J Glob Optim (2009) 44:193–212
DOI 10.1007/s10898-008-9318-6

Hybrid spectral gradient method for the unconstrained minimization problem

William La Cruz · Gilberto Noguera

Received: 12 June 2007 / Accepted: 4 June 2008 / Published online: 8 July 2008

Abstract We present a hybrid algorithm that combines a genetic algorithm with the Barzilai–Borwein gradient method. Under specific assumptions the new method guarantees the convergence to a stationary point of a continuously differentiable function, from any arbitrary initial point. Our preliminary numerical results indicate that the new methodology finds efficiently and frequently the global minimum, in comparison with the globalized Barzilai–Borwein method and the genetic algorithm of the Toolbox of Genetic Algorithms of MatLab.

Keywords Unconstrained minimization problem · Genetic algorithms · Gradient methods · Barzilai–Borwein method

Mathematics Subject Classifications (2000) 65K10 · 49M37 · 90C59

1 Introduction

We consider the unconstrained minimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function defined on $\mathbb{R}^n$.

Problem (1) is frequently solved using iterative methods (e.g., Newton's method, Quasi-Newton methods, gradient methods) that generate an approximate solution at each iteration.

W. La Cruz (✉)
Departamento de Electrónica, Computación y Control, Facultad de Ingeniería, Universidad Central de Venezuela, Caracas 1051-DF, Venezuela
e-mail: william.lacruz@ucv.ve

G. Noguera
Área de Matemática, Universidad Nacional Abierta, Caracas, Venezuela
e-mail: noguerag@ucv.ve

Another approach that can be used is the family of direct search methods (e.g., Nelder–Mead method [51]).

The Barzilai–Borwein (BB) method [2] is a gradient method that uses a very special step length. Concretely, the BB method for problem (1) is defined by

$$x_{k+1} = x_k - \frac{1}{\alpha_k}\nabla f(x_k), \tag{2}$$

where $\nabla f(x_k)$ is the gradient vector of f at x_k and the scalar α_k, called the spectral step length, is given by

$$\alpha_k = \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}}, \tag{3}$$

where $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = \nabla f(x_k) - \nabla f(x_{k-1})$.

When f is a strictly convex quadratic, Raydan [53] demonstrates that BB converges to the solution of (1), and when f is an arbitrary function, Raydan [54] incorporates a globalization strategy to BB that allows the convergence to a local minimum of f, starting from any initial point x_0. This way, Raydan [54] builds the Global Barzilai–Borwein (GBB) method (see Raydan [54, p. 28]).

The structure of iteration (2) is very attractive, especially when one deals with large-scale problems. Each iteration only needs the evaluation of $\nabla f(x_k)$ and some additional floating point operations which are linear in terms of n. By its simplicity and easy implementation, BB is an efficient method for large-scale minimization that has been satisfactorily used in many applications [5,6,8,18,21,38–40,50,58]. Additional theoretical properties have been recently studied [24–31,35,36].

There are variations of BB method for solving systems of nonlinear equations [45–47], that have competed favorably with the Newton-Krylov schemes [3,16,17]. Also the projected version of BB (Spectral Projected Gradient method [11–13]) has been intensively used in applications of great interest [1,4,7,9,10,14,15,22,23,32,37,61,52,55–57,60].

On the other hand, *Genetic Algorithms* (GA) have been used for solving problem (1) considering an objective function that does not possess good properties as continuity, differentiability, or the Lipschitz condition, among others [34,41,43,44,49]. According to Golberg [41] the GAs are search algorithms based on the natural selection and the natural genetics. These algorithms maintain and handle a population of solutions and carry out a strategy of survival of the "best individual" in their search to obtain better solutions.

Therefore, problem (1) has been solved using iterative deterministic methods and also heuristic methods. The iterative methods generally are used under certain differentiability conditions and only guarantee the convergence to local minima. Otherwise, the heuristic methods do not need differentiability and guarantee convergence to global minima, but require a large number of function evaluations and a large CPU time.

The main objective of this work is to solve problem (1) combining the GAs with the BB method. Under specific assumptions the new method guarantees the convergence to a stationary point of f from an initial point $x_0 \in \mathbb{R}^n$ chosen arbitrarily. The new methodology can be considered as a hybrid GA for optimization, but not in the traditional form. The difference with our approach will be described below.

Traditionally a hybrid GA carries out first a certain number of generations and then an iterative method (Newton's method or Quasi-Newton, etc.), or another heuristic (simulated annealing, etc.), is applied to refine the approximations. On the contrary in the proposed method, the iterative algorithm is applied to guide the search of the GA, that is to say, every time that BB generates an iterate $x_k \in \mathbb{R}^n$, the GA looks inside a certain region that contains x_k for some other point that improves the value of $f(x_k)$.

Our preliminary numerical results indicate that the new methodology finds efficiently and quite frequently the global minimum, in comparison with the method of Global Barzilai–Borwein and the GA of the Toolbox of Genetic Algorithms of MatLab.

The paper is structured in the following way. In Sect. 2 the new method is described and some convergence results are presented. In Sect. 3 numerical experiences are presented in the solution of some test problems and also in the solution of molecular conformation problems. Finally, in Sect. 4 some final comments are included.

2 Hybrid spectral gradient method

This section describes the new method for the unconstrained minimization problem, denoted by *Hybrid Spectral Gradient* (HSG) method. Section 2.1 shows the implementation of the GAs in the resolution of the unconstrained minimization problem. In Sect. 2.2 the algorithm of HSG method is presented. Finally, in Sect. 2.3 we prove some convergence results.

2.1 Genetic algorithm for unconstrained minimization

Initially the GA chooses randomly an *initial population* and then carries out a local search to generate a new population through a process that emulates the natural selection, and the operators such as crossover and mutation. A *generation* in a GA is the process of obtaining a new population starting from a current population.

Each individual of the population represents a solution of the optimization problem and it should be coded with symbols to be used in the GA. The individuals are coded through a binary representation. Other representations are also used, such as the real representation introduced by Wright [59]. To the effects of applying the GAs to the problem (1), a population P is a finite subset of $\mathbb{R}^n$ given by

$$P = \{y_1, y_2, \dots, y_N\}, \tag{4}$$

where N is the population size and an individual $y_i = (y_i^1, y_i^2, \dots, y_i^n)^T$ is a point of $\mathbb{R}^n$, where the coordinates y_i^j are their genes for $i = 1, \dots, n$.

To select an individual of a population the GA needs to know their adaptation. According to Davis [34] the *adaptation function* is the union between the GA and the considered problem. In the case of problem (1) the adaptation function is the same merit function. In other words, for an individual y_i of the population P given in (4) the value $f(y_i)$ is its adaptation.

The reproduction of new individuals considers the genetic operators that constitute, inside a GA, the processes that simulate the genetic movements of the cellular division. The genetic operators that are commonly used in the GAs are: crossover and mutation. For the problem (1) we use the *simple crossover*. With the *selection process* (roulette wheel) two individuals denoted parents are chosen. Then the crossing point is chosen, which is the coordinate where the exchange of the segments of the parents will be carried out, and it is obtained generating randomly an integer j in the set $\{1, 2, \dots, n-1\}$. Then the segments are exchanged (before and after the crossing point) of the parents. For example, if

$$x = (x^1, x^2, \dots, x^n)^T \qquad \text{and} \qquad y = (y^1, y^2, \dots, y^n)^T$$

are the parents and j is the crossing point, then the crossing of x with y generates two children h_1 and h_2 given by

$$h_1 = (x^1, \ldots, x^j, y^{j+1}, \ldots, y^n)^T \quad \text{and} \quad h_2 = (y^1, \ldots, y^j, x^{j+1}, \ldots, x^n)^T.$$

We use as mutation operator to the *Gaussian mutation* (see James [44]). A children $h = (h_1, \ldots, h_n)^T$ is randomly selected. Then, a vector $v = (v_1, \ldots, v_n)^T$ is randomly generated such that v_i follows a normal distribution with mean 0 and variance 1. Lastly, the Gaussian mutation of h, denoted by h_m, is defined as $h_m = h + \xi v$, where $\xi > 0$. In our numerical experiences we use $\xi = 0.5$.

Commonly, at each generation of a GA the current population is replaced completely. This leads to the frequent loss of the best individual of the population at each generation, that can affect the efficiency of the GA. To overcome such difficulty, we use the *elitism* technique. The elitism conserves the best individual of the current population at each generation, and so, if the population size is N, then the new population is obtained creating $N - 1$ new individuals, and then adding the individual that possesses the best adaptation in the current population.

Algorithm 1 describes the evaluation and reproduction of a population of a GA.

Algorithm 1 (Genetic Algorithm)

```
Start: Choose randomly an initial population $P_0$, an integer $G > 0$ and $k = 0$.
1: Evaluate each individual of $P_k$;
2: repeat
3:    reproduce $P_{k+1}$ starting from $P_k$;
4:    evaluate each individual of $P_{k+1}$;
5:    $k = k + 1$;
6: until $k = G$.
```

Remark 1

(i) The stopping criteria of Algorithm 1 is the maximum number of generations given by the integer $G > 0$.
(ii) The evaluation of the population P_k consists of computing the adaptation of each individual of P_k.
(iii) The reproduction process consists of applying the genetics operators such as selection, crossover and mutation, to the population P_k, for obtaining new individuals that conform the population P_{k+1}. As described before, the population P_{k+1} conserves the best individual of P_k.

2.2 The HSG algorithm

The HSG method combines the GAs with a global version of BB that is, in turn, obtained as a combination of the BB method with a globalization strategy developed by La Cruz et al. [46] and La Cruz [45].

Given $x_0 \in \mathbb{R}^n$, HSG first generates a sequence

$$z_k = x_k - \lambda_k \nabla f(x_k), \qquad \text{for } k = 0, 1, \ldots \tag{5}$$

that satisfies the condition

$$f(z_k) \leq f(x_k) + \eta_k - \gamma \lambda_k^2 \|d_k\|^2, \qquad \text{for } k = 0, 1, \ldots \tag{6}$$

where $\|\cdot\|$ denotes the Euclidean norm, $d_k = -\nabla f(x_k)$, $\lambda_k > 0$, $\gamma \in (0, 1)$ and $\{\eta_k\}$ is a fixed sequence of positive numbers such that

$$\sum_{k=0}^{\infty} \eta_k < \eta < \infty. \tag{7}$$

Then, HSG randomly chooses a point $w_k \in \mathbb{R}^n$ from a box in $\mathbb{R}^n$ with center in z_k. For it, HSG generates an initial population P_0 of size N contained in a box in $\mathbb{R}^n$ with center in z_k; then the GA given in Algorithm 1 obtains G generations. Subsequently the point $w_k \in \mathbb{R}^n$ is chosen as the best individual of the population P_G. The box trust region, $z = (z^1, z^2, \ldots, z^n)^T \in P_0$, is given by

$$|z^i - z_k^i| \le r, \qquad i = 1, 2, \ldots, n, \tag{8}$$

where $r > 0$.

If $f(w_k) \le f(z_k)$, then HSG defines $x_{k+1} = w_k$ as an approximate solution of (1). On the contrary, if $f(w_k) > f(z_k)$ then HSG defines x_{k+1} as follows

$$x_{k+1} = \begin{cases} x_k - (\lambda_k/2)\nabla f(x_k), & \text{if } f(x_k) < f(z_k); \\ z_k, & \text{if } f(x_k) \ge f(z_k). \end{cases}$$

The whole previous process is repeated until $\|\nabla f(x_k)\| = 0$ is satisfied for some $k \ge 1$, which guarantees that the iterate x_k is a stationary point of $f(x)$. Algorithm 2 below is a formal description of the HSG method.

Algorithm 2 (**Hybrid Spectral Gradient Method**)

```
Start: Choose $x_0 \in \mathbb{R}^n$, $\alpha_0 \in \mathbb{R}$, a positive sequence $\{\eta_k\}$ that satisfies (7), $\gamma \in (0,1)$,
    $0 < \sigma_1 < \sigma_2 < 1$, $0 < \epsilon < 1$, $\delta > 0$, integers $G > 0$ and $N > 0$, $r > 0$ and $k = 0$.
1:  repeat
2:    if $\alpha_k \le \epsilon$ or $\alpha_k \ge 1/\epsilon$, let $\alpha_k = \delta$;
3:    let $\lambda = 1/\alpha_k$;
4:    while $f(x_k - \lambda\nabla f(x_k)) > f(x_k) + \eta_k - \gamma\lambda^2 \|\nabla f(x_k)\|^2$ do
5:      choose $\sigma \in [\sigma_1, \sigma_2]$;
6:      let $\lambda = \sigma\lambda$;
7:    end while
8:    let $\lambda_k = \lambda$;
9:    let $z_k = x_k - \lambda_k \nabla f(x_k)$;
10:   obtain $G$ generations of Algorithm 1 with initial population $P_0$ of size $N$ that satisfies
      (8);
11:   let $w_k = \min_{w \in P_G}[f(w)]$;
12:   if $f(w_k) < f(z_k)$ then
13:     $x_{k+1} = w_k$;
14:   else
15:     if $f(x_k) < f(z_k)$ then
16:       let $\lambda_k = \lambda_k/2$;
17:       let $x_{k+1} = x_k - \lambda_k \nabla f(x_k)$;
18:     else
19:       let $x_{k+1} = z_k$;
20:     end if
21:   end if
22:   let $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$;
```

```
23:   let $\alpha_{k+1} = -\left(\nabla f(x_k)^T y_k\right) / \left(\lambda_k \nabla f(x_k)^T \nabla f(x_k)\right)$;
24:   $k = k + 1$;
25: until $\|\nabla f(x_k)\| = 0$.
```

Remark 2

(i) Algorithm 2 is well defined. Indeed, by continuity of f and $\eta_k > 0$, the condition (6) is satisfied after a finite number of reductions of λ.

(ii) The sequence $\{x_k\}$ generated by Algorithm 2 satisfies:

$$f(x_{k+1}) \leq f(x_k) + \eta_k - \gamma \lambda_k^2 \|\nabla f(x_k)\|^2, \quad \text{for all } k \geq 0. \tag{9}$$

2.3 Convergence analysis

Next we present two technical results that describe some characteristics of the HSG method. These results will allow us to demonstrate some convergence results.

The following proposition shows that the sequence $\{x_k\}$ generated by Algorithm 2 is contained in a certain set.

Proposition 2.1 *The sequence $\{x_k\}$ generated by Algorithm 2 is contained in the set*

$$\Omega_0 = \left\{x \in \mathbb{R}^n : f(x) \leq f(x_0) + \eta\right\}.$$

Proof By (9) we can write for $j \geq 0$:

$$\begin{aligned} f(x_{j+1}) &\leq f(x_j) + \eta_j \\ f(x_{j+2}) &\leq f(x_{j+1}) + \eta_{j+1} \leq f(x_j) + \eta_j + \eta_{j+1} \\ f(x_{j+3}) &\leq f(x_{j+2}) + \eta_{j+2} \leq f(x_j) + \eta_j + \eta_{j+1} + \eta_{j+2} \\ &\vdots \end{aligned}$$

Following with this inductive process we obtain:

$$f(x_{j+k}) \leq f(x_j) + \sum_{i=j}^{j+k-1} \eta_i, \text{ for } j \geq 0, \text{ and } k \geq 1. \tag{10}$$

By (7) and (10), $f(x_k) \leq f(x_0) + \eta$, for $k \geq 1$. In other words, the sequence $\{x_k\}$ is contained in the set Ω_0. □

Proposition 2.2 *Assume that the set Ω_0 is compact and let $\{x_k\}$ be the sequence generated by Algorithm 2. Then*

$$\lim_{k \to \infty} \lambda_k \|\nabla f(x_k)\| = 0. \tag{11}$$

Proof By (9) we have

$$\lambda_k^2 \|\nabla f(x_k)\|^2 \leq \frac{\eta_k}{\gamma} + \frac{f(x_k) - f(x_{k+1})}{\gamma}, \text{ for all } k \geq 0. \tag{12}$$

Since η_k satisfies (7), adding all terms both sides of (12) it follows that

$$\sum_{k=0}^{\infty} \lambda_k^2 \|\nabla f(x_k)\|^2 \leq \frac{1}{\gamma}\left(\sum_{k=0}^{\infty} \eta_k + \sum_{k=0}^{\infty}(f(x_k) - f(x_{k+1}))\right)$$
$$\leq \frac{1}{\gamma}\left(\left|\sum_{k=0}^{\infty} \eta_k\right| + \left|\sum_{k=0}^{\infty}(f(x_k) - f(x_{k+1}))\right|\right)$$
$$\leq \frac{\eta + |f(x_0)|}{\gamma} < \infty. \tag{13}$$

Since $\lambda_k \|\nabla f(x_k)\| \geq 0$, then by (13) equation (11) holds. □

In Theorem 2.1 we prove that all limit points of the sequence $\{x_k\}$ generated by Algorithm 2 are stationary points of f, in other words, the sequence $\{\nabla f(x_k)\}$ converges to 0. In this theorem it is assumed that Ω_0 is a compact set. Therefore, under this specific assumptions Theorem 2.1 says that the HSG method always finds a stationary point of f.

Theorem 2.1 *Assume that the set Ω_0 is compact and let $\{x_k\}$ be the sequence generated by Algorithm 2. Then*

$$\lim_{k\to\infty} \nabla f(x_k) = 0. \tag{14}$$

Proof Let x_* be a limit point of $\{x_k\}$. Without loss of generality we can assume that the sequence $\{x_k\}$ converges to x_*. By Proposition 2.2 we have

$$\lim_{k\to\infty} \lambda_k \|\nabla f(x_k)\| = 0.$$

This is true if

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$$

or if

$$\liminf_{k\to\infty} \lambda_k = 0. \tag{15}$$

If $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$, (14) holds. If (15) holds and $\nabla f(x_*) \neq 0$, there exists an infinite set of indices L such that

$$\lim_{k\to\infty, k\in L} \lambda_k = 0.$$

By the way λ_k was chosen in Algorithm 2, there exists an index $\bar{k}$ sufficiently large such that for all $k \geq \bar{k}$, $k \in L$, there exists σ_k ($\sigma_1 \leq \sigma_k \leq \sigma_2$) for which $\lambda = \lambda_k/\sigma_k$ fails to satisfy condition (6), i.e.,

$$f(x_k - (\lambda_k/\sigma_k)\nabla f(x_k)) > f(x_k) + \eta_k - \gamma(\lambda_k^2/\sigma_k^2)\|\nabla f(x_k)\|^2$$
$$> f(x_k) - \gamma(\lambda_k^2/\sigma_k^2)\|\nabla f(x_k)\|^2.$$

Therefore,

$$\frac{f(x_k - (\lambda_k/\sigma_k)\nabla f(x_k)) - f(x_k)}{(\lambda_k/\sigma_k)} > -\gamma(\lambda_k/\sigma_k)\|\nabla f(x_k)\|^2$$
$$> -\gamma(\lambda_k/\sigma_1)\|\nabla f(x_k)\|^2. \tag{16}$$

Since $d_k = -\nabla f(x_k)$, by the Mean Value Theorem there is $t_k \in [0, \lambda_k/\sigma_k]$ that tends to zero when $k \to \infty$ such that

$$\frac{f(x_k + (\lambda_k/\sigma_k)d_k) - f(x_k)}{(\lambda_k/\sigma_k)} = \nabla f(x_k + t_k d_k)^T d_k,$$

in other words,

$$\frac{f(x_k - (\lambda_k/\sigma_k)\nabla f(x_k)) - f(x_k)}{(\lambda_k/\sigma_k)} = -\nabla f(x_k - t_k \nabla f(x_k))^T \nabla f(x_k). \tag{17}$$

By (16) and (17) we obtain, for all $k \geq \overline{k}$, $k \in L$,

$$-\nabla f(x_k - t_k \nabla f(x_k))^T \nabla f(x_k) > -\gamma(\lambda_k/\sigma_1)\|\nabla f(x_k)\|^2. \tag{18}$$

Since $(x_k - t_k\nabla f(x_k)) \to x_*$ as $k \to \infty$ and $k \in L$, then taking limits in (18) as $k \to \infty$, $k \in L$, we deduce that

$$-\nabla f(x_*)^T \nabla f(x_*) \geq 0.$$

This is true if and only if $\nabla f(x_*) = 0$. This completes the proof. □

3 Numerical results

We compare the numerical behavior of the implementations in MatLab of the methods GBB and HSG, and the function `ga` of the Toolbox of Genetic Algorithms of MatLab, for a set of test functions (Appendix A) and the molecular conformation problem. All the runs were carried out on a Pentium IV computer at 3.0 GHz with machine epsilon equal 2×10^{-6}.

3.1 Implementation

For the function `ga` of the Toolbox of Genetic Algorithms of MatLab we use the option:

```
opts = optimset('TolFun',etol,'GradObj','on','LargeScale','on'),

  options = gaoptimset('PopulationSize',50,'Generations',500,...
                 'PopInitRange',[ℓi;ℓs],'HybridFcn',{@fminunc, opts}),
```

where $\ell_i \leq x^i \leq \ell_s$, for all $x = (x^1, x^2, \ldots, x^n) \in \mathbb{R}^n$, and choose $etol = 10^{-3}$ for the test problems and $etol = 10^{-6}$ for the molecular conformation problem.

The function `ga`, with the selected options, is a hybrid GA that uses the function `fminunc` of the Toolbox of optimization of MatLab. The option selected with `optimset`, allows to use a large-scale algorithm and the analytic expression of the gradient in the function `fminunc`. In this case, `fminunc` is a subspace trust region method which is based on the interior-reflective Newton's method described in [20,19]. Each iteration involves the approximate solution of a large linear system using the preconditioned conjugate gradient method.

We implement the GBB method in MatLab with the parameters described in Raydan [54, p. 30].

We implement the Algorithm 2 in MatLab with the following parameters:

- $\alpha_0 = 1$; $\epsilon = 10^{-10}$; $\gamma = 10^{-4}$; $\sigma_1 = 0.1$; $\sigma_2 = 0.5$;

- $\delta > 0$ define by

$$\delta = \begin{cases} 1, & \text{if } \|\nabla f(x_k)\| > 1; \\ \|\nabla f(x_k)\|, & \text{if } 10^{-5} \leq \|\nabla f(x_k)\| \leq 1; \\ 10^{-5}, & \text{if } \|\nabla f(x_k)\| < 10^{-5}; \end{cases}$$

- $\eta_k = \theta\rho_k$, where $\theta > 0$ and $\{\rho_k\}$ is a sequence of positive numbers such that $\sum_{k=0}^{\infty} \rho_k < \infty$. The parameters θ and ρ_k depend on the treated problem. For the test functions we set $\rho_k = (0.99)^k$ and

$$\theta = \begin{cases} |f(x_0)|, & \text{if } |f(x_0)| \leq 1; \\ 5, & \text{if } 1 < |f(x_0)| \leq 10; \\ 10, & \text{if } 10 < |f(x_0)| \leq 50; \\ 50, & \text{if } 50 < |f(x_0)| \leq 100; \\ 100, & \text{if } 100 < |f(x_0)| \leq 1000; \\ 10^4, & \text{if } |f(x_0)| > 1000. \end{cases}$$

For the molecular conformation problem we set $\rho_k = (0.9)^k$ and $\theta = |f(x_0)|$.

- For the GA given in Algorithm 1, we define the probabilities of crossing (p_c) and mutation (p_m) as:

$$p_c = \begin{cases} 0.8, & \text{if } G = 1; \\ \left(\frac{0.8}{G-1}\right) j + 0.1 \text{ for } j = 1, 2, \ldots, G-1, & \text{if } G \geq 2; \end{cases}$$

$$p_m = \begin{cases} 0.39, & \text{if } G = 1; \\ \left(\frac{0.39}{1-G}\right) j + 0.4 \text{ for } j = 1, 2, \ldots, G-1, & \text{if } G \geq 2. \end{cases}$$

To choose σ in Algorithm 2 we use the parabolic model described in [33, p. 127].

For the GBB method we use the stopping criterion

$$\|\nabla f(x_k)\| \leq etol, \tag{19}$$

and for Algorithm 2 we use the stopping criterion given by the conditions (19) and

$$|f(x_k) - f(x_{k-1})| \leq ftol, \tag{20}$$

where $0 < etol \ll 1$ and $ftol = 10^{-10}$.

3.2 Test functions

Initially we compare the algorithms GBB and HSG using Function 11 of the Appendix A with $n = 2$. For the GA given in Algorithm 1 we set the population size $N = 2$, the number of generations $G = 3$ and $r = 5$ to build the box given by (8).

The Function 11 with $n = 2$ is defined as $f(x, y) = \sin^2(x^2 + y^2) + x^2 + y^2$, whose global minimum is $x_\star = (0, 0)^T$ with $f(x_\star) = 0$. This function possesses multiple stationary points.

In Table 1 the numerical results of GBB and HSG are shown for 12 initial points x_0. In this table we report the number of function evaluations (FE), CPU time in seconds (T) and the error

Table 1 Results of GBB and HSG for Function 11 with $n = 2$

x_0	GBB			HSG		
	FE	T	$e_\star$	FE	T	$e_\star$
$(0.5, 0.5)^T$	13	0.000	8.1e-8	65	0.016	1.1e-13
$(1, 1)^T$	12	0.000	1.1e-7	85	0.031	1.7e-13
$(10, 10)^T$	23	0.000	1.5e-7	185	0.047	1.3e-11
$(50, 50)^T$	11	0.000	2.4e-8	213	0.047	6.4e-12
$(100, 100)^T$	22	0.000	1.2e2	256	0.125	1.2e-14
$(200, 200)^T$	26	0.016	5.9e1	142	0.063	4.1e-14
$(-0.5, -0.5)^T$	13	0.000	8.1e-8	65	0.016	1.1e-13
$(-1, -1)^T$	12	0.000	1.1e-7	92	0.016	1.1e-12
$(-10, -10)^T$	23	0.000	1.5e-7	178	0.047	2.6e-12
$(-50, -50)^T$	11	0.000	2.4e-8	149	0.031	3.0e-12
$(-100, -100)^T$	22	0.000	1.2e2	87	0.031	5.6e-14
$(-200, -200)^T$	26	0.000	5.9e1	114	0.031	1.1e-12

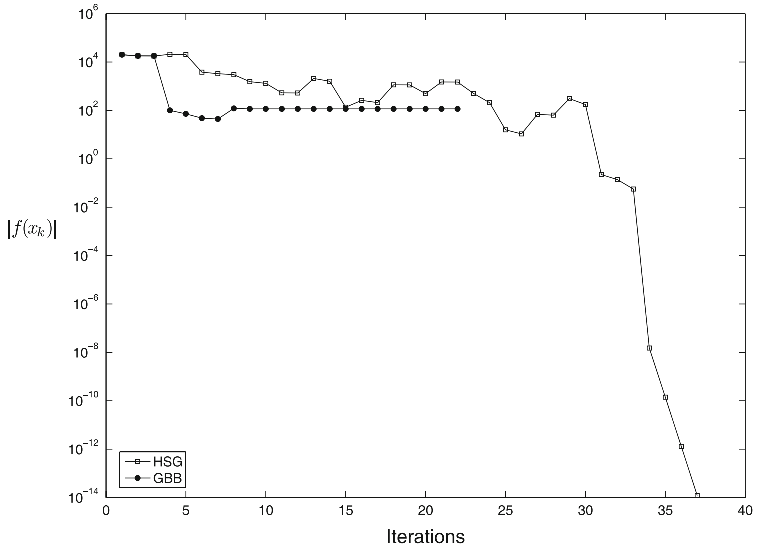

Fig. 1 Behavior of GBB and HSG for Function 11 and $x_0 = (100, 100)^T$

$$e_\star = |f(x_\star) - f(\overline{x})|, \tag{21}$$

where $f(x_\star)$ is the value of the objective function at the global minimum $x_\star$ and $f(\overline{x})$ is the value of the objective function at the solution $\overline{x}$ found by the algorithm.

In Figs. 1 and 2 we observe the behavior of GBB and HSG for Function 11 with $n = 2$ and the initial iterates $(100, 100)^T$ and $(200, 200)^T$, respectively. For these initial points, we observe that GBB converges to a local minimum and HSG converges to the global minimum.

We also observe that HSG finds the global minimum $x_\star$ for each one of the initial iterates. On the other hand, GBB finds the global minimum for almost all initial iterates, except for the initial iterates $(-100, -100)^T$, $(-200, -200)^T$, $(100, 100)^T$ and $(200, 200)^T$. We also

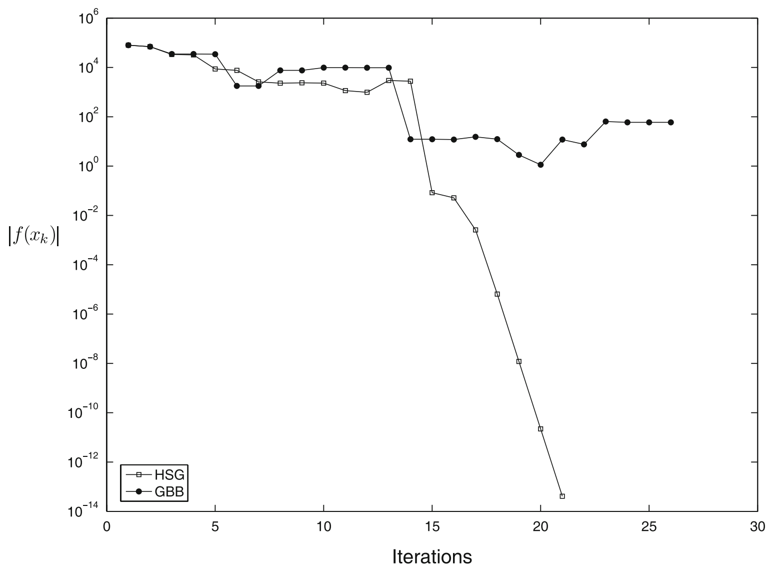

Fig. 2 Behavior of GBB and HSG for Function 11 and $x_0 = (200, 200)^T$

Table 2 Results of ga and HSG for Function 11 with $n = 2$

Interval	ga			HSG		
	FE	T	$e_\star$	FE	T	$e_\star$
[−0.5, 0.5]	5698	0.242	8.70e-9	56	0.018	1.42e-12
[−1, 1]	5169	0.219	4.84e-9	77	0.028	2.22e-12
[−10, 10]	5137	0.222	1.10e-8	110	0.041	4.72e-12
[−50, 50]	5318	0.232	2.94e-1	155	0.047	5.23e-12
[−100, 100]	5333	0.233	8.85e-1	174	0.050	6.67e-12
[−200, 200]	5202	0.232	8.79e0	194	0.053	9.77e-12
[−500, 500]	5370	0.238	4.39e1	214	0.058	2.86e-2

observe that GBB spends significantly less CPU time than HSG, that is due to the fact that the number of function evaluations of GBB is significantly smaller than those of HSG.

Now we compare the algorithms HSG and ga using Function 11 with $n = 2$. In Table 2 the numerical results of ga and HSG are shown for 7 search intervals. In this table we report the average number of function evaluations, the average CPU time, and the average error $e_\star$ given by (21), for 100 runs for each test function.

In this numerical experiment we observe that if the search interval is large, then the error obtained by ga increases significantly. For example, for the search interval [−0.5, 0.5] ga obtained the error $e_\star = 8.7\text{e-}9$ and for the search interval [−500, 500] the error is $e_\star = 43.9$. We also observe that the CPU time and the number of function evaluations of HSG are significantly smaller than those of ga. Also, the error obtained by HSG is mainly the same for all search intervals, except for the interval [−500, 500] for which the error obtained by HSG is $e_\star = 2.62\text{e-}2$. This last error can be diminished taking $ftol = 10^{-12}$ in (20). In this case HSG obtains an error $e_\star = 4.96\text{e-}14$ with an average of 0.063 seconds and 223 function evaluations.

Table 3 Results of HSG and RSG for different values of N and G

N	G	HSG			RSG			GBB		
		FE	T	$e_\star$	FE	T	$e_\star$	FE	T	$e_\star$
2	2	153	0.035	2.43e-11	177	0.005	2.08e3	55	0.007	1.16e3
2	5	333	0.090	2.73e-12	353	0.012	2.86e-1	53	0.006	7.25e1
2	10	500	0.183	3.92e-12	570	0.013	4.68e0	48	0.003	3.96e0
2	25	1491	0.968	3.41e-12	2684	0.059	7.43e-1	57	0.005	7.00e0
2	50	3160	3.554	3.04e-12	3382	0.075	6.00e-1	51	0.006	7.76e3
5	2	338	0.023	2.05e-11	499	0.016	1.86e0	58	0.005	9.94e0
5	5	779	0.053	2.07e-12	660	0.016	1.56e0	41	0.005	1.20e3
5	10	1613	0.119	1.09e-11	1916	0.044	5.47e0	52	0.002	1.50e0
5	25	5094	0.487	6.98e-11	2876	0.062	3.93e-12	51	0.003	5.88e0
5	50	7132	0.981	1.32e-11	9139	0.191	3.00e-1	56	0.005	6.67e0
10	2	587	0.039	4.01e-12	1241	0.030	3.24e0	55	0.003	3.26e1
10	5	1288	0.080	8.23e-12	1638	0.037	2.86e-1	62	0.006	3.55e1
10	10	3210	0.198	2.06e-11	7043	0.147	8.71e-1	49	0.000	7.87e3
10	25	7796	0.566	5.30e-11	6517	0.137	1.43e-1	71	0.003	6.21e1
10	50	15183	1.407	3.41e-11	15183	0.309	1.43e-1	45	0.002	6.09e1
25	2	1801	0.113	3.58e-11	1231	0.026	3.80e-12	61	0.005	2.92e2
25	5	4506	0.252	6.45e-11	3726	0.077	1.43e-1	53	0.003	1.33e2
25	10	6315	0.347	7.82e-12	10795	0.220	6.00e-1	67	0.005	4.96e0
25	25	38063	2.208	2.85e-10	14181	0.289	4.07e-12	60	0.002	1.92e1
25	50	48666	3.198	1.05e-10	25085	0.511	3.77e-12	55	0.005	2.27e3
50	2	3501	0.209	8.94e-11	3306	0.072	1.43e-1	57	0.004	1.41e2
50	5	6277	0.338	2.13e-11	6215	0.127	3.95e-12	49	0.004	2.86e1
50	10	17212	0.902	7.47e-11	11250	0.227	1.43e-1	65	0.005	1.34e0
50	25	32216	1.718	4.33e-11	28025	0.569	2.07e-12	52	0.004	8.93e3
50	50	72281	4.116	5.43e-11	53274	1.082	3.28e-12	52	0.006	4.98e0

Again we use Function 11 with $n = 2$ to see the efficacy of GA routine over the random search within the box in the HSG method. We call the algorithm *Random Spectral Gradient* (RSG) method which employs a simple random local search within a box trust region, instead of GA part of Algorithm 2. This local search evaluates the objective function at some number of randomly selected points in the box; for example, if $N = 2$ and $G = 5$, 10 points are selected. We set $[-500, 500]$ as the search interval.

Table 3 show the results of HSG, RSG and GBB, for 20 runs with each one of the values of the population size and the number of generations. We report in this table the average number of function evaluations, the average CPU time and the average error $e_\star$ given in (21). In each run we consider an initial iterate $x_0 = (x_0^1, x_0^2)^T$ randomly generated such that $x_0^1, x_0^2 \in [-500, 500]$.

We noticed that the population size and the number of generations affect the performance of HSG. When the population size and the number of generations are large, the number of function evaluations and the CPU time considerably increase. We also observe that solution quality of HSG is practically the same for all cases. On the other hand, these numerical results show that RSG and HSG possess similar CPU time and number of function evaluations, but the solution quality of HSG is significantly better. This fact allows to justify the use of GA as a local search scheme. We also observe that GBB not finds the global minimum in all the runs.

Subsequently we compare the behavior of GBB, HSG, `ga` and RSG for all test functions. In Tables 4 and 5 we report the average number of function evaluations, the average CPU time and the average error $e_\star$ given in (21), for 20 runs for each test function, where the global

Table 4 Results of GBB and HSG for the test functions

Function (n)	GBB				HSG			
	FE	T	$e_\star$	fbest	FE	T	$e_\star$	fbest
1(2)	463	0.019	8.3e-7	4.0e-1	820	0.204	3.6e-7	4.0e-1
2(2)	881	0.059	2.4e-4	−1.0e0	9332	2.141	3.4e-8	−1.0e0
3(2)	6380	0.170	5.3e1	3.0e0	11814	1.681	2.8e-11	3.0e0
4(2)	57	0.002	7.1e-1	4.7e-10	429	0.114	4.0e-1	9.0e-13
5(2)	116	0.003	7.2e-1	6.1e-8	423	0.120	1.2e-1	4.7e-8
6(2)	3733	0.158	8.8e-6	−1.9e2	5156	0.832	8.8e-6	−1.9e2
7(6)	968	0.096	4.2e-2	−3.3e0	3538	0.999	5.3e-1	−3.3e0
8(6)	11	0.000	3.0e-7	8.5e-8	137	0.058	3.9e-11	4.9e-12
9(5)	936	0.041	2.1e-7	1.0e-8	3260	0.652	1.3e-10	7.7e-17
9(10)	3730	0.163	2.0e-7	3.0e-11	10874	2.083	2.8e-9	6.7e-12
9(50)	328126	23.622	8.5e-4	3.2e-4	173746	29.286	3.4e-8	1.2e-16
10(5)	337	0.018	3.1e-7	7.8e-8	2071	0.459	3.0e-10	1.6e-10
10(10)	2177	0.177	3.3e-7	1.6e-7	14230	3.424	2.7e-10	2.2e-10
10(25)	3058	0.313	5.e-1	5.0e-1	350120	95.178	5.0e-1	5.0e-1
11(10)	72	0.009	5.6e0	2.7e-8	673	0.180	4.3e-1	8.6e-16
11(50)	128	0.012	2.8e1	7.0e-8	700	0.198	7.3e0	1.8e-14
11(100)	167	0.017	3.6e1	1.2e1	729	0.206	1.1e2	1.7e-12
12(100)	24	0.002	1.5e-7	1.0e2	186	0.062	4.1e-12	1.0e2
12(500)	25	0.004	1.5e-7	5.0e2	192	0.095	2.4e-12	5.0e2
12(1000)	24	0.003	1.4e-7	1.0e3	191	0.106	9.3e-13	1.0e3

Table 5 Results of `ga` and RSG for the test functions

Function (n)	`ga`				RSG			
	FE	T	$e_\star$	fbest	FE	T	$e_\star$	fbest
1(2)	5488	0.183	3.8e-7	4.0e-1	913	0.023	3.6e-7	4.0e-1
2(2)	4584	0.155	2.6e-5	−1.0e0	131013	3.646	3.0e-6	−1.0e0
3(2)	5766	0.193	4.1e0	3.0e0	15363	0.330	5.0e1	3.0e0
4(2)	5260	0.177	1.1e-9	0.0e0	366	0.010	7.6e-1	2.6e-11
5(2)	4968	0.176	1.3e-7	4.7e-8	396	0.012	4.8e-1	4.7e-8
6(2)	5210	0.233	8.7e-6	−1.9e2	4567	0.158	3.2e0	−1.9e2
7(6)	10345	0.659	6.6e-2	−3.3e0	3202	0.172	2.0e-1	−3.3e0
8(6)	9245	0.413	2.1e-9	0.0e0	131	0.003	3.8e-11	5.2e-12
9(5)	12591	0.696	8.4e-5	2.3e-8	2895	0.102	7.9e-10	6.4e-17
9(10)	11332	0.645	1.9e-6	7.1e-10	7273	0.255	2.5e-9	7.6e-12
9(50)	6319	0.602	5.4e-7	2.5e-8	197996	9.655	5.0e-8	1.3e-15
10(5)	9259	0.431	7.0e-5	9.0e-6	1164	0.041	2.8e-10	2.1e-10
10(10)	12219	0.598	2.3e-1	2.3e-8	14421	0.489	2.5e-10	1.9e-10
10(25)	28386	1.366	6.3e-1	5.0e-1	350128	21.925	5.0e-1	5.0e-1
11(10)	14146	0.634	1.4e1	2.3e-10	276	0.014	3.3e1	3.1e-15
11(50)	34878	2.824	7.9e2	2.9e0	420	0.028	1.9e1	7.1e-14
11(100)	48149	6.594	7.1e3	6.0e0	528	0.061	7.2e0	3.9e-15
12(100)	47514	3.156	2.5e-4	1.0e2	203	0.009	1.0e-11	1.0e2
12(500)	13672	51.913	1.9e-4	5.0e2	203	0.036	4.1e-12	5.0e2
12(1000)	5897	436.094	6.0e-5	1.0e3	191	0.069	9.3e-13	1.0e3

minimum $x_\star$ of each test function is shown in Appendix A. We also report in these tables the function number and dimension (Function(n)), and the smallest value in the objective function (fbest) reached by the algorithm in the 20 runs. In each run we consider an initial iterate $x_0 = (x_0^1, x_0^2, \ldots, x_0^n)^T$ randomly generated such that $x_0^j \in [\ell_i, \ell_s]$, where the interval $[\ell_i, \ell_s]$ is defined in Appendix A for each test function.

The numerical results of Tables 4 and 5 show that in general the error obtained by HSG is significantly smaller than the errors of GBB and `ga`. We also observe that the CPU time and the number of function evaluations of GBB are significantly smaller than those of HSG and `ga`. When comparing HSG and `ga` we observe that the CPU time and the number of function evaluations obtained by `ga` are bigger than those obtained by HSG. Also, for almost all the problems, HSG obtained the smallest value in the objective function. When `ga` obtained a better result, the difference among the best value in the objective function obtained by HSG and the value obtained by `ga` is not significant.

3.3 Molecular conformation problem

Of agreement to Meza and Martinez [48]:

> An important area of investigation in computational biochemical is the study of molecules for specific applications. Examples of such applications are: the development of enzymes for the elimination of toxic garbages, the development of new catalysts for the production of materials, and the study of new anti-cancerigenic agents. The development of these substances depends on the exact determination of the structure of the biological molecules. This problem is known as the molecular conformation problem that consists of finding the configuration of a molecule whose free energy is the lowest.
> Under the assumption that the native structure of a molecule corresponds to a conformation for which the energy is at or near the global minimum, the molecular conformation problem can be formulated as a problem of optimization. [48, pp. 627–628]

Next we describe the used model. In the same way in Meza and Martinez [48] we consider a two-dimensional polymer that consists of K atoms connected by rigid sticks. A potential E describing the energy of this system can be given by

$$E = \sum_{j=1}^{M} \sum_{i=j+1}^{K} \left(\|a_i - a_j\|^2 - d_{ij}^2 \right)^2, \tag{22}$$

where $a_i = (x^i, y^i)^T \in \mathbb{R}^2$ is the atom i with coordinates (x^i, y^i), d_{ij} is the known distance of the atom i to the atom j, and $1 \leq M \leq K - 1$. The integer M measures the number of distances among known atoms, i.e., the distance of the first M atoms to all the other atoms.

Since $\|a_i - a_j\|^2 = (x^i - x^j)^2 + (y^i - y^j)^2$, the problem of finding the minimum of E can be outlined as an unconstrained minimization problem of a function $f : \mathbb{R}^{2K} \to \mathbb{R}$ defined as

$$f(x) = \sum_{j=1}^{M} \sum_{i=j+1}^{K} \left((x^i - x^j)^2 + (y^i - y^j)^2 - d_{ij}^2 \right)^2, \tag{23}$$

where

$$x = (x^1, x^2, \ldots, x^K, y^1, y^2, \ldots, y^K)^T \in \mathbb{R}^{2K}.$$

It is easy to verify that $(x^1, x^2, \ldots, x^K, y^1, y^2, \ldots, y^K)^T \in \mathbb{R}^{2K}$ solves the problem if and only if $f(x^1, x^2, \ldots, x^K, y^1, y^2, \ldots, y^K) = 0$. Therefore, the structure of a molecule can be determined finding the global minimum of f given by (23). The objective is to find the position of each atom in the molecule satisfying all constraints imposed by the known

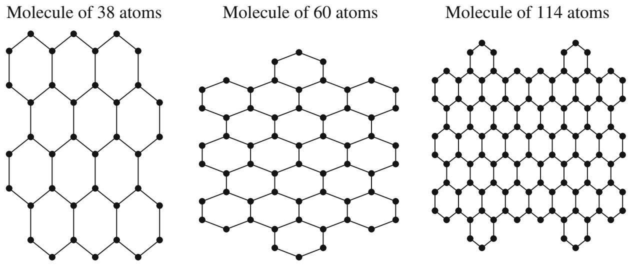

Fig. 3 Optimal conformation of the molecules of 38, 60 and 114 atoms

Table 6 Results of ga, GBB and HSG for the problems M38, M60 and M114

	ga			GBB			HSG		
	M38	M60	M114	M38	M60	M114	M38	M60	M114
FE	25019	25021	25023	100034	100048	76968	6278	7213	7625
T	28.18	56.92	81.20	105.63	169.80	392.47	13.60	22.02	54.59
$e_\star$	8.1e-10	1.3e-9	6.5e-10	4.9e-15	3.5e-12	2.3e-18	1.3e-15	5.4e-16	1.2e-16
fbest	2.2e-11	1.3e-11	9.6e-12	4.3e-15	1.5e-13	4.0e-19	1.3e-15	5.0e-16	1.2e-16

distance (i.e., that $\|a_i - a_j\| = \sqrt{(x^i - x^j)^2 + (y^i - y^j)^2} = d_{ij}$ for $i = 1, 2, \ldots, M$, $j = 1, 2, \ldots, K$). For some reference, see [42].

We consider three problems: M38, M60 and M114 that consist of molecules of 38, 60 and 114 atoms, respectively, and whose optimal conformation is shown in Fig. 3. For each run we randomly generate an initial iterate x_0 such that $x_0^j \in [-15, 15]$. For the numerical experiments we take $M = \lceil 0.6K \rceil$. For the GA given in Algorithm 1 we set the population size $N = 2$, the number of generations $G = 3$ and $r = 5$ to build the box given by (8).

In Table 6 we report the average number of function evaluations of f given in (23), the average CPU time, the average error $e_\star = |f(\overline{x})|$ and the value of f at the best solution of the algorithm, for each problem test, in 10 runs.

We observe that the CPU time and the number of function evaluations of HSG are significantly smaller than those of GBB and ga. Practically, the errors are all the same regardless of employed algorithms. Also, for all problems, GBB and HSG obtained the smallest value in the objective function. The difference among the best value in the objective function obtained by HSG and the value obtained by GBB is not significant.

4 Final remarks

We present a new method for unconstrained minimization denoted by Hybrid Spectral Gradient (HSG) Method. The HSG method can be considered as a hybrid GA that combines the method of Barzilai–Borwein with a GA. By its simplicity, the method is very easy to implement, it requires a minimum of storage, and for that reason, it is very attractive for the resolution of large-scale minimization problems (the code in MatLab written by the authors is available upon request).

We compare the numerical behavior of the methods HSG, GBB and the function ga of the Toolbox of Genetic Algorithms of MatLab, with a set of test functions and the molecular conformation problem. Our preliminary results indicate that the new methodology finds the global optimum more frequently than GBB and `ga`. For the test functions, the HSG method obtained, on average, the best solution quality. On the other hand, we also observed that all algorithms required a larger number of FEs and CPU times for the molecular conformation problem, but HSG obtained the best performance. The good behavior of HSG can be a consequence of the combination of the methods GBB and GA.

In the numerical comparison of GBB with HSG using Function 11 of the Appendix A, we noticed that HSG finds the global minimum for each initial iterate, and GBB converges to a stationary point of the objective function when the initial iterate is not close enough to the global minimum. This is because HSG combines the BB method with a GA to perform a local search about the current point whenever the BB method generates a new iterate. The incorporation of the GA can allow HSG to avoid the local minima and to converge to the global minimum.

Now, when we compare `ga` with HSG using Function 11, we obtained that HSG finds the global minimum for each search interval, and `ga` does not find the global minimum if the search interval is large. Since `ga` is a global search algorithm, the size of the search space affects their performance. On the other hand, since HSG requires only an initial iterate, the size of the search space does not affect its performance. What affects their performance is the location of the initial iterate with regard to the global optimum.

We numerically verify that the population size and the number of generations of GA can affect the performance of HSG. When the population size and the number of generations are large, the number of function evaluations and the CPU time increase, but the solution quality can be practically the same.

Lastly, an investigation topic that arises as a consequence of the present work, is the development of a derivative-free version of HSG, in other words, to design a new approach of HSG that does not use the gradient of f as search direction. The models without differentiability are in general very difficult to solve, although they are of great relevance in science and engineering.

Acknowledgements We are grateful to Marcos Raydan for many suggestions which greatly improved the quality and presentation of this paper. We are also indebted to two anonymous referees for valuable comments and suggestions. This work was supported by CDCH-UCV project PI-08-14-5463-2006.

A List of test functions

A.1 Branin RCOS function

- Definition: $f(x_1, x_2) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x^1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-5, 15]$, $i = 1, 2$.
- Number of local minima: no local minima.
- Global minima: $x_\star = (-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475)$; $f(x_\star) = 0.397887$.

A.2 Easom function

- Definition: $f(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-10, 10]$, $i = 1, 2$.
- Number of local minima: several local minima.
- Global minima: $x_\star = (\pi, \pi)$; $f(x_\star) = -1$.

A.3 Goldstein and Price function

- Definition: $f(x_1, x_2) = (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2))$ $(30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-2, 2]$, $i = 1, 2$.
- Number of local minima: 4 local minima.
- Global minima: $x_\star = (0, -1)$; $f(x_\star) = 3$.

A.4 Rastrigin function

- Definition: $f(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-1, 1]$, $i = 1, 2$.
- Number of local minima: many local minima.
- Global minima: $x_\star = (0, 0)$; $f(x_\star) = 0$.

A.5 Hump function

- Definition: $f(x_1, x_2) = 1.0316285 + 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-5, 5]$, $i = 1, 2$.
- Number of local minima: no local minima.
- Global minima: $x_\star = (0.0898, -0.7126)$, $(-0.0898, 0.7126)$; $f(x_\star) = 0$.

A.6 Shubert function

- Definition: $f(x_1, x_2) = \left(\sum_{j=1}^{5} j\cos((j+1)x_1 + j)\right)\left(\sum_{j=1}^{5} j\cos((j+1)x_2 + j)\right)$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-10, 10]$, $i = 1, 2$.
- Number of local minima: 760 local minima.
- Global minima: 18 global minima and $f(x_\star) = -186.7309$.

A.7 Hartmann function ($H_{6,4}$)

- Definition: $f(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$.

i	a_{ij}						c_i
1	10.0	3.0	17.0	3.50	1.70	8.0	1.0
2	0.05	10.0	17.0	0.10	8.0	14.0	1.2
3	3.0	3.5	1.7	10.0	17.0	8.0	3.0
4	17.0	8.0	0.05	10.0	0.1	14.0	3.2

i	p_{ij}					
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [0, 1]$, $i = 1, \ldots, 6$.
- Number of local minima: 6 local minima.
- Global minima: $x_\star = (0.201690, 0.150011, 0.476874, 0.275332, 0.311652, 0.657300)$ and $f(x_\star) = -3.32237$.

A.8 Griewank function

- Definition: $f(x) = \sum_{j=1}^{6} \frac{x_j^2}{4000} - \prod_{j=1}^{6} \cos(x_j/\sqrt{j}) + 1$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-1, 1]$, $i = 1, \ldots, 6$.
- Number of local minima: many local minima.
- Global minima: $x_\star = 0$ and $f(x_\star) = 0$.

A.9 Zakharov function

- Definition: $f(x) = \sum_{j=1}^{n} x_j^2 + \left(\sum_{j=1}^{n} 0.5 j x_j\right)^2 + \left(\sum_{j=1}^{n} 0.5 j x_j\right)^4$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-5, 10]$, $i = 1, \ldots, n$.
- Number of local minima: no local minima.
- Global minima: $x_\star = 0$ and $f(x_\star) = 0$.

A.10 Dixon function

- Definition: $f(x) = (1 - x_1)^2 + (1 - x_{10})^2 + \sum_{j=1}^{9} (x_i^2 - x_{i+1})^2$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-5, 10]$, $i = 1, \ldots, n$.
- Number of local minima: no local minima.
- Global minima: $x_\star = (1, 1, \ldots, 1)$ and $f(x_\star) = 0$.

A.11 Function 11

- Definition: $f(x) = \sum_{j=1}^{m} \left[\sin^2(x_{2j-1}^2 + x_{2j}^2) + (x_{2j-1}^2 + x_{2j}^2)\right]$, where $m = n/2$ and $n > 0$ is an even integer.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-50, 50]$, $i = 1, \ldots, n$.
- Number of local minima: several local minima.
- Global minima: $x_\star = 0$ and $f(x_\star) = 0$.

A.12 Strictly Convex function

- Definition: $f(x) = \sum_{j=1}^{n} \left[\exp(x_j) - x_j\right]$.
- Range of initial points $[\ell_i, \ell_s]$: $x_i \in [-3, 3]$, $i = 1, \ldots, n$.
- Number of local minima: no local minima.
- Global minima: $x_\star = 0$ and $f(x_\star) = n$.

References

1. Azofeifa, D., Clark, N., Vargas, W.: Optical and electrical properties of terbium films as a function of hydrogen concentration. Phys. Stat. B – Basic Solid State Phys. **242**, 2005–2009 (2005)
2. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. **8**, 141–148 (1988)
3. Bellavia, S., Morini, B.: A globally convergent Newton-GMRES subspace method for systems of nonlinear equations. SIAM J. Sci. Comput. **23**, 940–960 (2001)
4. Bello, L., Raydan, M.: Convex constrained optimization for the seismic reflection tomography problem. J. Appl. Geophys. **62**, 158–166 (2007)
5. Bielschowsky, R.H., Friedlander, A., Gomes, F.A., Martínez, J.M., Raydan, M.: An adaptive algorithm for bound constrained quadratic minimization. Invest. Oper. **7**, 67–102 (1997)
6. Birgin, E.G., Chambouleyron, I., Martínez, J.M.: Estimation of the optical constants and the thickness of thin films using unconstrained optimization. J. Comput. Phys. **151**, 862–880 (1999)

7. Birgin, E.G., Chambouleyron, I., Martínez, J.M.: Optimization problems in the estimation of parameters of thin films and the elimination of the influence of the substrate. J. Comput. Appl. Math. **152**, 35–50 (2003)
8. Birgin, E.G., Evtushenko, Y.G.: Automatic differentiation and spectral projected gradient methods for optimal control problems. Optim. Methods Softw. **10**, 125–146 (1998)
9. Birgin, E.G., Martínez, J.M., Mascarenhas, W.F., Ronconi, D.P.: Method of sentinels for packing items within arbitrary convex regions. J. Oper. Res. Soc. **57**, 735–746 (2006)
10. Birgin, E.G., Martínez, J.M., Nishihara, F.H., Ronconi, D.P.: Orthogonal packing of rectangular items within arbitrary convex regions by nonlinear optimization. Comput. Oper. Res. **33**, 3535–3548 (2006)
11. Birgin, E.G., Martínez, J.M., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. **10**, 1196–1211 (2000)
12. Birgin, E.G., Martínez, J.M., Raydan, M.: Algorithm 813: SPG—software for convex-constrained optimization. ACM Trans. Math. Softw. **27**, 340–349 (2001)
13. Birgin, E.G., Martínez, J.M., Raydan, M.: Inexact spectral projected gradient methods on convex sets. IMA J. Numer. Anal. **23**, 539–559 (2003)
14. Birgin, E.G., Martínez, J.M., Ronconi, D.P.: Minimization subproblems and heuristics for an applied clustering problem. Eur. J. Oper. Res. **146**, 19–34 (2003)
15. Birgin, E.G., Martínez, J.M., Ronconi, D.P.: Optimizing the packing of cylinders into a rectangular container: a nonlinear approach. Eur. J. Oper. Res. **160**, 19–33 (2005)
16. Brown, P.N., Saad, Y.: Hybrid Krylov methods for nonlinear systems of equations. SIAM J. Sci. Comput. **11**, 450–481 (1990)
17. Brown, P.N., Saad, Y.: Convergence theory of nonlinear Newton–Krylov algorithms. SIAM J. Optim. **4**, 297–330 (1994)
18. Castillo, Z., Cores, D., Raydan, M.: Low cost optimization techniques for solving the nonlinear seismic reflection tomography problem. Optim. Eng. **1**, 155–169 (2000)
19. Coleman, T.F., Li, Y.: On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. Math. Program. **67**, 189–224 (1994)
20. Coleman, T.F., Li, Y.: An interior trust region approach for nonlinear minimization subject to bounds. SIAM J. Optim. **6**, 418–445 (1996)
21. Cores, D., Fung, G.M., Michelena, R.J.: A fast and global tow point low storage optimization technique for tracing rays in 2D and 3D isotropic media. J. Appl. Geophys. **45**, 273–287 (2000)
22. Cores, D., Loreto, M.: A generalized two point ellipsoidal anisotropic ray tracing for converted waves. Optim. Eng. **8**, 373–396 (2007)
23. Curiel, F., Vargas, W.E., Barrera, R.G.: Visible spectral dependence of the scattering and absorption coefficients of pigmented coatings from inversion of diffuse reflectance spectral. Appl. Opt. **41**, 5969–5978 (2002)
24. Dai, Y.H.: On the nonmonotone line search. J. Optim. Theory Appl. **112**, 315–330 (2002)
25. Dai, Y.H.: Alternate step gradient method. Optimization **52**, 395–415 (2003)
26. Dai, Y.H., Fletcher, R.: On the asymptotic behaviour of same new gradient methods. Math. Program. **103**, 541–559 (2005)
27. Dai, Y.H., Fletcher, R.: Project Barzilai–Borwein methods for large-scale box-constrained quadratic programming. Numerische Mathematik **100**, 21–47 (2005)
28. Dai, Y.H., Hager, W., Schittkowski, K., Zhang, H.C.: The cyclic Barzilai–Borwein method for unconstrained optimization. IMA J. Numer. Anal. **26**, 604–627 (2006)
29. Dai, Y.H., Liao, L.Z.: R-linear convergence of the Barzilai and Borwein gradient method. IMA J. Numer. Anal. **22**, 1–10 (2002)
30. Dai, Y.H., Yuan, J.Y., Yuan, Y.X.: Modified two-point stepsize gradient methods for unconstrained optimization. Comput. Optim. Appl. **22**, 103–109 (2002)
31. Dai, Y.H., Zhang, H.C.: Adaptive two-point stepsize gradient algorithm. Numer. Algorithms **27**, 377–385 (2001)
32. Deidda, G.P., Bonomi, E., Manzi, C.: Inversion of electrical conductivity data with tikhonov regularization approach: some considerations. Ann. Geophys. **46**, 549–558 (2003)
33. Dennis, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice-Hall, Englewoog Cliffs (1983)
34. Davis, L.: The Handbook of Genetic Algorithms. Van Nostrand Reingold, New York (1991)
35. Fletcher, R.: Low storage methods for unconstrained optimization. Lectures in Applied Mathematics (AMS) **26**, 165–179 (1990)
36. Fletcher, R.: On the Barzilai–Borwein method. Appl. Optim. **96**, 235–236 (2006)
37. Francisco, J.B., Martínez, J.M., Martínez, L.: Density-based globally convergent trust-region methods for self-consistent field electronic structure calculations. J. Math. Chem. **40**, 349–377 (2006)

38. Friedlander, A., Martínez, J.M., Molina, B., Raydan, M.: Gradient method with retards and generalizations. SIAM J. Numer. Anal. **36**, 275–289 (1999)
39. Friedlander, A., Martínez, J.M., Raydan, M.: A new method for large-scale constrained convex quadratic minimization problems. Optim. Methods Softw. **5**, 57–74 (1995)
40. Glunt, W., Hayden, T.L., Raydan, M.: Molecular conformation from distance matrices. J. Comput. Chem. **14**, 114–120 (1993)
41. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Readomg (1989)
42. Hendrickson, B.A.: The molecule problem: exploiting structure in global optimization. SIAM J. Optim. **5**, 835–857 (1995)
43. Holland, J.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
44. James, C.S.: Introduction to Stochastics Search and Optimization. Wiley-Interscience, New Jersey (2003)
45. La Cruz, W.: Derivative-free residual algorithm for solving weakly nonlinear equations. IMA J. Numer. Anal. (2007), Submitted
46. La Cruz, W., Martínez, J.M., Raydan, M.: Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. Math. Comput. **75**, 1449–1466 (2006)
47. La Cruz, W., Raydan, M.: Nonmonotone spectral methods for large-scale nonlinear systems. Optim. Methods Softw. **18**, 583–599 (2003)
48. Meza, J.C., Martinez, M.L.: On the use of direct search methods for the molecular conformation problem. J. Comput. Chem. **15**, 627–632 (1994)
49. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. AI Series. Springer-Verlag, New York (1994)
50. Molina, B., Raydan, M.: Preconditioned Barzilai–Borwein method for the numerical solution of partial differential equations. Numer. Algorithms **13**, 45–60 (1996)
51. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**, 308–313 (1965)
52. Ramirez-Porras, A., Vargas-Castro, E.: Transmission of visible light through oxidized copper films: feasibility of using a spectral projected gradient method. Appl. Opt. **43**, 1508–1514 (2004)
53. Raydan, M.: On the Barzilai and Borwein choice of steplength for the gradient method. IMA J. Numer. Anal. **13**, 321–326 (1993)
54. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. **7**, 26–33 (1997)
55. Serafini, T., Zanghirati, G., Zanni, T.: Gradient projection methods for quadratic programs and applications in training support vector machines. Optim. Methods Softw. **20**, 353–378 (2005)
56. Vargas, W.E.: Inversion methods from Kubelka–Munk analysis. J. Opt. A - Pure Appl. Opt. **4**, 452–456 (2002)
57. Vargas, W.E., Azofeifa, D.E., Clark, N.: Retrieved optical properties of thin films on absorbing substrates from transmittance measurements by application of a spectral projected gradient method. Thin Solid Films **425**, 1–8 (2003)
58. Wells, C., Glunt, W., Hayden, T.L.: Searching conformational space with the spectral distance geometry algorithm. J. Mol. Struct. (Theochem) **308**, 263–271 (1994)
59. Wright A.H.: Genetic algorithms for real parameter optimization. In: Rawlins J.E. (ed.) Foundations of Genetic Algorithms, pp. 205–218. Morgan Kaufmann (1991)
60. Zeev, N., Savasta, O., Cores, D.: Non-monotone spectral projected gradient method applied to full waveform inversion. Geophys. Prospect. **54**, 525–534 (2006)
61. Zhina, J.: Applications of conditional nonlinear optimal perturbation to the study of the stability and sensitivity of the Jovian atmosphere. Adv. Atmos. Sci. **23**, 775–783 (2006)