.2 Consider Problem (5) and assume $F(\mathbf{x}) \in C^2(R^n)$.

If in a BFGS-type iterative scheme:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mu_k \mathcal{B}^{(k)-1} \nabla F(\mathbf{x}^{(k)})$$

the following conditions are satisfied:

$$\mathbf{x}^L \leq \mathbf{x}^{(k)} \leq \mathbf{x}^U, \quad \forall k, \tag{13}$$

$$\text{cond}(\mathcal{B}^{(k)}) \leq N, \tag{14}$$

$$\frac{\|\nabla F(\mathbf{x}^{(k+1)}) - \nabla F(\mathbf{x}^{(k)})\|^2}{(\nabla F(\mathbf{x}^{(k+1)}) - \nabla F(\mathbf{x}^{(k)}))^T \lambda_k \mathbf{d}^{(k)}} = \frac{\|\mathbf{y}_k\|^2}{\mathbf{y}_k^T \mathbf{s}_k} \leq M. \tag{15}$$

then the algorithm is convergent to the optimal solution of (5).

Proof By the assumptions it follows: $\text{cond}(\nabla^2 F(\mathbf{x})) \leq c$.

Hence, by (6) we have $\forall m$:

$$\exists \alpha_m^* = \max_{\mathbf{x}_{c(m)}} \alpha_{\mathbf{x}_{c(m)}}. \tag{16}$$

Set:

$$\tilde{L}_{c(m)}(\mathbf{x}_{c(m)}) = F(\mathbf{x}_{c(m)}) + \alpha_m^* (\mathbf{x}_{c(m)}^L - \mathbf{x}_{c(m)})(\mathbf{x}_{c(m)}^U - \mathbf{x}_{c(m)}). \tag{17}$$

Therefore, by (16) and (17) $\forall m$:

$$\begin{cases} L_{c(m)}(\mathbf{x}_{c(m)}) \geq \tilde{L}_{c(m)}(\mathbf{x}_{c(m)}), & \forall \mathbf{x}_{c(m)}, \\ \tilde{L}_{c(m)}(\mathbf{x}_{c(m)}) \text{ convex } \forall \mathbf{x}_{c(m)}. \end{cases}$$

So:

$$F_{c(m)}^L = \inf_{\mathbf{x}_{c(m)}} L_{c(m)}(\mathbf{x}_{c(m)}) = \min_{\mathbf{x}_{c(m)}} \tilde{L}_{c(m)}(\mathbf{x}_{c(m)}), \forall m. \tag{18}$$

Let $\tilde{\mathbf{x}}_{c(m)}^{(k_m)}$ be a local minimum in the box $c(m)$. If $\mathbf{x}_{c(m+1)}^L \leq \tilde{\mathbf{x}}_{c(m)}^{(k_m)} \leq \mathbf{x}_{c(m+1)}^U$ and $F((\mathbf{x}_{c(m+1)}^L + \mathbf{x}_{c(m+1)}^U)/2) \geq F(\tilde{\mathbf{x}}_{c(m)}^{(k_m)})$, then define:

$$F_{c(m+1)}^U = F(\tilde{\mathbf{x}}_{c(m)}^{(k_m)}). \tag{19}$$

Else, set:

$$\begin{cases} \mathbf{x}_{c(m+1)}^{(0)} = (\mathbf{x}_{c(m+1)}^L + \mathbf{x}_{c(m+1)}^U)/2, \\ F_{c(m+1)}^U = F(\mathbf{x}_{c(m+1)}^{(0)}) \end{cases} \tag{20}$$

being $\mathbf{x}_{c(m+1)}^{(0)}$ a local minimum evaluated by the starting point $\mathbf{x}_{c(m+1)}^{(0)}$ and contained in the box $c(m + 1)$. Since the assumptions of Theorem 3.1 are satisfied, by the results of [13] (see Theorem 2 and Corollary 2), it follows that (14), (15) imply that $\forall \epsilon_b > 0, \exists \{\tilde{\mathbf{x}}_{c(m_i)}^{(k_{m_i})}\}$:

$$\begin{cases} F_{c(m_i)}^U \geq F_{c(m_{i+1})}^U, \\ \|\nabla F(\tilde{\mathbf{x}}_{c(m_i)}^{(k_{m_i})})\| < \epsilon_b. \end{cases} \tag{21}$$

Applying Theorem 3.1, by the inequalities (8) and (21) and by setting: $\epsilon = \max\{\epsilon_a, \epsilon_b\}$, we have that $\forall \epsilon > 0, \exists \{\mathbf{x}_{c(m_i)}^{(\tilde{k}_{m_i})}\}$:

$$\begin{cases} \|\nabla F(\mathbf{x}_{c(m_i)}^{(\tilde{k}_{m_i})})\|_2 < \epsilon, \\ \|\mathbf{x}_{c(m_i)}^U - \mathbf{x}_{c(m_i)}^L\|_2 \leq \sqrt{4\epsilon/c_2}, \\ F(\mathbf{x}_{c(m_i)}^{(\tilde{k}_{m_i})}) - F_{min} \leq F_{c(m_i)}^U - F_{c(m_i)}^L < \epsilon. \end{cases}$$

This completes the proof. □

Remark 3.1 Although the local minimization phases are performed effectively by the BFGS-type algorithm, the convergence of the method to the global minimum is usually very slow by the very nature of the αBB approach. In particular, the number of the upper bounds $F_{c(m_i)}^U$ and the corresponding boxes m_i , requested to obtain a satisfactory approximation can be unacceptable from a computational point of view. In order to overcome this problem, the fast determination of “good” local minima is essential. Hence, it is quite important to superimpose in the method efficient “tunneling phases” to avoid the unfair entrapment in a “bad” local minimum, i.e. when the condition:

$$F_{c(m+1)}^U = F(\mathbf{x}_{c(m+1)}^{(\tilde{k}_{m+1})}) = F(\mathbf{x}_{c(m+1)}^{(\tilde{k}_m)}) = F_{c(m)}^U \gg F_{min}$$

is verified for several iterations. For this purpose, the *power of the repellers*, utilized in the tunneling phases, plays a crucial role. The classical and well known use of scalar repellers is often unsuitable, when the dimension n of the problem assumes values of operational interest. In next paragraph, we will deal with this problem by a novel approach.

4 Matrix Structures in a Global Minimization Scheme

Let $\mathbf{x}^{(\tilde{k})}$ be an approximation of a local minimizer for a function $F(\mathbf{x}) \in C^1$.

A matrix $\mathcal{A}^{(\tilde{k})}$ is called a *repeller matrix* for $\mathbf{x}^{(\tilde{k})}$ if $\exists \hat{\mathbf{x}}$:

$$\begin{cases} \hat{\mathbf{x}} = \mathbf{x}^{(\tilde{k})} - \mathcal{A}^{(\tilde{k})} \nabla F(\mathbf{x}^{(\tilde{k})}), \\ F(\hat{\mathbf{x}}) < F(\mathbf{x}^{(\tilde{k})}). \end{cases} \tag{22}$$

In order to increase the power of a tunneling phase, the method tries to determine a repeller matrix $\mathcal{A}_{c(m)}^{(\tilde{k}_m)}$ in every box $c(m)$ and for any given computed local minimizer $\mathbf{x}_{c(m)}^{(\tilde{k}_m)}$. By the results of [22] (see Sect. 2), $\mathcal{A}_{c(m)}^{(\tilde{k}_m)}$ can be approximated in the following way:

$$\mathcal{A}_{c(m)}^{(\tilde{k}_m)} \approx \lambda_{c(m)}^{(\tilde{k}_m)} I + (I/\mu + R_{c(m)})^{-1}, \quad 2 \leq \text{rank}(R_{c(m)}) \leq 4$$

being, by terminal attractors theory, $\lambda_{c(m)}^{(\tilde{k}_m)}$ the *maximal scalar repeller* ([6], see Definition 1), i.e.:

$$\lambda_{c(m)}^{(\tilde{k}_m)} = \frac{\epsilon}{\|\nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})\|^2}, \quad \|\nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})\| \ll \sqrt{\epsilon} \tag{23}$$

and $R_{c(m)}$ with the following structure:

$$\begin{cases} R_{c(m)} = \mu_{c(m)}^{(1)} P_{c(m)} P_{c(m)}^T + \mu_{c(m)}^{(2)} q_{c(m)} q_{c(m)}^T + \mu_{c(m)}^{(3)} P_{c(m)} r_{c(m)}^T \\ \quad + \mu_{c(m)}^{(4)} r_{c(m)} q_{c(m)}^T, \\ \mu_{c(m)}^{(1)}, \mu_{c(m)}^{(2)}, \mu_{c(m)}^{(3)}, \mu_{c(m)}^{(4)} \text{ scalars,} \\ P_{c(m)}, q_{c(m)}, r_{c(m)} \text{ suitable vectors.} \end{cases}$$

The main steps of each optimization cycle can be stated as follows:

1. Compute a local minimum $\mathbf{x}_{c(m)}^{(\tilde{k}_m)}$ in the box $c(m)$.
2. Apply a scalar repeller $\lambda_{c(m)}^{(\tilde{k}_m)}$ and compute $\mathbf{x}_{c(m)}^{(\tilde{k}_{m+1})}$.
3. Approximate $\mathcal{A}_{c(m)}^{(\tilde{k}_m)}$ with a $R_{c(m)}$ correction, $\text{rank}(R_{c(m)}) = 2$.
4. Compute $\mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}$.
5. If $F(\mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}) < F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})$,
 set $\mathbf{x}_{c(m)}^{(0)} = \mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}$ and start a new local search in $c(m)$.
6. Else
 Approximate $\mathcal{A}_{c(m)}^{(\tilde{k}_m)}$ with a $R_{c(m)}$ correction, $\text{rank}(R_{c(m)}) = 3, 4$.
7. Repeat 4. and 5.
8. Set $m = m + 1$ and go to 1.

The simplest tunneling scheme is to deal with the $\text{rank}(R_{c(m)}) = 2$ correction case only. This leads to the following:

Algorithm $FB\alpha BB$

Step I

Compute a local minimum $\mathbf{x}_{c(m)}^{(\tilde{k}_m)}$ in the box $c(m)$.

Step II

Compute: $\mathbf{x}_{c(m)}^{(\tilde{k}_{m+1})} = \mathbf{x}_{c(m)}^{(\tilde{k}_m)} - \lambda_{c(m)}^{(\tilde{k}_m)} \nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})$, $\lambda_{c(m)}^{(\tilde{k}_m)} = \epsilon_a / \|\nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})\|^2$.

Step III

Define $R_{c(m)}^{(\tilde{k}_m)}(\mu)$, $\mu > 0$:

$$R_{c(m)}^{(\tilde{k}_m)}(\mu) = \frac{q_{c(m)} q_{c(m)}^T}{q_{c(m)}^T P_{c(m)}} - (I/\mu) \frac{P_{c(m)} P_{c(m)}^T}{P_{c(m)}^T P_{c(m)}}$$

being: $p_{c(m)} = \mathbf{x}_{c(m)}^{(\tilde{k}_{m+1})} - \mathbf{x}_{c(m)}^{(\tilde{k}_m)}$, $q_{c(m)} = \nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_{m+1})}) - \nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})$.

By applying Sherman-Morrison-Woodbury formula, it follows:

$$\begin{cases} (I/\mu + R_{c(m)}^{(\tilde{k}_m)}(\mu))^{-1} \\ = \mu I + (1 + \mu \frac{q_{c(m)}^T q_{c(m)}}{q_{c(m)}^T p_{c(m)}}) \frac{p_{c(m)} p_{c(m)}^T}{q_{c(m)} p_{c(m)}} - \mu (\frac{p_{c(m)} q_{c(m)}^T + q_{c(m)} p_{c(m)}^T}{q_{c(m)} p_{c(m)}}), \\ \mu > 0. \end{cases}$$

Thus, we obtain a *memoryless updating formula*.

Step IV

Define $\mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}$ in the following way:

$$\begin{aligned} \mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})} &= \mathbf{x}_{c(m)}^{(\tilde{k}_{m+1})} - \left(I/\mu_0 + R_{c(m)}^{(\tilde{k}_m)}(\mu_0) \right)^{-1} \nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)}), \\ F(\mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}) &= \min_{\mu} F \left[\mathbf{x}_{c(m)}^{(\tilde{k}_{m+1})} - \left(I/\mu + R_{c(m)}^{(\tilde{k}_m)}(\mu) \right)^{-1} \nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)}) \right]. \end{aligned}$$

Therefore:

$$\mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})} = \mathbf{x}_{c(m)}^{(\tilde{k}_m)} - \left[\lambda_{c(m)}^{(\tilde{k}_m)} I + \left(I/\mu_0 + R_{c(m)}^{(\tilde{k}_m)}(\mu_0) \right)^{-1} \right] \nabla F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)}).$$

Step V

If:

$$F(\mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}) < F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})$$

Define: $\mathbf{x}_{c(m)}^{(0)} = \mathbf{x}_{c(m)}^{(\tilde{k}_{m+2})}$ and go to Step I.

Else

Step VI

Define the new box $\mathbf{x}_{c(m+1)}^L \leq \mathbf{x}_{c(m+1)} \leq \mathbf{x}_{c(m+1)}^U$.

Start the next local minimization by evaluating:

$$F_{c(m+1)}^L = \min_{\mathbf{x}_{c(m+1)}} L_{c(m+1)}(\mathbf{x}_{c(m+1)})$$

if $\mathbf{x}_{c(m+1)}^L \leq \mathbf{x}_{c(m)}^{(\tilde{k}_m)} \leq \mathbf{x}_{c(m+1)}^U$, set:

$$F_{c(m+1)}^U = \min \{ F((\mathbf{x}_{c(m+1)}^L + \mathbf{x}_{c(m+1)}^U)/2), F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)}) \}.$$

Else, set:

$$F_{c(m+1)}^U = F((\mathbf{x}_{c(m+1)}^L + \mathbf{x}_{c(m+1)}^U)/2).$$

It is important to point out that if $\mathbf{x}_{c(m)}^{(\tilde{k}_m)}$ is not in the box $c(m + 1)$, the latter point cannot be the global minimum of $F(\mathbf{x})$.

Step VII

If

$$F_{c(m+1)}^U = F((\mathbf{x}_{c(m+1)}^L + \mathbf{x}_{c(m+1)}^U)/2),$$

- Define $\mathbf{x}_{c(m+1)}^{(0)} = F_{c(m+1)}^U$.
- Set $m = m + 1$ and go to Step I, i.e. start a new local search in $c(m + 1)$.
- Perform the corresponding tunneling phase.

Else

- Repeat Step VI in the box $c(m + 2)$.

We show now that under suitable conditions the convergence of Algorithm $FB\alpha BB$ is guaranteed by Theorem 3.2.

Theorem 4.1 Consider Problem (5) with $F(\mathbf{x}) \in C^2$. Moreover, assume that $F(\mathbf{x})$ has a finite number P of local minima if $\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U$. Then, algorithm $FB\alpha BB$ satisfies (13), (14) and (15) and therefore is convergent by Theorem 3.2.

Proof By the properties of αBB computational scheme it follows immediately that (13) holds. In order to verify that (14) is satisfied, it is sufficient to observe that the matrices $\mathcal{B}^{(k)}$ can be adaptively modified in the practical implementation of the algorithm by the very nature of the BFGS-type method because every *p.d.* matrix $\mathcal{B}^{(k)}$ can be associated to a descent direction by implementing a restarting procedure (see [13], Theorem 3).

Finally, let us prove (15). Since $F(\mathbf{x}) \in C^2$ and has a finite number P of local minima, by Theorem 3.1 (10) is satisfied for a suitable m^* . Furthermore, $FB\alpha BB$ evaluates by definition any local minimum just a single time in each box $c(m)$. Hence, by (10) the algorithm computes the local minima at most a finite number of times $S_i \geq 0, i = 1, 2, \dots, P$.

On the other hand, (15) holds if the function $F(\mathbf{x})$ is locally convex and has continuous and bounded second derivatives (see [9], Proposition 3.4). Since the latter condition is clearly verified in the neighborhood of every local minimum, \exists a set of subsequences $\{k_j^{(i)}\}$ where the following inequalities are verified:

$$\frac{\|\mathbf{y}^{(k_j^{(i)})}\|^2}{(\mathbf{y}^{(k_j^{(i)})})^T \mathbf{s}^{(k_j^{(i)})}} \leq M_j^{(i)},$$

$$k_j^{(i)} > \hat{k}_j^{(i)}, \quad i = i_1, i_2, \dots, i_Q, \quad Q \leq P, \quad j = S_i \geq 1.$$

Moreover, for the finite set $\{k_j^{(i)} \leq \hat{k}_j^{(i)}, i = i_1, \dots, i_Q, Q \leq P, j = S_i \geq 1\}$, we have:

$$\frac{\|\mathbf{y}^{(k_j^{(i)})}\|^2}{(\mathbf{y}^{(k_j^{(i)})})^T \mathbf{s}^{(k_j^{(i)})}} \leq \hat{M}_j^{(i)}.$$

Therefore, (15) can be satisfied by setting:

$$M = \max\{\hat{M}_j^{(i)}, M_j^{(i)}, i = i_1, \dots, i_Q, Q \leq P, j = S_i \geq 1\}. \quad \square$$

Remark 4.1

- Every application of Sherman-Morrison-Woodbury formula in the tunneling phase has in our case a cost $O(n)$.
- The one-dimensional optimal search of μ_0 can be efficiently performed by applying Armijo-Goldstein method ([20, 23]).
- A satisfactory application of the algorithm depends on:

$$\left\{ \begin{array}{l} \text{the structure of eigenvalues of } (\mathcal{A}_{c(m)}^{\tilde{k}_m})^{-1}, \\ \text{the condition number of } \lambda_{c(m)}^{\tilde{k}_m} I + (I/\mu + R_{c(m)}^{\tilde{k}_m})^{-1}. \end{array} \right.$$

- The number of box-iterations and/or the operations performed in each iteration is in general considerably reduced with respect to the classical αBB procedure.

Remark 4.2

- The value α_m^* can be estimated for every box $c(m)$ by:

$$\max\left\{0, -\frac{1}{2} \min_i \lambda_i(\mathcal{B}_{c(m)}^{(k)})\right\}. \quad (24)$$

- The best algorithm must find an optimal compromise between the number of iterations \tilde{k}_m in each local minimization and the steps performed in the tunneling phase for every box $c(m)$.
- Alternative and more refined approximations of $\mathcal{A}_{c(m)}^{\tilde{k}_m}$ can be computed. For example, the application of a *black dot-type algorithm* [22, 24] might significantly increase the power of the tunneling phase.

Remark 4.3

- The effectiveness of algorithm $FB\alpha BB$ can be tested by several performances indexes on a suitable set of benchmark problems.
- In this paper numerical performances are evaluated in the simplest case $\text{rank}(R_{c(m)}) = 2$, i.e. when the repeller matrix $A_{c(m)}^{\tilde{k}_m}$ is the sum of a diagonal matrix and a rank 2 correction.
- We utilize in the local optimization phases a modified version of the BFGS-type algorithm $GB1$ described in [11] and based on diagonal approximations $\mathcal{B}_{c(m)}^{(k)}$ of $\nabla^2 F(\mathbf{x}_{c(m)})$ with a complexity of $\mathcal{O}(n)$ per step.

- The latter choice improves the efficiency of the algorithm particularly in the computation of (24).

5 Preliminary Computational Experience of Algorithm $FB\alpha BB$

The aim of this paragraph is to apply Algorithm $FB\alpha BB$ to some classical and well known optimization problems.

The functions investigated are the following ones:

$$\left\{ \begin{array}{l}
 \text{Camelback (CA):} \\
 (4 - 2.1x_1^2 + x_1^4/3)x_1^2 + x_1x_2 - 4(1 - x_2^2)x_2^2, \quad -5 \leq x_1, x_2 \leq 5. \\
 \text{Goldstein and Price (GP):} \\
 (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) \\
 (30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)), \\
 -2 \leq x_1, x_2 \leq 2. \\
 \text{Rastrigin (RA2, RA5, RA10, RA15, RA20):} \\
 10n + \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i)), \quad -5.12 \leq x_i \leq 5.12, n = 2, 5, 10. \\
 \text{Modified Himmelblau (MHB):} \\
 (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 + 0.1((x_1 - 3)^2 + (x_2 - 2)^2), \\
 -6 \leq x_1, x_2 \leq 6. \\
 \text{Rosenbrock (ROS10, ROS15, ROS20, ROS30):} \\
 \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2), \quad -5 \leq x_i \leq 10, n = 10, 15, 20. \\
 \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2), \quad -2 \leq x_i \leq 2, n = 30. \\
 \text{n-dimensional Test (n-dT):} \\
 \frac{1}{2} \sum_{i=1}^n (x_i^4 - 16x_i^2 + 5x_i) - \alpha \sum_{i=1}^n (x_i - 2.90353)^2, \\
 \alpha = 0, 0.3, -5 \leq x_i \leq 5. \\
 \text{Griewank (GW2, GW8):} \\
 \sum_{i=1}^n x_i^2/d - \prod_{i=1}^n \cos(x_i/\sqrt{i}) + 1, \\
 -600 \leq x_i \leq 600, n = 2, 8, d \text{ constant.} \\
 \text{Zakharov (ZA10, ZA15, ZA20):} \\
 \sum_{i=1}^n x_i^2 + \sum_{i=1}^n (0.5i x_i)^2 + \sum_{i=1}^n (0.5i x_i)^4, \\
 -5 \leq x_i \leq 10, n = 10, 15, 20. \\
 \text{Levy (L10, L50, L100):} \\
 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} ((y_i - 1)^2(1 + 10 \sin^2(\pi y_i + 1)) \\
 + (y_n - 1)^2(1 + 10 \sin^2(2\pi y_n))), \\
 y_i = 1 + (x_i - 1)/4.
 \end{array} \right.$$

Table 1 n -dimensional Test (n-dT), $n = 2$

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{(\bar{k}_m)})$	$\mathbf{x}_{c(m)}^{(\bar{k}_m)}$
0	(-10, -10)	(10, 10)	14.1367	(5.949, -6.0284)
1	(-10, -10)	(10, 0)	0	(2.7468, -2.9035)
2	(-10, -10)	(0, 0)	0	(2.7468, -2.9035)
3	(-10, -5)	(0, 0)	0	(-2.9035, -2.9035)

Let LM and GM indicate the local minima and the global minimum(a) of the corresponding functions, respectively. It is well known that:

- (CA) has 6LM and GM is located in (0.089482, -0.712656).
- (GP) has 4LM and GM is located in (0, -1).
- (RA-n) has more than 50LM and GM is assumed in (0, 0, ..., 0).
- (MHB) has 4LM and GM is located in (3, 2).
- (ROS-n) has several LM and GM is assumed in (1, 1, ..., 1).
- (n-dT) has 2^n LM and GM is located in (-2.9035, -2.9035, ..., -2.9035).
- (GW-n) has several hundreds of LM and GM is located in (0, 0, ..., 0).
- (ZA-n) has several local minima LM and GM is located in (0, 0, ..., 0).
- (L-n) has several local minima LM and GM is located in (1, 1, ..., 1).

We summarize here the main results of this preliminary computational experience, which was performed on a PC with a 2 GHz processor, 1 GB of RAM and a Matlab 2007 platform.

1. Benchmark problems are solved 100 times, by generating randomly the starting point in the feasible region.
2. Algorithm $FB\alpha BB$ is able to compute the global minimum for all the functions and the minimal accuracy is 10^{-4} .
3. Approximation of α_m^* in (16) is a critical problem.
4. Algorithm $FB\alpha BB$ approximates α_m^* by $\max\{0, -\frac{1}{2} \min_i \lambda_i (\mathcal{B}_{c(m)}^{(k)})\}$.
5. Algorithm $FB\alpha BB$ determines an optimal threshold $\delta(n)$ in order to avoid too many box iterations.
6. By 3. and 4. α_m^* can be evaluated only in a limited number of cases by $\max\{0, -\frac{1}{2} \min_i \lambda_i (\nabla^2 F(\mathbf{x}_{c(m)}^{(k)}))\}$.

The following tables show a sample of results for the indicated functions.

Remark 5.1

- Tables 1–10 point out that the number of iterations m required to compute the optimal solution is in general an acceptable value.
- One of the main advantages of Algorithm $FB\alpha BB$ is based on the synergy between the local minimization phases and the box iterations.
- Alternative and more refined approximations of $A_{c(m)}^{(\bar{k}_m)}$ might be utilized. For instance, if $\text{rank}(R_{c(m)}) = 3$ or 4 the power of the tunneling phases can be considerably increased.

Table 2 n -dimensional Test (n-dT), $n = 8$

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
1	$(-10, -10, \dots, -10)$	$(10, 10, \dots, 10, 0)$	56.5469	$(2.7468, -2.9035, 2.7468, 2.7468, 2.7468, -2.9035, -2.9035, -2.9035)$
.
4	$(-10, -10, \dots, -10)$	$(10, 0, \dots, 0, 0)$	0	$(-2.9035, \dots, -2.9035)$

Table 3 Modified Himmelblau (MHB)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
1	$(-6, 0)$	$(6, 6)$	3.4871	$(-2.7871, 3.1282)$
.	$(-6, 0)$	$(6, 6)$	3.4871	$(-2.7871, 3.1282)$
.
16	$(-4.9883, -4.5843)$	$(6, 6)$	1.5044	$(3.5815, -1.8208)$
17	$(0.50585, -4.5843)$	$(6, 6)$	1.5044	$(3.5815, -1.8208)$
18	$(0.50585, 0.79787)$	$(6, 6)$	2.8379e-11	$(3, 2)$

Table 4 Goldstein and Price (GP)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
1	$(-2, -2)$	$(2, 0)$	13777.4281	$(1.0417, -1.6848)$
.
3	$(0, -1)$	$(2, 0)$	90.0743	$(0.91592, 0.39076)$
4	$(0, -1)$	$(1, 0)$	3	$(8.4206e-09, -1)$
5	$(0, -1)$	$(1, -0.5)$	3	$(0, -1)$

Table 5 Rastrigin (RA-5)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
2	$(-5.12, \dots, 0)$	$(5.12, \dots)$	3.98	$(0, \dots, 1.9899)$
.
10	$(0, \dots, 0)$	$(5.12, 2.56, \dots)$	0.995	$(0, \dots, 0.99496)$
.
21	$(0, \dots, 0)$	$(0.64, \dots, 0.64)$	0	$(0, \dots, 0)$

Table 6 Rastrigin (RA-10)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\tilde{\mathbf{x}}_{c(m)}^{(k_m)})$	$\tilde{\mathbf{x}}_{c(m)}^{(k_m)}$
4	(-5.12, ..., 0, 0, 0)	(5.12, ...)	11.9395	(0, ..., 1.9899, ...)
.
.
.
22	(0, ..., 0)	(2.56, ..., 1.28)	9.9496	(0.99496, ...)
.
.
.
38	(0, ..., 0)	(1.28, .., 0.64, ...)	2.9849	(0.99496, .., 0, ...)
39	(0, ..., 0)	(1.28, .., 0.64, ...)	1.9899	(0.99496, 0.99496, 0, ...)
41	(0, ..., 0)	(0.64, ...)	0	(0, ..., 0)

Table 7 Rosenbrock (ROS-10)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\tilde{\mathbf{x}}_{c(m)}^{(k_m)})$	$\tilde{\mathbf{x}}_{c(m)}^{(k_m)}$
3	(-5, ..., 2.5, 2.5)	(10, ..., 10)	117555.33	(2.5, ..., 6.25, 6.25)
.
.
.
11	(-5, ..., 2.5, 2.5)	(2.5, ..., 10, 10)	115468.45	(-1.25, ..., 6.25, 6.25)
.
.
.
18	.	.	75.26	.
.
.
.
24	.	.	2.6743	.
.
.
.
76	.	.	0.	(1, ..., 1)

- Problems involving a number of variables $n \geq 10$ are solved efficiently by Algorithm $FB\alpha BB$. The latter fact is strictly dependent of the low complexity of the BFGS-type algorithm implemented in the local optimization phases ($\mathcal{O}(n)$ per iteration).
- In Table 11 the classical Number of Function Generations (NFG) index is shown for a set of experiments. RTA1 and RTA2 indicate the random tunneling algorithm

Table 8 Griewank (GW-2)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
1	(−600, −600)	(600, 600)	0	(0, 0)

Table 9 Griewank (GW-8)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
1	(−600, ...)	(600, ...)	1.6e−09	(2.e−05, 1.e−05, ..., 3.e−05)
2	(−600, ...)	(600, ...)	0	(0, ..., 0)

Table 10 Zakharov (ZA-10)

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\tilde{k}_m})$	$\mathbf{x}_{c(m)}^{\tilde{k}_m}$
1	(−5, ..., −5)	(10, ..., 10)	14511.91	.
2	(−5, ..., −5)	(10, ..., 10)	0	(0, ..., 0)

Table 11 NFG

Function	$F B \alpha B B$	RTA1	RTA2	FAEA
(CA)	131	76	135	303
(GP)	56	130	113	490
(RA-2)	320	656	383	544
(RA-5)	508	2092	687	2762
(RA-10)	766	9604	–	–
(RA-15)	885	23511	–	–
(RA-20)	946	41726	–	–
(MHB)	133	209	–	–
(ROS-10)	2171	399	–	–
(ROS-15)	2792	606	–	–
(4-dT) $\alpha = 0.3$	122	5634	–	–
(6-dT) $\alpha = 0.3$	188	9042	–	–
(8-dT) $\alpha = 0.3$	248	12922	–	–
(10-dT) $\alpha = 0.3$	304	17146	–	–
(GW-2)	43	1306	281	7804
(GW-8)	62	2381	465	–
(ZA-10)	48	238	–	–
(ZA-15)	63	437	–	–
(ZA-20)	84	673	–	–

described in [8] and [7] respectively. We underline that in Table 11 gradient calls are included in NFG.

- In the experiments the values shown are obtained by averaging the results associated to 100 different random starting points for all the algorithms.

Table 12 SR

Function	$FB\alpha BB$	RTA1
(2-dT)	100	100
(4-dT)	100	91
(6-dT)	100	46
(8-dT)	100	16
(10-dT)	100	5
(2-dT) $\alpha = 0.3$	100	100
(4-dT) $\alpha = 0.3$	100	81
(6-dT) $\alpha = 0.3$	100	33
(8-dT) $\alpha = 0.3$	100	10
(10-dT) $\alpha = 0.3$	100	4

Table 13 NLS/CPU

n	$FB\alpha BB$	HDM
10	1–2/2 s	2–3/2 s
50	1–3/5–30 s	–
100	1–5/10–80 s	4–6/183–739 s

- NFG associated to one of the best annealing evolutionary algorithm (FAEA) [25] are also shown. FAEA combines the features of a genetic approach with the properties of a fast simulated annealing algorithm.
- Poor performances of Algorithm $FB\alpha BB$ for (ROS- n) problems are due to propagation errors induced by diagonal approximations $B_{c(m)}^{(k)}$ of the Hessian matrix $\nabla^2 F(\mathbf{x}_{c(m)})$. This is not in general advisable if the function is based on powers of finite differences and *is not constrained in a small box* (see next Table 17). In the latter case, the utilization of the general BFGS-type method using dense and structured matrices and having a complexity of $\mathcal{O}(n \log n)$ per iteration (see [9, 10]) is recommended in the local optimization phases.
- In Table 12 are shown the Success Rates (SR) regarding both the classical (n-dT) problem and the modified (n-dT) problem (i.e. for $\alpha = 0.3$) (see [8] for a useful comparison). It is important to point out that the (n-dT) problem, $n = 2, 4, \dots, 10$, is solved by $FB\alpha BB$ with 100% success rate.
- In the case of (n-dT) and (RA- n) problems $FB\alpha BB$ outperforms RTA1 in terms of NFG. $FB\alpha BB$ is also more efficient than FAEA.

Let us consider further examples and comparisons.

In Table 13 are shown the results obtained by $FB\alpha BB$ when applied to the classical Levy function $L(y_1, y_2, \dots, y_n)$ for several values of n .

We have compared the performances of $FB\alpha BB$ with a recent and efficient Hybrid Descent Method (HDM) (see [26]) consisting of a combination of a simulated annealing algorithm and a gradient-based method.

The power of the repeller matrices in the tunneling phase is clearly pointed out in Table 13 in terms of number of local searches (NLS) and CPU.

Table 14 Levy $n = 100$

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\bar{k}_m})$	$\mathbf{x}_{c(m)}^{\bar{k}_m}$
1	$(-10, \dots, -10)$	$(10, \dots, 10)$	1078.79	$(-12.67, 4.58, 2.59, \dots, 7.23, 0.91)$
2	–	–	$6.27\text{e}-13$	$(1, 1, \dots, 1, 1)$

Table 15 Levy $n = 100$

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{\bar{k}_m})$	$\mathbf{x}_{c(m)}^{\bar{k}_m}$
1	$(-10, \dots, -10)$	$(10, \dots, 10)$	625.69	$(0.38, -4.36, -4.52, \dots, -0.3, 0.7)$
2	$(-10, \dots, -10, 0)$	$(10, \dots, 10)$	619.6	$(0.37, -4.39, -4.6, \dots, -0.3, 0.7)$
3	$(-10, \dots, 0, 0)$	$(10, \dots, 10)$	0.7	$(1.001, 1, \dots, 1, 3.65, 2.98)$
4	$(-10, \dots, 0, 0, 0)$	$(10, \dots, 10)$	0.45	$(1, 1, \dots, 1, 3.68, 0.99)$
·	·	·	·	·
7	–	–	$1.07\text{e}-13$	$(1, 1, \dots, 1, 1)$

The shown Tables 14 and 15, which refer to two different results related to the case $n = 100$, exhibit how the use of repeller matrices increases the efficiency and accuracy of $FB\alpha BB$.

We observe in particular that:

- In Table 14 a single repeller matrix is sufficient to evaluate the global minimum.
- In Table 15 two repeller matrices are applied (iterations 1 and 3).

Let us examine now the application of algorithm $FB\alpha BB$ to the Griewank function $GW(x_1, \dots, x_n)$:

$$GW = \sum_{i=1}^n x_i^2 / 2000 - \prod_{i=1}^n \cos(x_i / \sqrt{i}) + 1, \quad -600 \leq x_i \leq 600.$$

Table 16 illustrates the performances in terms of CPU time and NFG for a large number n of variables. Once again, we underline that all the values are obtained by averaging the results associated to 100 different starting points. Although it is well known that evaluating the global minimum of (GW) is harder for lower dimensions than higher ones [27, 28], Table 16 clearly shows that the utilization of a low-complexity BFGS-type algorithm guarantees satisfactory performances in all cases. (GW) has in fact a $\sum_{i=1}^n \cos(x_i / \sqrt{i})$ component causing linkages among variables, thereby making difficult to reach the global optimum for any algorithm.

Finally, it is interesting to compare our results with those ones recently published in [29]. The latter paper, which employs a Differential Evolution (DE) algorithm in conjunction with a non linear simplex method, named $NSDE$, is certainly more efficient both than traditional DE [30] and the Opposition based DE(ODE) [31].

Table 17 shows the performances obtained by applying $FB\alpha BB$ and $NSDE$ to the same problems investigated in the present work with the same number of variables (see [29] Table 2). In [29], however, Rosenbrock problem was defined in the set

Table 16 *GW-FB α BB*

n	CPU	NFG
30	0.5 s	18
50	1 s	12
100	2.5 s	13
200	6 s	15
500	15 s	17

Table 17 CPU-NFG

n	Function	<i>FBαBB</i> -CPU	<i>NSDE</i> -CPU	<i>FBαBB</i> -NFG	<i>NSDE</i> -NFG
30	GW	0.5 s	1.7 s	18	52240
30	L	1 s	1.11 s	115	35330
30	ZA	1.7 s	1.81 s	46	44960
10	RA	6 s	2.13 s	383	265420
30	ROS	3 s	10.4 s	441	232860

Table 18 Rosenbrock $n = 30$

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{(\bar{k}_m)})$	$\mathbf{x}_{c(m)}^{(\bar{k}_m)}$
1	$(-2, \dots, -2)$	$(2, \dots, 2)$	149.22	$(0.01, -0.18, 0.78, \dots, 0.03, 0.08)$
2	$(-2, \dots, 0)$	$(2, \dots, 2)$	126.28	$(0.01, -0.05, 0.75, \dots, 0.01, 0.02)$
3	–	–	124.77	$(0.01, -0.04, 0.78, \dots, 0, 0)$
4	–	–	1.66e–13	$(1, 1, \dots, 1, 1)$

$-2 \leq x_i \leq 2$. In the latter box *FB α BB* is able to determine repeller matrices having a remarkable effect in terms of CPU-time, NFG and accuracy.

In all the tables NFG do not include gradient calls. Note that, although the CPU-time per iteration (box $c(m)$) is in general a critical parameter of *FB α BB* with respect to *NSDE*, the total CPU-time of the former algorithm is in the majority of cases quite satisfactory. The latter fact is obviously due to the low NFG and/or NLS requested by *FB α BB*. Last but not least, the linear complexity per step of the BFGS-type algorithm plays an important role.

Tables 18–19 show in details the importance of repeller matrices in the case of Rosenbrock problem, previously pointed out for Levy function.

We underline that:

- In Table 18 two repeller matrices at iterations 1 and 2 respectively allow to evaluate efficiently the global minimum.
- In Table 19 four repeller matrices are applied (iterations 1, 3, 4 and 5) and the algorithm requires NFG = 241 only.

It is worth while summarizing some final considerations.

Table 19 Rosenbrock $n = 30$

m	$\mathbf{x}_{c(m)}^L$	$\mathbf{x}_{c(m)}^U$	$F(\mathbf{x}_{c(m)}^{(\tilde{k}_m)})$	$\mathbf{x}_{c(m)}^{(\tilde{k}_m)}$
1	$(-2, \dots, -2)$	$(2, \dots, 2)$	127.84	$(0.06, -0.18, 0.32, \dots, -0.01, 0.02)$
2	$(-2, \dots, -2, 0)$	$(2, \dots, 2)$	113.62	$(0.04, -0.1, 0.14, \dots, 0.02, 0.005)$
3	–	–	103.66	$(0.04, -0.07, 0.02, \dots, 0, 0.005)$
4	–	–	62.77	$(0.04, 0.008, 0.02, \dots, 0.01, 0.02)$
5	–	–	55.35	$(0.03, 0.006, 0.02, \dots, 0.008, 0)$
6	–	–	51.87	$(0.02, -0.002, 0.01, \dots, 0.0001, 0)$
7	–	–	1.46e–13	$(1, 1, \dots, 1, 1)$

Remark 5.2

1. Since the CPU-time per box-iteration (outer cycle $c(m)$ of $FB\alpha BB$) is certainly an important constraint, an extensive utilization of repeller matrices is recommended in order to minimize the iterations m .
2. In the majority of cases repeller matrices induce a satisfactory value of the performance index NLS.
3. On the other hand, when $FB\alpha BB$ computes a low NLS, the linear complexity per step of the BFGS-type algorithm (inner cycle k_m of $FB\alpha BB$) guarantees excellent values of the performance index NFG.
4. When NLS and NFG assume low values, the propagation error of the whole algorithm is minimized, thereby implying a maximal accuracy in the computation of the optimal solution.

6 Conclusions and Future Research

One of the advantages of the present method depends upon the fact that the local optimization phases are performed by means of a BFGS-type algorithm with $\mathcal{O}(n)$ complexity per iteration. The latter property has in general a remarkable effect on functions of operational interest [10, 11] and is clearly illustrated in the benchmark problems presented in this paper.

It is important to stress that the classical BFGS method has $\mathcal{O}(n^2)$ complexity per iteration and therefore cannot be implemented efficiently to minimize functions with a large number of variables (see [6, 12]).

Although all the experiments were performed by utilizing a non-professional software in a Matlab platform, the results clearly indicate that the approach described in this paper is extremely promising. Of course, in order to deal with problems involving in general several hundreds of variables, many improvements must be added in the practical implementation of $FB\alpha BB$.

More precisely, suitable optimizations of the latter algorithm (i.e., restarting rules in local minima evaluation, skilled criteria in the operational matching with the αBB computational scheme, stronger accuracy etc.) are recommended and must be inserted in the software.

Moreover, the new class of general purpose convex underestimators introduced in [32, 33] and implemented for box-constrained problems in [34] (see Algorithm $G\alpha BB$) might further improve the effectiveness of $FB\alpha BB$. The present algorithm, in fact, can be also applied in the frame of $G\alpha BB$ computational approach. Furthermore, as pointed out in Remark 5.1, the power of the repeller matrices in the tunneling phases can be increased by utilizing matrices $A_{c(m)}^{(k_m)}$ containing corrections $R_{c(m)}$ of rank 3 or 4. Obviously, one of the aims of the future research is to find an optimal compromise between the performance indexes NLS-NFG and the computational effort requested to evaluate $R_{c(m)}$.

The author underlines that the idea of combining low-complexity BFGS-type algorithms in the local optimizations and suitable repeller matrices in the tunneling phases is quite new.

Work is in progress to investigate the best way to implement all the above cited optimizations by utilizing the approach described in this paper.

References

1. Levy, A.V., Montalvo, A.: The tunneling algorithm for the global minimization of functions. *SIAM J. Sci. Stat. Comput.* **6**, 15–29 (1985)
2. Yao, Y.: Dynamical tunneling algorithm for global optimization. *IEEE Trans. Syst. Man Cybern.* **19**, 1222–1230 (1989)
3. Cetin, B.C., Barhen, J., Burdick, J.W.: Terminal repeller unconstrained subenergy tunneling for fast global optimization. *J. Optim. Theory Appl.* **77**, 97–126 (1993)
4. Barhen, J., Protopopescu, V., Reister, D.: TRUST: A deterministic algorithm for global optimization. *Science* **276**, 1094–1097 (1997)
5. Barhen, J., Burdick, J.W., Cetin, B.C.: Global descent replaces gradient descent to avoid local minima problem in learning with artificial neural networks. In: *ICNN93*, vol. 2, pp. 836–842 (1993)
6. Di Fiore, C., Fanelli, S., Zellini, P.: Computational experiences of a novel global algorithm for optimal learning in MLP-networks. In: *ICONIP'02*, vol. 1, pp. 317–321 (2002)
7. Jiang, H., Cai, W., Shao, X.: A random tunneling algorithm for the structural optimization problem. *Phys. Chem. Chem. Phys.* **4**, 4782–4788 (2002)
8. Srinivasa, M., Rangaiah, G.P.: Implementation and evaluation of random tunneling algorithm for chemical engineering applications. *Comput. Chem. Eng.* **30**(9), 1400–1415 (2006)
9. Di Fiore, C., Fanelli, S., Lepore, F., Zellini, P.: Matrix algebras in quasi-Newton methods for unconstrained optimization. *Numer. Math.* **94**, 479–500 (2003)
10. Bortoletti, A., Di Fiore, C., Fanelli, S., Zellini, P.: A new class of quasi-Newtonian methods for optimal learning in MLP-networks. *IEEE Trans. Neural Netw.* **14**, 263–273 (2003)
11. Di Fiore, C., Fanelli, S., Zellini, P.: An efficient generalization of Battiti-Shanno's quasi-Newton algorithm for optimal learning in MLP-networks. In: *Neural Information Processing*, vol. 1, pp. 483–488. Springer, Calcuta (2004)
12. Di Fiore, C., Fanelli, S., Zellini, P.: Low complexity minimization algorithms. *Numer. Linear Algebra Appl.* **12**, 755–768 (2005)
13. Di Fiore, C., Fanelli, S., Zellini, P.: Low complexity secant quasi-Newton minimization algorithms for nonconvex functions. *J. Comput. Appl. Math.*, **210**, 167–174 (2007)
14. Floudas, C.A., Visweswaran, V.: A primal relaxed dual global optimization approach. *J. Optim. Theory Appl.* **78**(2), 187–225 (1993)
15. Floudas, C.A.: *Deterministic Global Optimization*. Dordrecht, Kluwer (2000)
16. Meyer, C.A., Floudas, C.A.: Convex underestimation of twice continuously differentiable function by piecewise quadratic perturbation: spline αBB underestimators. *J. Glob. Optim.* **232**(2), 221–258 (2005)
17. Gounaris, C.E., Floudas, C.A.: Tight convex underestimators for C^2 -continuous problems: I univariate functions. *J. Glob. Optim.* **42**(1), 51–67 (2008)

18. Gounaris, C.E., Floudas, C.A.: Tight convex underestimators for C^2 -continuous problems: II multivariate functions. *J. Glob. Optim.* **42**(1), 69–89 (2008)
19. Gounaris, C.E., Floudas, C.A.: Convexity of products of univariate functions and convexification transformations for geometric programming. *J. Optim. Theory Appl.* **138**(3), 407–427 (2008)
20. Nocedal, J., Wright, S.J.: *Numerical Optimization*. Springer, Berlin (1999)
21. Powell, M.J.D.: Some global convergence properties of a variable metric algorithm for minimization without exact line search. In: *Nonlinear Programming*. SIAM-AMS Proc., vol. 9, pp. 53–72 (1976)
22. Oseledets, I., Tyrtshnikov, E.: A unifying approach to the construction of circulant preconditioners. *Linear Algebra Appl.*, **418**, 435–449 (2006)
23. Wolfe, P.: Convergence conditions for descent methods. *SIAM Rev.* **11**, 226–235 (1969)
24. Tyrtshnikov, E.: Private communication
25. Cai, W., Shao, X.: A Fast annealing evolutionary algorithm for global optimization. *J. Comput. Chem.* **23**, 427–435 (2002)
26. Yiu, K.F.C., Liu, Y., Teo, K.L.: A hybrid descent method for global optimization. *J. Glob. Optim.* **28**, 229–238 (2004)
27. Liang, J.J., Qin, A.K.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans. Evol. Comput.*, **10**(3), 281–295 (2006)
28. Whitley, D., Mathias, K., Rana, S., Dzubera, J.: Evaluating evolutionary algorithms. *Artif. Intell.* **85**, 245–276 (1996)
29. Musrrat, A., Millie, P., Ajith, A.: Simplex differential evolution. *Acta Polytech. Hung.*, **6**, 95–115 (2009)
30. Storn, R., Price, K.: DE—a simple and efficient heuristics for global optimization over continuous space. *J. Glob. Optim.* **11**(4), 41–359 (1997)
31. Rahnamayan, S., Tizhoosh, H.R., Salman, M.A.: Opposition-based differential evolution. *IEEE Trans. Evol. Comput.* **12**(1), 64–79 (2008)
32. Akrotirianakis, I.G., Floudas, C.A.: A new class of improved convex underestimators for twice continuously differentiable constrained NLP's. *J. Glob. Optim.*, **30**(4), 367–390 (2004)
33. Floudas, C.A., Gounaris, C.E.: A review of recent advances in global optimization. *J. Glob. Optim.* **45**, 3–38 (2009)
34. Akrotirianakis, I.G., Floudas, C.A.: Computational experience with a new class of convex underestimators: box-constrained NLP problems. *J. Glob. Optim.* **29**(3), 249–264 (2004)