Journal of Advances in Mathematics and Computer Science

26(5): 1-20, 2018; Article no.JAMCS.39798

ISSN: 2456-9968

(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

A Heuristic Fast Gradient Descent Method for Unimodal Optimization

George Anescu[1*]

[1] *Power Plant Engineering Faculty, Polytechnic University of Bucharest, Romania.*

Author's contribution

The sole author designed, analyzed and interpreted and prepared the manuscript.

Article Information

DOI: 10.9734/JAMCS/2018/39798

Editor(s):

(1) Jacek Dziok, Professor, Institute of Mathematics, University of Rzeszow, Poland.

Reviewers:

(1) Ali Durmus, Erciyes University, Turkey.

(2) Hongli Yang, Mathematics and Systems Science College, Shandong University of Science and Technology, China.

Complete Peer review History: http://sciencedomain.org/review-history/23378

Original Research Article

Received: 31st December 2017

Accepted: 20th February 2018

Published: 28th February 2018

Abstract

The known gradient descent optimization methods applied to convex functions are using the gradient's magnitude in order to adaptively determine the current step size. The paper is presenting a new heuristic fast gradient descent (*HFGD*) approach, which uses the change in gradient's direction in order to adaptively determine the current step size. The new approach can be applied to solve classes of unimodal functions more general than the convex functions (e.g., quasi-convex functions), or as a local optimization method in multimodal optimization. Testing conducted on a testbed of 16 test functions showed an overall much better efficiency and an overall better success rate of the proposed *HFGD* method when compared to other three known first order gradient descent methods.

Keywords: Optimization; unimodal/multimodal functions; convex optimization; quasi-convex optimization; golden ratio; fibonacci sequence; first-order optimization algorithms; Heuristic Fast Gradient Descent (HFGD); Backtracking Line Search (BLS); Accelerated Gradient Descent (AGD); Fast Iterative Shrinkage/Thresholding Algorithm (FISTA).

2010 Mathematics Subject Classification: 68T 20, 68W 10, 90C 26, 90C 56, 90C 59.

**Corresponding author: E-mail: george.anescu@gmail.com;*

1 Introduction

The general optimization problem can be formulated as:

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) \\ &\text{subject to} \quad \mathbf{x} \in D, \end{aligned} \tag{1}$$

where $\mathbf{x}$ is a real vector of decision variables, $D \subset \mathbb{R}^n$ is a subset of the n-dimensional real vector space and $f : D \to \mathbb{R}$ is a real value objective function. Usually, D is defined as a boxed domain:

$$D = \{\mathbf{x} : \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}, \tag{2}$$

where $\mathbf{l}$ and $\mathbf{u}$ are explicit, finite, component-wise lower and upper bounds on $\mathbf{x}$. By convention, only the minimization problems are studied, considering that the maximization problem for $f(\mathbf{x})$ is equivalent to, and can be treated as, the minimization problem for $-f(\mathbf{x})$. In most cases the function $f(\mathbf{x})$ is at least continuous on D, although other stronger restrictions can be imposed, such as smoothness: the function $f(\mathbf{x})$ is differentiable with continuous derivatives; or piecewise smoothness: D can be broken into distinct pieces and on each piece the function $f(\mathbf{x})$ is differentiable with continuous derivatives.

Since the present study is concerned with unimodal functions, a rigorous approach to modes and unimodality is required, and it will be the object of Section 2. Section 3 will present some important subsets of unimodal functions and will introduce some interesting optimization function examples. Section 4 will present the design details of the proposed Heuristic Fast Gradient Descent (*HFGD*) algorithm. Section 5 will present the set of test optimization problems used in the testing experiments and the statistical results obtained by comparing the proposed *HFGD* method with other known first-order gradient descent methods: Backtracking Line Search (*BLS*, [1]) and two accelerated variants of *BLS*: Nesterov's Accelerated Gradient Descent (*AGD*, [2], [3]) and Fast Iterative Shrinkage-Thresholding Algorithm (*FISTA*, [4]). Finally, Section 6 will summarize the results of the paper, draw the conclusions and indicate some further research directions.

2 Introduction to Unimodal Functions

The presentation in this section is based on the definitions introduced in [5], with the main results of the referenced paper presented without repeating the demonstrations. First, the objective function $f(\mathbf{x})$ is extended to $f^{ext} : \mathbb{R}^n \to (-\infty, +\infty)$ defined as:

$$f^{ext}(\mathbf{x}) = \begin{cases} f(\mathbf{x}), \mathbf{x} \in D \\ +\infty, \mathbf{x} \notin D \end{cases} \tag{3}$$

For simplification of the notations, in the presentation that follows it will be assumed that function $f(\mathbf{x})$ is already extended, so that the notation $f(\mathbf{x})$ is further used instead of $f^{ext}(\mathbf{x})$. For an objective function $f(\mathbf{x})$, the point $\mathbf{x}^{**}$ satisfying

$$f(\mathbf{x}^{**}) \leq f(\mathbf{x}), \forall \mathbf{x} \in \mathbb{R}^n \tag{4}$$

is called the global minimum point. The point $\mathbf{x}^*$, for which there exists an open sphere $B(\mathbf{x}^*; \epsilon_1)$ of radius $\epsilon_1 > 0$ with $\mathbf{x}^*$ as center, and satisfying

$$f(\mathbf{x}^*) \leq f(\mathbf{x}), \forall \mathbf{x} \in B(\mathbf{x}^*; \epsilon_1), \tag{5}$$

is called a local minimum point. If the objective function's values in neighboring points of a local minimum point, other than the local minimum point, are larger than the value in the minimum point, i.e. there exists an open sphere $B(\mathbf{x}_s^*; \epsilon_2)$ of radius $\epsilon_2 > 0$ with $\mathbf{x}_s^*$ as center, such that

$$f(\mathbf{x}_s^*) < f(\mathbf{x}), \forall \mathbf{x} \in B(\mathbf{x}_s^*; \epsilon_2), \mathbf{x} \neq \mathbf{x}_s^*, \tag{6}$$

then the point $\mathbf{x}_s^*$ is called a local minimum point in the narrow sense. The objective functions having only minimum points in the narrow sense are easier to handle, and their local minimum points in the narrow sense are called modes, while the objective functions with only one local minimum point in the narrow sense are known as unimodal functions. For unimodal functions, any minimum mode found is guaranteed to be the global minimum mode. A difficulty appears when the objective function has flat (or plateau) regions, i.e. compact subsets of D for which f($\mathbf{x}$) is constant, in which case the set of local minimum points can be extended to the flat regions, but not all the flat regions can be considered as modes. A very good example of such a function is given in [5], and its graph representation is reproduced here in Fig. 1.

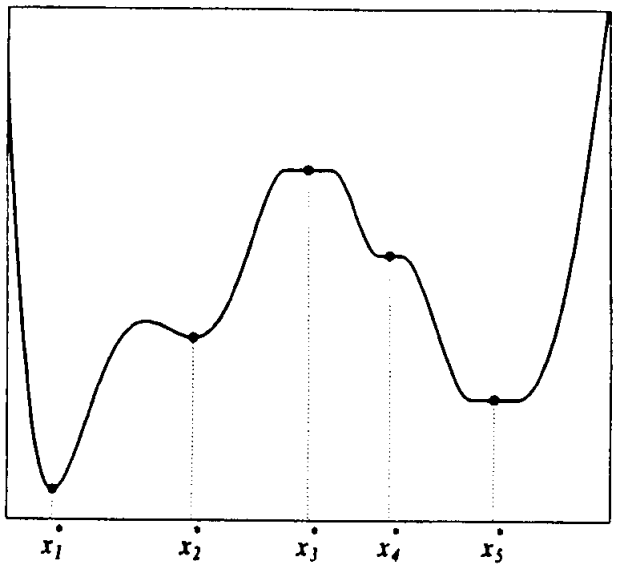

Fig. 1. Flat regions function example ([5])

The function in Fig. 1 has 2 local minimum points in the narrow sense, x_1^* and x_2^*, and 3 minimum point flat regions, x_3^*, x_4^* and x_5^*, but only x_1^*, x_2^* and x_5^* can be considered modes, while x_3^* and x_4^* are only stationary points, although they satisfy the definition of local minimum points. Furthermore, the points x_3^* in the flat region are local maximum points. Intuitively, a unimodal function has only one minimum point or minimum flat region and the rest of the graph goes up from there, but a more rigorous mathematical definition is needed in order to cover all the flat region cases, and it was the object of study in [5]. In the referenced paper [5] the concept of minimal value set is introduced, and based on the number of connected components of the minimal value set, a function is considered unimodal when the number of connected components is 1, and multimodal when he number of connected components > 1. But in order to rigorously introduce the concept of minimal value set, some preliminary definitions are needed ([5]):

Definition 1: Level set of function $f(\mathbf{x})$ on α.

For an objective function f($\mathbf{x}$) on $\alpha \in R$, the level set $L(\alpha) \subset \mathbb{R}^n$, and respectively the minor level set $L^s(\alpha)$, are defined as follows:

$$L(\alpha) = \{\mathbf{x} | f(\mathbf{x}) \leq \alpha, \mathbf{x} \in \mathbb{R}^n\}, \tag{7}$$

$$L^s(\alpha) = \{\mathbf{x}|f(\mathbf{x}) < \alpha, \mathbf{x} \in \mathbb{R}^n\}. \tag{8}$$

By the assumption that $f(\mathbf{x})$ is continuous and extended (according to (3)), $L(\alpha)$ is always a closed set.

Definition 2: Connected level set $L_c(\alpha; \mathbf{x})$ of $\mathbf{x}$:

The subset of the level set $L(\alpha)$ which is a connected component containing $\mathbf{x}$ is called the connected level set containing $\mathbf{x}$, and is written as $L_c(\alpha; \mathbf{x})$. As an illustration, see Fig. 2 reproduced from [5]. The subset of the level set $L(f(\mathbf{x}))$ which is a connected level set containing $\mathbf{x}$ is written as $L_c(f(\mathbf{x}))$. As an illustration, see Fig. 3 reproduced from [5]. Similarly, the subset of the minor level set $L^s(\alpha)$ which is a connected component containing $\mathbf{x}$ is called the connected minor level set containing $\mathbf{x}$, and is written as $L_c^s(\alpha; \mathbf{x})$.

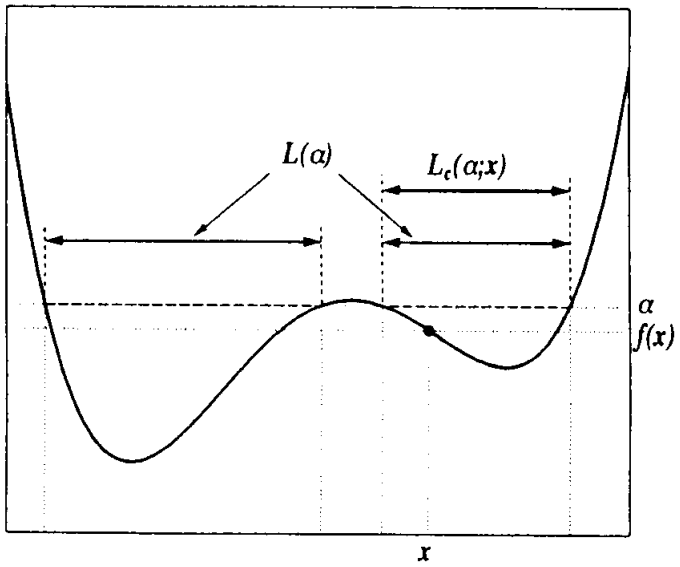

Fig. 2. Connected level set $L_c(\alpha; \mathbf{x})$ example ([5])

Definition 3: The minimal value set $E_c(f(\mathbf{x}^*))$ for function value $f(\mathbf{x}^*)$ at point $\mathbf{x}$:

If there exists $\mathbf{x}^* \in \mathbb{R}^n$ and the condition

$$L(f(\mathbf{x}^*) - \epsilon) \cap L_c(f(\mathbf{x}^*)) = \varnothing \tag{9}$$

is satisfied for any $\epsilon > 0$, the above $L_c(f(\mathbf{x}^*))$ is called the minimal value set at $\mathbf{x}^*$, and it is written as $E_c(f(\mathbf{x}^*))$. Figs. 4 and 5, reproduced from [5], show the cases where $E_c(f(\mathbf{x}^*))$ is the minimal value set, and respectively $E_c(f(\mathbf{x}^*))$ is not the minimal value set.

Definition 4: Minimal value set E^*:

The set of all $\mathbf{x}^*$ that satisfy Eq. (9) is called the minimal value set E^*. In other words, the minimal value set E^* is represented as follows:

$$E^* = \{\mathbf{x}^*|L(f(\mathbf{x}^*) - \epsilon) \cap L_c(f(\mathbf{x}^*)) = \varnothing, \forall \epsilon > 0, \mathbf{x}^* \in \mathbb{R}^n\}. \tag{10}$$

Definition 5: Modality of function $f(\mathbf{x})$, unimodal (minimal) function and multimodal (minimal) function.

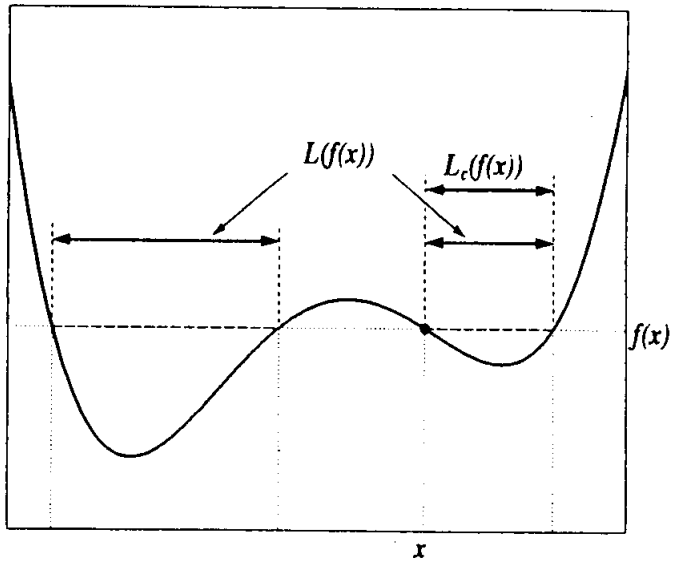

Fig. 3. Connected level set $L_c(\alpha; \mathbf{x})$ example ([5])

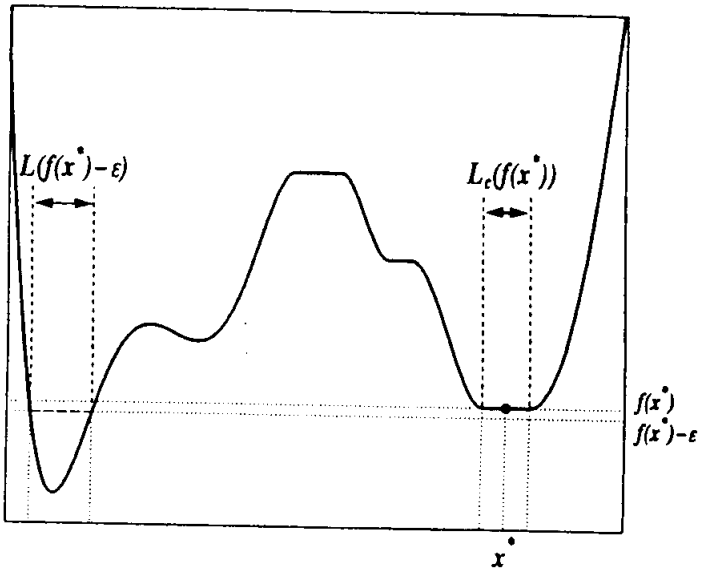

Fig. 4. The case where $\mathbf{x}^*$ is the minimal value set ([5])

For a function $f : \mathbb{R}^n \to (-\infty, +\infty)$, the number $N_c^* = |E^*|$ of connected components of the minimal value set E^* (called minimal value set components) is called the (lower) modality of the function $f(\mathbf{x})$. Based on this definition, the function with $N_c^* = 1$ is defined as the unimodal (minimal) function, and the function with $N_c^* > 1$ is defined as the multimodal (minimal) function.

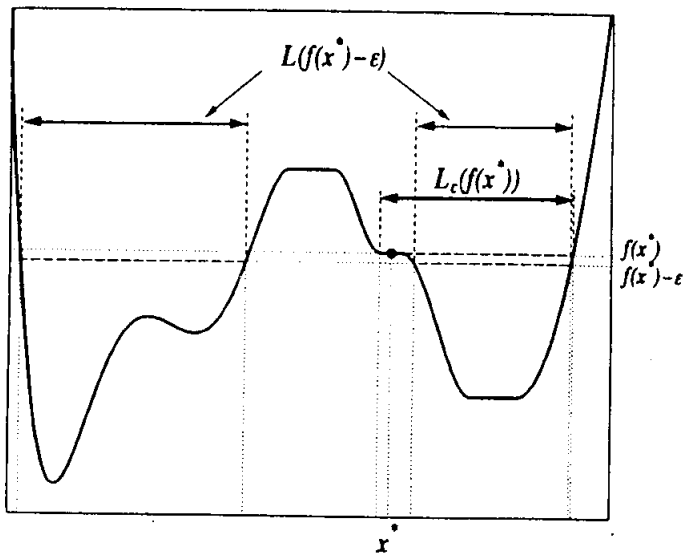

Fig. 5. The case where x^* is not the minimal value set ([5])

Two important Theorems are demonstrated in [5]: (1) The set of local minimum points includes the minimal value set, and (2) The minimal value set includes the set of minimum points in the narrow sense. The presented theoretical results allow to rigorously discriminate among the different flat regions (with minimum points) and to classify them in: (1) minimum modes, (2) maximum modes, and (3) flat regions that are neither minimum modes, nor maximum modes. For the defined optimization problem (1) only the flat regions of type (1) are of interest.

3 Important Subsets of Unimodal Functions

The most researched subset, from the general set of unimodal functions, is the one formed by the convex functions. In the conditions of definition (1), the function f($\mathbf{x}$) is convex if it satisfy the additional condition:

$$f(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1-\lambda) f(\mathbf{x}_2), \forall \mathbf{x}_1, \mathbf{x}_2 \in D, \forall \lambda \in [0,1]. \tag{11}$$

In order to eliminate the possibility of a flat region, the additional condition for a strictly convex function with a minimum point in the narrow sense is:

$$f(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) < \lambda f(\mathbf{x}_1) + (1-\lambda) f(\mathbf{x}_2), \forall \mathbf{x}_1 \neq \mathbf{x}_2 \in D, \forall \lambda \in (0,1). \tag{12}$$

There are many known methods to solve convex optimization problems, first order methods (based only on the first order derivatives, which should exist) and second order methods (based on the second order derivatives, which should exist). A less restrictive condition than (11) is generating a wider subset of unimodal functions, the quasi-convex functions:

$$f(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \max\{f(\mathbf{x}_1), f(\mathbf{x}_2)\}, \forall \mathbf{x}_1, \mathbf{x}_1 \in D, \forall \lambda \in [0,1], \tag{13}$$

and the corresponding condition for strictly quasi-convex functions is:

$$f(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) < \max\{f(\mathbf{x}_1), f(\mathbf{x}_2)\}, \forall \mathbf{x}_1 \neq \mathbf{x}_1 \in D, \forall \lambda \in (0,1). \tag{14}$$

For a comprehensive study of the quasi-convex functions see [6]). The meaning of strict quasi-convexity is that along any stretch of the curve $f(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2)$, $0 \leq \lambda \leq 1$, the highest point is one of the endpoints. An alternative definition is to require that the level curves:

$$C_\alpha = \{\mathbf{x} : f(\mathbf{x}) = \alpha, \alpha \in \mathbb{R}\} \tag{15}$$

are convex curves. Another important Theorem, demonstrated in [5], is that the quasi-convex functions are unimodal functions, and so they are a subset of the unimodal functions. Due to the inequality:

$$\lambda f(\mathbf{x}_1) + (1-\lambda) f(\mathbf{x}_2) \leq \max\{f(\mathbf{x}_1), f(\mathbf{x}_2)\}, \tag{16}$$

it is obvious that all the convex functions are quasi-convex functions, and therefore the quasi-convexity can be considered a generalization of the convexity (the subset of convex functions is including the subset of quasi-convex functions). The reciprocal is not true and, as a counter-example, the quasi-convex unidimensional function presented in Fig. 6, which has regions of concavity, can be considered.

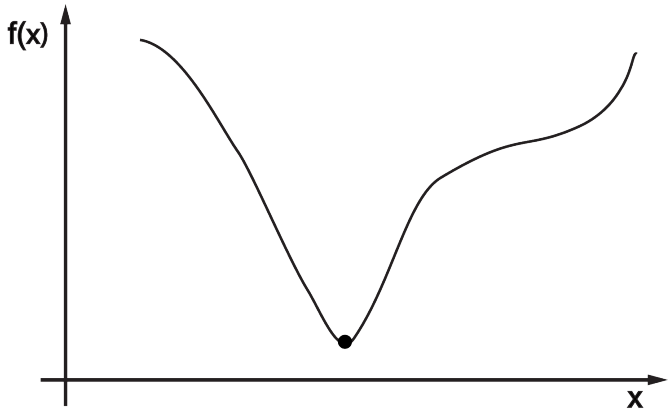

Fig. 6. Quasi-convex function example

For this reason, first order and second order optimization methods designed for convex functions are usually not successful when applied to quasi-convex functions. With the subset of quasi-convex functions the general set of unimodal functions are not exhausted. Not all the unimodal functions are quasi-convex, and in order to prove it a simple counter-example is sufficient. Consider the function:

$$\begin{gathered} f_{15}(\mathbf{x}) = \sum_{j=1}^{n} x_j^4 + 16 \sum_{j=1}^{n} x_j^2 x_{j+1}^2, \\ -2 \leq x_j \leq 2, \;\; j = 1, \ldots, n, \;\; x_{n+1} = x_1, \end{gathered} \tag{17}$$

with its 2-dimensional case being presented in Fig. 7.

This function is obviously unimodal with the minimum of 0 in $(0, 0, \ldots, 0)$. But if we consider the points $\mathbf{x}_1 = (1, 0, 0, \ldots, 0)$, $\mathbf{x}_2 = (0, 1, 0, \ldots, 0)$ and $\mathbf{x}_3 = \frac{\mathbf{x}_1 + \mathbf{x}_2}{2} = (0.5, 0.5, 0, \ldots, 0)$, we have $f(\mathbf{x}_3) = \frac{1}{16} + \frac{1}{16} + 16 \times \frac{1}{16} = 1 + \frac{1}{8} > 1 = \max\{1, 1\} = \max\{f(\mathbf{x}_1), f(\mathbf{x}_2)\}$, which proves that $f(\mathbf{x})$ is not quasi-convex. For a comprehensive introduction in the field of convex optimization see [3].

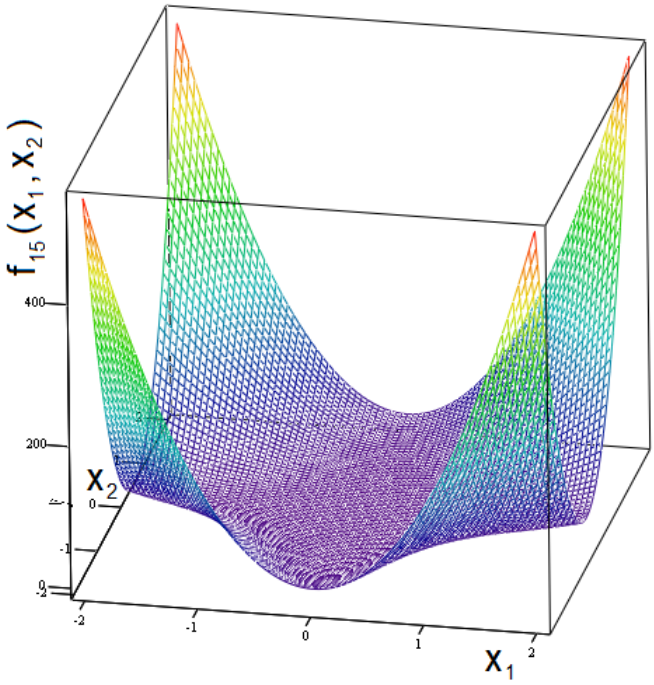

Fig. 7. Example of non-quasi-convex unimodal function

4 The Heuristic Fast Gradient Descent Algorithm

The Heuristic Fast Gradient Descent (*HFGD*) algorithm is designed in order to approach general unimodal optimization functions with the important assumption (restriction) that the functions should not have flat regions of type (3): i.e. flat regions that are neither minimum modes, nor maximum modes (according o the classification in Section 2). Some simple strategies can be devised in order to approach the first two types of flat regions. For example, for the flat regions of type (2) (maximum modes), the search algorithm can maintain the search direction and the search step constant (to the last calculated values) when a flat region condition is detected (based on 0 derivatives), and eventually it will be able to escape from the flat region in a finite number of steps. For flat regions of type (1) (minimum modes) the search algorithm can also maintain the search direction and the search step constant (to the last calculated values) when a flat region condition is detected, and if after a finite (large enough) number of steps it cannot escape from the flat region, it can decide that the flat region is a minimum mode. The problem with the flat regions of type (3) is that the escape path can be very narrow and difficult to reach, even after a large number of steps. Therefore it can become difficult to decide if a flat region is of type (1), or of type (3). In order to clarify these aspects a function example was specially designed:

$$f_{16}(\mathbf{x}) = -(n+1)e^{-10\sqrt{n}\left[\sum_{j=1}^{n}(x_j-1)^2\right]^{1/2}} + \max\{\sum_{j=1}^{n}x_j^2, n+\frac{0.01}{n}-0.2\} \tag{18}$$

$$-2 \le x_j \le 2, \;\; j = 1, \ldots, n,$$

the 2-dimensional case of it being represented in Fig. 8.

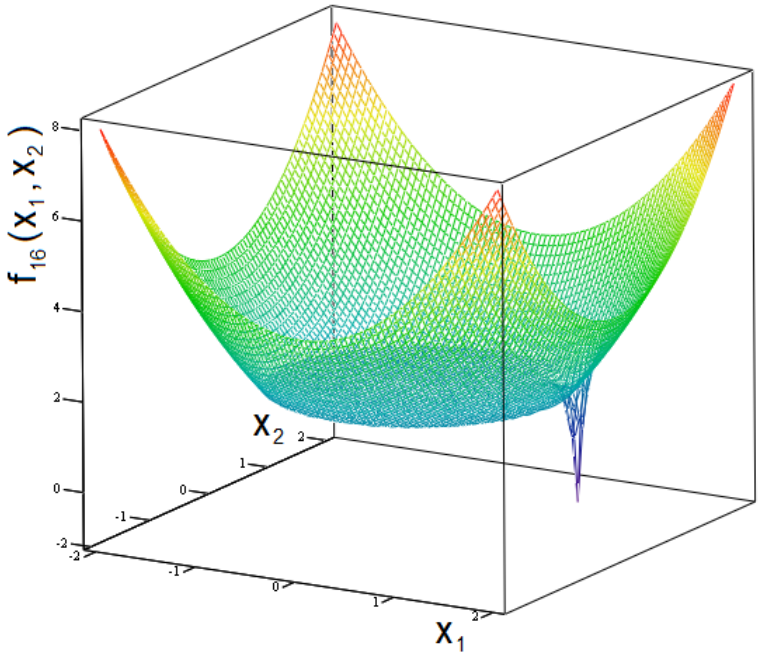

Fig. 8. Example of quasi-convex unimodal function with flat region of type (3))

This function is unimodal, according to the definitions in Section 2, with the minimum of -1 in $(1, 1, \ldots, 1)$, but it can be seen from Fig. 8 that the escape path to the minimum point is very narrow.

Another important assumption on *HFGD* method is that it should be able to correctly calculate the steepest descent direction (gradient), or a good numerical approximation to it.

The main idea of the *HFGD* algorithm is to obtain a faster convergence to the optimization solution by using an adaptive step size, which is updated (increased or decreased) dependent of the change in gradient's direction. The main steps of the *HFGD* algorithm are summarized below in Algorithm 1 and the algorithmic steps are further detailed.

Step 1: Set the computing precision ϵ, which is used as the termination condition for breaking the while loop. Choose $\varphi \in [1.5, 2]$, with the recommended value $\varphi = \frac{\sqrt{5}+1}{2}$, the Golden Ratio Number.

Algorithm 1 *HFGD* algorithm

1: Set ϵ and φ
2: Initialization for $k = 0$: Set $\mathbf{x}_0$ and γ_1
3: Set $k = 1$
4: **while** $\gamma_k < \epsilon$ **do**
5: Update γ_{k+1}
6: Update the solution $\mathbf{x}_{k+1}$
7: Increment $k := k + 1$

Step 2: Initialization for $k = 0$: Choose γ_1 in a uniform pseudo-random manner from the real interval $[0.2, 0.5]$. Choose the initial position $\mathbf{x}_0$ in a uniform pseudo-random manner in the limiting box D defined by (2). Compute the unit vector in gradient's direction:

$$\mathbf{n}_0 = \frac{\boldsymbol{\nabla} f(\mathbf{x}_0)}{||\boldsymbol{\nabla} f(\mathbf{x}_0)||_2}. \tag{19}$$

Step 5: Compute the unit vector:

$$\mathbf{n}_k = \frac{\boldsymbol{\nabla} f(\mathbf{x}_k)}{||\boldsymbol{\nabla} f(\mathbf{x}_k)||_2}. \tag{20}$$

Compute the scalar product as a measure of change in gradient's direction :

$$p_k = \mathbf{n}_{k-1} \cdot \mathbf{n}_k, \tag{21}$$

with $-1 \leq p_k \leq 1$. Compute $\varphi_k = \phi(p_k)$, with the function $\phi(p)$ defined by:

$$\phi(p) = \left[\frac{1}{2}\left(\varphi + \frac{1}{\varphi}\right) - 1\right] p^2 + \frac{1}{2}\left(\varphi - \frac{1}{\varphi}\right) p + 1, \tag{22}$$

the function $\phi(p)$ being derived by polynomial (Lagrange) interpolation in 3 points: $\phi(-1) = \frac{1}{\varphi}$, $\phi(0) = 1$, $\phi(1) = \varphi$ (see Fig. 9).

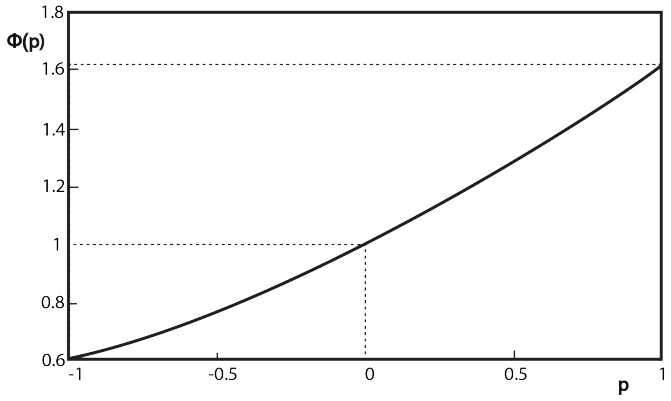

Fig. 9. The function $\phi(p)$

Update the method's step γ_{k+1}:

$$\gamma_{k+1} = \varphi_k \gamma_k. \tag{23}$$

The main idea in Step 5 is to make the updated γ_{k+1} dependent on the change in gradient's direction. If the angle of change in gradient's direction, $\theta_k = \arccos(p_k)$, is less than the right

angle ($\theta_k < \pi/2$), then there should be an increase in γ_{k+1}, but if it is greater than the right angle ($\theta_k > \pi/2$), then there should be a decrease in γ_{k+1}, with $\gamma_{k+1} = \gamma_k$ maintained unchanged for $\theta_k = \pi/2$. Also, if there is no change in gradient's direction, then the increase in γ_{k+1} should be maximal, $\gamma_{k+1} = \varphi\gamma_k$, while if there is a change in the opposed direction the decrease in γ_{k+1} should be maximal, $\gamma_{k+1} = \gamma_k/\varphi$. The proposed $\phi(p_k)$ function satisfies all these conditions and provides a good fitting for the other intermediate values of the angle of change in gradient's direction.

Step 6: Update the solution $\mathbf{x}_{k+1}$:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \gamma_{k+1}\mathbf{n}_k. \tag{24}$$

If $f(\mathbf{x}_{k+1}) > f(\mathbf{x}_k)$, then the new proposed solution $\mathbf{x}_{k+1}$ is not taken, γ_{k+1} is reset to the previous value:

$$\gamma_{k+1} = \gamma_k, \tag{25}$$

and the update equation (24) is applied again. This resetting logic is applied not more than once for the current iteration.

Note on the Golden Ratio Number: As mentioned above at *Step 1*, the *HFGD* optimization method works well for $\varphi \in [1.5, 2]$, but it was found experimentally that the method is more efficient (faster convergence) when $\varphi = \frac{\sqrt{5}+1}{2} \approx 1.618034$, the Golden Ratio Number (Golden Section, Golden Proportion, Golden Cut, etc.), which is the recommended value for φ. The Golden Ratio Number is related to the Fibonacci sequence and it is considered to model the growth and decay rates in natural processes. There are many studies on the Golden Ratio Number and its applications in science, technology and arts, for further reading see [7], [8] and [9].

One problem encountered with first order optimization methods is the zig-zagging movement that appears with some ill-posed optimization problems, where the gradients are pointing approximately in an orthogonal direction to the direction toward the optimization solution. Due to the zig-zagging behavior, the optimization methods can converge prematurely to wrong solutions. In order to cope with the zig-zagging problem, two complementary acceleration techniques are applied: (1) the *Inertia Technique*, and (2) the *Piercing Technique*.

4.1 The inertia technique

The inertia (or momentum) technique is inspired from the acceleration techniques first introduced by Nesterov in 1983 (see [2], [3]), and it is based on the idea of using more than one of the previous solutions in order to update the current solution. The variant of Inertia Technique employed by *HFGD* is based on statistics applied to a queue structure (FIFO list) maintaining the signs of the scalar products p_k with the purpose to detect when the algorithm presents a dominant zig-zagging behavior. The recommended queue size, in order to obtain good statistics while reflecting a more recent and local behavior, is 100. At each iteration $k+1$ the following statistical ratio is evaluated:

$$r_{k+1} = \frac{n^-_{k+1}}{n^-_{k+1} + n^+_{k+1}}, \tag{26}$$

with n^-_{k+1} being the number of negative signs in the queue and n^+_{k+1} being the number of non-negative signs in the queue (note that $n^-_{k+1} + n^+_{k+1}$ is the queue size, most of the time 100). If $r_{k+1} \leq 0.50$, then it is considered that the zig-zagging behavior is not dominant, and the Inertia Technique should be applied. When the Inertia Technique is applicable, the following weights are introduced:

$$w_1 = 0.5 - r_{k+1}, \tag{27}$$

$$w_2 = 1.0 - w_1 = 0.5 + r_{k+1}, \tag{28}$$

and equation (24) is modified to:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - w_2\gamma_{k+1}\mathbf{n}_k + w_1(\mathbf{x}_k - \mathbf{x}_{k-1}), \tag{29}$$

where two previous solutions are used, $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$.

4.2 The piercing technique

The Piercing Technique complements the Inertia Technique when the zig-zagging behavior is dominant, and it is applied every time when there are two consecutive changes in gradient's direction with angles larger than $\pi/2$. The main idea in the Piercing Technique is to probe some search directions approximately perpendicular to the search directions given by the gradients. It is applied immediately after the update of $\mathbf{x}_{k+1}$ (using (24) or (29), which one is applicable), by first computing the directional unit vector $\mathbf{n}$ according to:

$$\mathbf{n}_1 = \frac{\mathbf{x}_{k+1} - \mathbf{x}_{k-1}}{||\mathbf{x}_{k+1} - \mathbf{x}_{k-1}||_2}, \tag{30}$$

$$\mathbf{n}_2 = \frac{\mathbf{x}_k - \mathbf{x}_{k-2}}{||\mathbf{x}_k - \mathbf{x}_{k-2}||_2}, \tag{31}$$

$$\mathbf{n} = \frac{\mathbf{n}_1 + \mathbf{n}_2}{||\mathbf{n}_1 + \mathbf{n}_2||_2}, \tag{32}$$

and then computing the start position:

$$\mathbf{x}_s = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}), \tag{33}$$

and moving from the start position in the determined direction

$$\mathbf{x'}_{k+1} = \mathbf{x}_s + \gamma_{k+1}\mathbf{n}. \tag{34}$$

These ideas are graphically illustrated in Fig. 10.

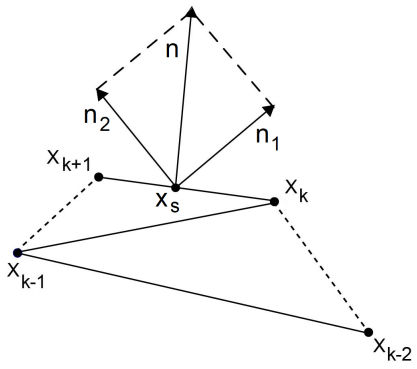

Fig. 10. The piercing direction

Further, the condition $f(\mathbf{x'}_{k+1}) < f(\mathbf{x}_{k+1})$ is tested, and if it is true then $\mathbf{x'}_{k+1}$ is taken as the new solution:

$$\mathbf{x}_{k+1} = \mathbf{x'}_{k+1}. \tag{35}$$

Then, a repeated search in the determined direction is probed with an increasing step size, as long as the objective function continues to decrease. This search is based on the following iterative algorithm:

Algorithm 2 *Piercing Technique* algorithm

1: $\mathbf{x}'_{k+1} = \mathbf{x}_s + \gamma_{k+1}\mathbf{n}$
2: $\gamma'_{k+1} = \varphi\gamma_{k+1}$
3: **while** $f(\mathbf{x}'_{k+1}) < f(\mathbf{x}_{k+1})$ **do**
4: $\quad \mathbf{x}_{k+1} = \mathbf{x}'_{k+1}$
5: $\quad \gamma_{k+1} = \gamma'_{k+1}$
6: $\quad \mathbf{x}'_{k+1} = \mathbf{x}_{k+1} + \gamma_{k+1}\mathbf{n}$
7: $\quad \gamma'_{k+1} = \varphi\gamma_{k+1}$

5 Testing and Results

The testing phase had the purpose to prove that the new proposed *HFGD* method is competitive when compared to existing gradient descent algorithms. For comparison, three other first order algorithms were chosen: *Backtracking Line Search* (*BLS*, [1]), a *Nesterov's Accelerated Gradient Descent* (*AGD*, [2], [3]) scheme (with backtracking) and the *Fast Iterative Shrinkage/Thresholding Algorithm* (*FISTA*, [4]) (with backtracking). A number of 100 runs were conducted for each considered method and each considered test function, the runs differing in the initial starting point $\mathbf{x}_0$, which was sampled pseudo-randomly (uniform distribution) in the search space, and the initial method's step size, γ_1 which was sampled pseudo-randomly (uniform distribution) in the $[0.2, 0.5]$ real interval. Therefore, different run results were obtained and their statistical analysis was needed.

In order to conduct the tests, an appropriate testing methodology was devised (see also [10], [11], [12]). When the quality of an optimization method is estimated, two (often conflicting) characteristics are of interest: a small number of function evaluations (NFE) and a high success rate (SR). For test functions with known solutions, the success can be simply defined as the achievement of an absolute or relative precision tolerance to the known solutions. By fixing the tolerance and choosing $iter_{max}$ high enough, so that this in never attained before the tolerance is attained, it is easy to measure the SR and average NFE to success ($\mu(NFE)$). There are other testing methodologies frequently applied in practice, like for example based on fixing NFE and reporting the best, the worst and the median results obtained after a number o runs, but in the author's opinion such methodologies are not recognizing the importance of success rate and are concealing it from reporting. A very efficient method (with a fast convergence), but with a low success rate, cannot be considered better than a less efficient method, but with a high success rate, because the former may need many repeated runs in order to obtain the correct result, while the later may get the correct result in less runs, which can entail a larger overall NFE (obtaining by summation) for the former compared to the later.

A testbed of 16 ($f_1, \ldots, f_{16}$) scalable, unconstrained, unimodal (excepting f_7, which has 2 modes) optimization functions was used for the tests run on the four compared optimization methods. The first 14 functions are known from the literature ([13], [14], [15], [16], [17], [18]), while the last 2 functions were specially designed for the present study. The analytical expressions of the used optimization functions are further given:

- *De Jong's First Function* (or sphere model) - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{aligned} f_1(\mathbf{x}) &= \sum_{j=1}^{n} x_j^2, \\ -100 \leq x_j &\leq 100, \;\; j = 1, \ldots, n \end{aligned} \tag{36}$$

- *Hyper-Ellipsoid Function* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{aligned} f_2(\mathbf{x}) &= \sum_{j=1}^{n} j x_j^2, \\ -100 \leq x_j &\leq 100, \;\; j = 1, \ldots, n \end{aligned} \tag{37}$$

- *Rotated Hyper Ellipsoid Function* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{aligned} f_3(\mathbf{x}) &= \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j^2 \right), \\ -100 \leq x_j &\leq 100, \;\; j = 1, \ldots, n \end{aligned} \tag{38}$$

- *Rosenbrock's Function* - minimum of 0 in $(1, 1, \ldots, 1)$:

$$\begin{aligned} f_4(\mathbf{x}) &= \sum_{j=1}^{n-1} \left[100(x_{j+1} - x_j^2)^2 + (1 - x_j)^2 \right], \\ -100 \leq x_j &\leq 100, \;\; j = 1, \ldots, n \end{aligned} \tag{39}$$

- *Gaussian Function* - minimum of 0 in $(1, 1, \ldots, 1)$:

$$\begin{aligned} f_5(\mathbf{x}) &= 1 - e^{-\frac{1}{n^2} \sum_{j=1}^{n} j(x_j - 1)^2}, \\ -5 \leq x_j &\leq 5, \;\; j = 1, \ldots, n \end{aligned} \tag{40}$$

- *Power of Differences Function* - minimum of 0 in $(1, 1, \ldots, 1)$:

$$\begin{aligned} f_6(\mathbf{x}) &= \sum_{j=1}^{n-1} (x_{j+1} - x_j)^2 + x_n^2 + x_1(x_1 - 2), \\ -100 \leq x_j &\leq 100, \;\; j = 1, \ldots, n \end{aligned} \tag{41}$$

- *Powell's Function* - minima of $-19n$ (-190 for $n = 10$, -380 for $n = 20$ and -570 for $n = 30$,) in $(-10, -10, \ldots, -10)$ and $(10, 10, \ldots, 10)$:

$$\begin{aligned} f_7(\mathbf{x}) &= \sum_{j=1}^{n} (|x_j| - 1)^2 - \sum_{j=1}^{n-1} x_j x_{j+1} - x_n x_1, \\ -10 \leq x_j &\leq 10, \;\; j = 1, \ldots, n \end{aligned} \tag{42}$$

- *Nesterov's Function* - $\rho = 100$, minimum of 0 in $(1, 1, \ldots, 1)$:

$$\begin{gathered} f_8(\mathbf{x}) = \frac{1}{4}(x_1 - 1)^2 + \rho \sum_{j=1}^{n-1} (x_{j+1} - 2x_j^2 + 1)^2, \\ -5 \leq x_j \leq 5, \;\; j = 1, \ldots, n \end{gathered} \tag{43}$$

- *Schwefel's Ridge* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{gathered} f_9(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2, \\ -100 \leq x_j \leq 100, \;\; j = 1, \ldots, n \end{gathered} \tag{44}$$

- *Neumaier Function #3* - minimum of $-n(n+4)(n-1)/6$ (-210 for $n = 10$, -1520 for $n = 20$, and -4930 for $n = 30$) in $x_j = j(n - j + 1)$, $j = 1, \ldots, n$:

$$\begin{gathered} f_{10}(\mathbf{x}) = \sum_{j=1}^{n} (x_j - 1)^2 - \sum_{j=2}^{n} x_j x_{j-1}, \\ -n^2 \leq x_j \leq n^2, \;\; j = 1, \ldots, n \end{gathered} \tag{45}$$

- *Zakharov's Function* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{gathered} f_{11}(\mathbf{x}) = \sum_{j=1}^{n} x_j^2 + \left(\sum_{j=1}^{n} 0.5 j x_j \right)^2 + \\ + \left(\sum_{j=1}^{n} 0.5 j x_j \right)^4, -5 \leq x_j \leq 10, \;\; j = 1, \ldots, n \end{gathered} \tag{46}$$

- *High Conditioned Elliptic Function* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{gathered} f_{12}(\mathbf{x}) = \sum_{j=1}^{n} \left(10^6\right)^{\frac{j-1}{n-1}} x_j^2, \\ -100 \leq x_j \leq 100, \;\; j = 1, \ldots, n \end{gathered} \tag{47}$$

- *Bent Cigar Function* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{gathered} f_{13}(\mathbf{x}) = x_1^2 + 10^6 \sum_{j=2}^{n} x_j^2, \\ -100 \leq x_j \leq 100, \;\; j = 1, \ldots, n \end{gathered} \tag{48}$$

- *Discus Function* - minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{gathered} f_{14}(\mathbf{x}) = 10^6 x_1^2 + \sum_{j=2}^{n} x_j^2, \\ -100 \leq x_j \leq 100, \;\; j = 1, \ldots, n \end{gathered} \tag{49}$$

- *Unimodal Non-Quasi-Convex Example Function* - the same as the function example given in Section 2), with the minimum of 0 in $(0, 0, \ldots, 0)$:

$$\begin{gathered} f_{15}(\mathbf{x}) = \sum_{j=1}^{n} x_j^4 + 16 \sum_{j=1}^{n} x_j^2 x_{j+1}^2, \\ -2 \leq x_j \leq 2, \;\; j = 1, \ldots, n, \;\; x_{n+1} = x_1 \end{gathered} \tag{50}$$

- *Almost Plateau Quasi-Convex Function* - it is a modification to the flat region function $f_{16}(\mathbf{x})$ from Section 4, with a small slope introduced toward the minimum mode, in order to help the escape of the optimization algorithm from the flat region. The minimum is -1 in $(1, 1, \ldots, 1)$:

$$\begin{gathered} f_{16}(\mathbf{x}) = -(n+1) e^{-10\sqrt{n}\left[\sum_{j=1}^{n} (x_j - 1)^2\right]^{1/2}} + \\ + \max\{\sum_{j=1}^{n} x_j^2, n + \frac{0.01}{n} - 0.2 + \frac{0.001}{n^3} \sum_{j=1}^{n} (x_j - 1)^2\} \\ -2 \leq x_j \leq 2, \;\; j = 1, \ldots, n \end{gathered} \tag{51}$$

The 16 scalable test functions were tested for 3 different increasing search space dimensions ($n = 10$, $n = 20$ and $n = 30$), in order to study the impact of the increase in the search space dimension on the performance of the tested optimization methods.

Table 1 presents the comparative testing results obtained for the search space dimension $n = 10$. It can be observed that the novel *HFGD* method was the only one capable to solve all the test problems. Also, from the efficiency perspective, *HFGD* clearly surpassed all the other methods with the exception of f_1, but in that case the difference was very small, considering that all the tested methods were capable to solve f_1 very efficiently. From the success rate perspective, *HFGD* surpassed the other methods, with the exception of f_4 and f_7, for which *AGD* gave better results. Notice that all methods had difficulties in solving f_8 (which is known as a very difficult problem) and only *HFGD* was capable to solve the specially designed non-quasi-convex function f_{15}.

Table 2 presents the comparative testing results obtained for the search space dimension $n = 20$. It can be observed that the novel *HFGD* method was again the only one capable to solve all the test problems. The obtained efficiency and success rate comparative characteristics were similar to the ones in Table 1. The function f_8 was still difficult to solve for all methods, while *BLS*, *AGD* and *FISTA* were still not capable to solve f_{15}, and additionally f_{16} (the specially designed quasi-convex function with almost a flat region).

Table 3 presents the comparative testing results obtained for the search space dimension $n = 30$. It can be observed that the novel *HFGD* method was again the only one capable to solve all the test problems. The obtained efficiency and success rate comparative characteristics were similar to the ones in Tables 1 and 2. The function f_8 was still difficult to solve for all methods, while *BLS*, *AGD* and *FISTA* were still not capable to solve f_{15} and f_{16}.

Table 1. HFGD versus BLS , AGD and FISTA, $n = 10$**,** $runs = 100$**,** $tolerance = 0.1\%$

Fn	$SR\%$ FGD	$\mu(NFE)$ FGD	$SR\%$ BLS	$\mu(NFE)$ BLS	$SR\%$ AGD	$\mu(NFE)$ AGD	$SR\%$ FISTA	$\mu(NFE)$ FISTA
f_1	100%	39	100%	**31**	100%	36	100%	41
f_2	100%	**49**	100%	300	100%	236	100%	429
f_3	100%	**122**	100%	676	100%	563	100%	615
f_4	61%	**7666**	65%	61199	**90%**	63639	89%	78613
f_5	100%	**16**	100%	75	100%	64	100%	69
f_6	**100%**	**805**	0%	N/A	6%	5692	4%	9874
f_7	53%	**24**	5%	35	**88%**	599	74%	199
f_8	12%	**1707**	6%	30386	12%	14551	9%	3877
f_9	**100%**	**291**	100%	1585	0%	N/A	97%	7158
f_{10}	100%	**105**	100%	149	100%	147	100%	148
f_{11}	100%	**275**	86%	3106	94%	6695	80%	4486
f_{12}	100%	**49**	100%	308	100%	543	100%	557
f_{13}	100%	**294**	1%	14758	100%	103264	100%	121338
f_{14}	100%	**386**	0%	N/A	100%	162940	69%	176857
f_{15}	**100%**	**40**	0%	N/A	0%	N/A	0%	N/A
f_{16}	**100%**	**375**	99%	569	13%	1869	12%	2060

Table 2. HFGD versus *BLS*, *AGD* and *FISTA*, $n = 20$, $runs = 100$, $tolerance = 0.1\%$

Fn	$SR\%$ FGD	$\mu(NFE)$ FGD	$SR\%$ BLS	$\mu(NFE)$ BLS	$SR\%$ AGD	$\mu(NFE)$ AGD	$SR\%$ FISTA	$\mu(NFE)$ FISTA
f_1	100%	41	100%	**31**	100%	48	100%	48
f_2	100%	**59**	100%	564	100%	915	100%	905
f_3	100%	**137**	100%	765	100%	718	100%	715
f_4	58%	**9738**	62%	100572	**88%**	65531	86%	80872
f_5	100%	**20**	100%	100	100%	90	100%	91
f_6	**100%**	**2207**	0%	N/A	1%	16405	0%	29824
f_7	**52%**	**31**	1%	36	32%	1384	24%	1657
f_8	10%	**1303**	10%	21249	**14%**	18258	12%	17250
f_9	**100%**	**459**	92%	5912	91%	30062	96%	29903
f_{10}	100%	**224**	100%	479	100%	314	100%	306
f_{11}	**100%**	**311**	87%	17912	66%	20558	65%	17110
f_{12}	100%	**49**	100%	309	100%	644	100%	657
f_{13}	100%	**259**	0%	N/A	100%	141876	100%	126689
f_{14}	**100%**	**412**	0%	N/A	44%	148371	47%	152574
f_{15}	**100%**	**39**	0%	N/A	0%	N/A	0%	N/A
f_{16}	**100%**	**777**	0%	N/A	0%	N/A	0%	N/A

Table 3. HFGD versus BLS, AGD and FISTA, $n = 30$**,** $runs = 100$**,** $tolerance = 0.1\%$

Fn	$SR\%$ FGD	$\mu(NFE)$ FGD	$SR\%$ BLS	$\mu(NFE)$ BLS	$SR\%$ AGD	$\mu(NFE)$ AGD	$SR\%$ FISTA	$\mu(NFE)$ FISTA
f_1	100%	42	100%	**31**	100%	42	100%	42
f_2	100%	**65**	100%	297	100%	420	100%	419
f_3	100%	**146**	100%	672	100%	625	100%	624
f_4	56%	**9222**	69%	59542	**85%**	58205	81%	82170
f_5	100%	**23**	100%	71	100%	67	100%	66
f_6	**100%**	**818**	0%	N/A	2%	14709	6%	11778
f_7	67%	**23**	3%	46	70%	283	**72%**	111
f_8	11%	**571**	11%	2964	11%	11763	9%	16351
f_9	**100%**	**292**	97%	1609	97%	7287	98%	7193
f_{10}	100%	**104**	100%	151	100%	141	100%	146
f_{11}	**100%**	**277**	83%	3567	74%	4603	83%	4093
f_{12}	100%	**49**	100%	300	100%	566	100%	566
f_{13}	100%	**293**	1%	184154	100%	132866	100%	137529
f_{14}	**100%**	**385**	0%	N/A	75%	139047	73%	151233
f_{15}	**100%**	**35**	0%	N/A	0%	N/A	0%	N/A
f_{16}	**100%**	**381**	0%	N/A	0%	N/A	0%	N/A

6 Conclusions

The presented *HFGD* algorithm introduced the novel idea of using the gradient direction's change in order to vary the step size. The testing conducted on a testbed of 16 test functions showed an overall much better efficiency and an overall better effectiveness of the novel *HFGD* method compared to other three known first-order gradient descent methods, even for convex unimodal functions. *HFGD* is heuristic in nature and no rigorous demonstration of its convergence to the correct solution was provided. Experimentally, it was found that *HFGD* gives correct results for some ill-posed optimization problems, only if the Piercing Technique is employed. Therefore the Piercing Technique is mandatory if black-box optimization models are approached. Further theoretical research is needed in order to establish the conditions in which *HFGD* converges to the correct solution. Further research will be concerned with designing a *HFGD* variant capable to approach constrained optimization problems. Also, large dimensional optimization problems, which are very important in fields like Machine Learning, will be approached. Another research direction will be the hybridization of *HFGD* (as a local search technique) with known global optimization metaheuristic techniques, in order to design improved global optimization methods (so called memetic optimization methods).

Competing Interests

Author has declared that no competing interests exist.

References

[1] Armijo L. Minimization of functions having lipschitz continuous first partial derivatives. Pacific J. Math. 1966;16(1):13.

[2] Nesterov Y. A method for solving a convex programming problem with convergence rate $O(1/k^2)$. Soviet Mathematics Doklady. 1983;27(2):372-376.

[3] Nesterov Y.Introductory lectures on convex optimization: a basic course. Kluwer Academic Publishers, Boston/Dordrecht/London;2004.

[4] Teboulle M, Beck A. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sciences. 2009;2(1):183-202.

[5] Kanemitsu H, Miyakoshi M, Shimbo M. Properties of unimodal and multimodal functions defined by the use of local minimal value set. Scripta Technica. Electron Comm Jpn Pt 3. 1998;81(1):4251.

[6] Greenberg HJ, Pierskalla WP. A review of quasi-convex functions. Operations Research. 1971;19(7):1553-1570.

[7] Elam K. Geometry of design, studies in proportion and composition. Princeton Architectural Press, New York;1981.

[8] Akhtaruzzaman Md, Shafie AA. Geometrical substantiation of Phi, the golden ratio and the baroque of nature, architecture, design and engineering. International Journal of Arts. 2011;1(1):1-22.

[9] Dragoi AL. A simple recursive geometrical construct method and some properties and occurrences of the golden ratio (Phi) in mathematics, physics, chemistry and biology (including human medicine), also related to the transcendentals PI and to euler's constant (e), Accessed Date:30/01/18. [Online].
Available: https://www.researchgate.net/publication/312983942

[10] Anescu G, Prisecaru I. NSC-PSO, A novel PSO variant without speeds and coefficients, in: proceedings of he 17th international symposium on symbolic and numeric algorithms for scientific computing. SYNASC 2015, Timisoara, Romania: September 2015;21-24.pp.460-467.

[11] Anescu G. An imperialistic strategy approach to continuous global optimization problem, in: proceedings of the 16th international symposium on symbolic and numeric algorithms for scientific computing, SYNASC 2014, Timisoara, Romania: September 2014;22-25.pp.549-556.

[12] Anescu G. Gradual and cumulative improvements to the classical differential evolution scheme through experiments. Annals of west university of timisoara - Mathematics and Computer Science. 2016;54(2):13-35.

[13] Moré JJ, Garbow BS, Hillstrom KE. Testing unconstrained optimization software. ACM Transactions on Mathematical Software. 1981;7(1):1741.

[14] Momin J, Yang XS. A literature survey of benchmark functions for global optimization problems. Int. Journal of Mathematical Modelling and Numerical Optimisation. 2013;4:150-194.

[15] Molga M, Smutnicki C. Test functions for optimization needs. 2005; Accessed Date:03/11/17. [Online].
Available: http://www.robertmarks.org/Classes/ENGR5358/Papers/functions.pdf

[16] Gürbüzbalaban M, Overton ML. On Nesterov's nonsmooth Chebyshev-rosenbrock functions. Nonlinear Analysis: Theory, Methods & Applications. 2012;75(3):1282-1289.

[17] Liang JJ, Qu BY, Suganthan PN, Hernandez-Diaz AG. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, technical report 201212, computational intelligence laboratory. Zhengzhou University, Zhengzhou China And Technical Report, Nanyang Technological University, Singapore;2013.

[18] Price K, Storn RM, Lampinen JA. Differential evolution: a practical approach to global optimization. Springer-Verlag Berlin Heidelberg;2005.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://www.sciencedomain.org/review-history/23378