Proceedings of A. Razmadze
Mathematical Institute
Vol. **161** (2013), 83–95

ON THE MODIFICATION OF HEAVY BALL METHOD

K. GELASHVILI, L. ALKHAZISHVILI, I. KHUTSISHVILI, N. ANANIAISHVILI

Abstract. This paper discusses the new continuous model of the heavy ball method. It examines in detail the modified heavy ball algorithm from a theoretical perspective in terms of geometric interpretation and programming realization. The new model assumes the possibility to slow down, stop or accelerate. The programming realization of the new algorithm is much more flexible. It provides for a richer set of parameters to select from and during the testing it consistently demonstrates much better results compared to the heavy ball method.

რეზიუმე. წარმოდგენილ ნაშრომში განიხილება მძიმე ბირთვის მეთოდის ახალი უწყვეტი მოდელი. საკმაოდ დეტალურად, თეორიული ასპექტების, გეომეტრიული გააზრების და პროგრამული რეალიზაციის თვალსაზრისით ხდება მძიმე ბირთვის მოდიფიცირებული ალგორითმის შესწავლა. ახალი მეთოდი უშვებს დამუხრუჭების, გაჩერების, აჩქარების შესაძლებლობას. ახალი ალგორითმის პროგრამული რეალიზაცია გაცილებით მოქნილია, აქვს პარამეტრების შერჩევის უფრო მდიდარი საშუალებები და ტესტებზე სტაბილურად აჩვენებს გაცილებით უკეთეს შედეგებს მძიმე ბირთვის მეთოდთან შედარებით.

Introduction

The "heavy ball" is a well-known and effective method to solve the unconstrained minimization problem of the smooth function, $f(x) \to \min$, $x \in R_n$. This method is multifunctional in its essence and can be used both for quickly attaining the "first" local minimal and for identifying global (or good local) minimal.

The heavy ball is an interesting topic from the perspective of mathematical modeling, since the idea of the dynamic process converging towards equilibrium can be realized through significantly different mathematical models (see [1]–[5]). A number of scientific papers (e.g. [6]–[10]), timely from the practical point of view, are currently devoted to the general cases of unconstrained optimization (e.g. smooth, convex, quasiconvex functions). In these papers first or second order ordinary differential systems are used

2010 *Mathematics Subject Classification.* 49M37, 65K05.

Key words and phrases. Heavy ball method, Fletcher-Reeves method, Eason's function, Pawell's function, Rosenbrock's function.

as mathematical models. Through experimenting we can become assured that the heavy ball method can be successfully used for the minimization of non-smooth functions.

The first paragraph describes heavy ball method and its new modification: continuous model of heavy ball method in terms of delayed neutral system; recurrent equations for the algorithm of standard heavy ball method; description of modified Heavy Ball method and theorems of convergence. It is well known that proving of convergence of the standard heavy ball method is much more difficult, than for example proving of convergence of the standard gradient method (with fixed step). In contrast, proving of convergence of our modified heavy ball method has the same level of difficulty as proving of convergence of the standard gradient method (see theorems).

The second paragraph presents a geometrical interpretation of the new approach.

The third paragraph provides the results of testing of the modified method's programming realization on several well-known problems and shows that this method has a number of advantages compared to the standard one.

The aim of the presented paper is to attract the attention of the specialists in this field to the new algorithm. Therefore, we prove the correctness of the new method, evaluation the speed of convergence relying solely on test results.

1. Modification of the Heavy Ball Method

The standard heavy ball method algorithm is defined by a recurrent equation:

$$\begin{cases} x_1 = x_0 - \beta_0 f'(x_0), \\ x_{i+1} = x_i + \alpha_i(x_i - x_{i-1}) - \beta_i f'(x_i), \quad i > 0, \end{cases} \tag{1}$$

where $f : \mathbb{R}_n \to \mathbb{R}$ is the objective function, x_0 is the initial approximation of the solution, $0 \le \alpha_i \le 1$,$i > 0$ and $\beta_i > 0$, $i \ge 0$. In extreme cases, when $\alpha_i = 0$ and $\alpha_i = 1$, we get a standard fixed step gradient method and endless frictionless movement of the heavy ball. It is widely accepted that in case of well-chosen $0 \le \alpha_i \le 1$ and $\beta_i > 0$ coefficients, the heavy ball is one of the best among the numeric optimization methods. This is achieved thanks to the fact that in the proximity of minimal the speed of convergence does not decrease. However, choosing the coefficients is quite complicated and it is essential to take into account the experience accumulated while experimenting with specific problems.

According to the widely accepted approach (see [2], [5], [6]), continuous model of scheme (1) is a second order differential system, but this model is less informative in some cases (see below) and we choose to use a different model.

If we assume that step in (1) is quite small and equal to delay $\tau > 0$, then we can consider the following neutral delayed system as the continuous model for scheme (1):

$$\begin{cases} \dot{x}(t) = 0, \quad t \le 0; \quad x(0) = x_0, \\ \dot{x}(t) = \alpha\, \dot{x}(t-\tau) - \beta\, f'(x(t)), \quad t > 0. \end{cases} \tag{2}$$

Let us discuss a modification of the heavy ball method, which is much faster and its coefficients can be chosen from a much vaster range.

Let us assume, $f : \mathbb{R}_n \to \mathbb{R}$ is the objective function, x_0 is the initial approximation of the solution, α_i, β_i are constants, equal to some $\alpha \ge 0$, $\beta > 0$ consequently (which is quite small) are constants and let's describe an algorithm:

Let us define $\{y_i\}_{i=0}^{\infty}$ sequence in the following way:

- Let us take $y_0 = x_0$. According to formula (1) we find the next approximation, until the values of the objective function are decreasing; as soon as for some x_i we will get $f(x_i) \le f(x_{i+1})$, we will take $y_1 = x_i$, $x_0 = y_1$. Let us note that $f(y_1) \le f(y_0)$.
- Let us assume that y_n is already constructed so that $f(y_n) < f(y_{n-1})$ and $x_0 = y_n$. Again using formula (1) let us find the next approximation, until the values of the objective function are decreasing; as soon as for some x_i we will get $f(x_i) < f(x_{i+1})$, we will take $y_{n+1} = x_i$, $x_0 = y_{n+1}$. We again have that $f(y_{n+1}) \le f(y_n)$.
- We continue defining the terms of $\{y_i\}$ sequence until the condition to stop is not satisfied
  (usually this has to do with small value of the objective function gradient in the point y_k).

Let us prove the convergence of the method under standard conditions (see example, [3]). We should note that $\{y_i\}_{i=0}^{\infty}$ sequence defined by us has the following properties: for every $k \in \{0, 1, 2, \dots\}$ index, y_k approximation already found defines transitional value of $y_k - \beta f'(y_k)$, using which we find y_{k+1} (how do we do that has no pertinence to proving the convergence), such that the inequality $f(y_{k+1}) \le f(y_k - \beta f(y_k))$ stands.

Theorem 1. *Let us assume that $f(\cdot)$ is derivable in $\mathbb{R}^n$ and the gradient of $f(\cdot)$ satisfies the Lipschitz's condition:*

$$\|f'(x) - f'(y)\| \le L\|x - y\|; \tag{3}$$

$f(\cdot)$ is bounded below

$$f(x) \ge f^* > -\infty; \tag{4}$$

β satisfies the condition:

$$0 < \beta < 2/L. \tag{5}$$

In $\{y_k\}_{k=0}^{\infty}$ *sequence* y_0 *is an arbitrarily taken vector and for every* $k \in \{0,1,2,\dots\}$ *index* y_k *and* y_{k+1} *approximations relate to each other only through the following inequality:* $f(y_{k+1}) \le f(y_k - \beta f'(y_k))$.

Then, for $\{y_i\}_{i=0}^{\infty}$ *sequence:*

- $f(\cdot)$ *function is monotonously decreasing on* $\{y_k\}_{k=0}^{\infty}$ *sequence: condition* $f(y_{k+1}) \le f(y_k)$ *is fulfilled for each* k.
- *Gradient converges towards* 0*:* $\lim\limits_{k\to\infty} f'(y_k) = 0$.

Proof. The proof repeats the main parts of the proof of the convergence of the simplest gradient method. Let us use the formula:

$$f(x+y) = f(x) + \int_0^1 f'(x+\tau y)\cdot y^T d\tau$$

which gives us:

$$\begin{aligned}
&f\big(y_k - \beta\, f'(y_k)\big) = \\
&= f(y_k) + \int_0^1 f'\big(y_k + \tau(-\beta\, f'(y_k))\big)\cdot\big(-\beta\, f'(y_k)\big)^T d\tau = \\
&\quad \Big(\text{add and subtract} \quad f'(y_k)\cdot\big(-\beta\, f'(y_k)\big)^T\Big) \\
&= f(y_k) + f'(y_k)\cdot\big(-\beta\, f'(y_k)\big)^T + \\
&\quad + \int_0^1 \Big[f'\big(y_k + \tau(-\beta\, f'(y_k))\big) - f'(y_k)\Big]\cdot\big(-\beta\, f'(y_k)\big)^T d\tau \le \\
&\le f(y_k) - \beta\big\|f'(y_k)\big\|^2 + \\
&\quad + \Big|\int_0^1 \Big[f'\big(y_k + \tau(-\beta\, f'(y_k))\big) - f'(y_k)\Big]\cdot\big(-\beta\, f'(y_k)\big)^T d\tau\Big| \le \\
&\le f(y_k) - \beta\big\|f'(y_k)\big\|^2 + \\
&\quad + \int_0^1 \Big\|f'\big(y_k + \tau(-\beta\, f'(y_k))\big) - f'(y_k)\Big\|\cdot\big\|-\beta\, f'(y_k)\big\| d\tau \le \\
&\le f(y_k) - \beta\big\|f'(y_k)\big\|^2 + \int_0^1 L\cdot\big\|\tau(-\beta\, f'(y_k))\big\|\cdot\big\|-\beta\, f'(y_k)\big\| d\tau = \\
&= f(y_k) - \beta\big\|f'(y_k)\big\|^2 + \frac{L\beta^2}{2}\big\|f'(y_k)\big\|^2 = f(y_k) - \beta\left(1 - \frac{L\beta}{2}\right)\big\|f'(y_k)\big\|^2.
\end{aligned}$$

We get

$$f\big(y_k - \beta f'(y_k)\big) \le f(y_k) - \beta\left(1 - \frac{L\beta}{2}\right)\big\|f'(y_k)\big\|^2.$$

Since $f(y_{k+1}) \le f(y_k - \beta f'(y_k))$, therefore

$$f(y_{k+1}) \le f(y_k) - \beta\left(1 - \frac{L\beta}{2}\right)\big\|f'(y_k)\big\|^2,$$

so

$$f(y_{k+1}) \le f(y_k) - \alpha\big\|f'(y_k)\big\|^2, \tag{6}$$

where $\alpha = \beta\big(1 - \frac{L\beta}{2}\big)$. This formula proves the first part of the theorem. If we use the same formula several times in the right part of (6) we will get:

$$f(y_{k+1}) \le f(y_0) - \alpha\Big(\big\|f'(y_0)\big\|^2 + \cdots + \big\|f'(y_k)\big\|^2\Big) =$$

$$= f(y_0) - \alpha\sum_{i=0}^{k}\big\|f'(y_i)\big\|^2.$$

with (5) in mind, $\alpha > 0$, therefore for every k it is true that

$$\sum_{i=0}^{k}\big\|f'(y_i)\big\|^2 \le \alpha^{-1}\big(f(y_0) - f(y_{k+1})\big) \le \alpha^{-1}\big(f(y_0) - f^*\big).$$

This inequality is also true for every i and the right side is not dependent on i, therefore it is also true that

$$\sum_{i=0}^{\infty}\big\|f'(y_i)\big\|^2 \le \alpha^{-1}\big(f(y_0) - f^*\big),$$

which proves that $\|f'(y_i)\| \to 0$ when $t \to \infty$. □

If for every $k \in \{0, 1, 2, \dots\}$ index, we denote $x_k = y_k - \beta f'(y_k)$, then several first terms of $\big\{y_k\big\}_{k=0}^{\infty}$ sequence can be located as it is shown in Figure 1.

Theorem 2. *If besides the conditions of the Theorem* 1 *the objective function is also strongly convex, then the sequence* $\{y_i\}_{i=1}^{\infty}$ *constructed in Theorem* 1 *converges to the unique global minimal of objective function* $f(\cdot)$.

Proof. Objective function $f(\cdot)$ as strongly convex function has unique global minimal $\widehat{y}$. It is also well-known that

$$f'(x) = 0$$

is the nessessary and sufficient condition for minimum of convex function. Again from the strong convexity of $f(\cdot)$ follows that the set

$$S = \big\{x \in \mathbb{R}_n \,|\, f(x) \le f(y_0)\big\}$$

is bounded and convex. Consequently, $\{y_i\}_{i=1}^{\infty}$ has limiting points. Let $\widetilde{y}$ be arbitrarily given limiting point of $\{y_i\}_{i=1}^{\infty}$ and $y_{i_j} \to \widetilde{y}$ as $j \to \infty$. $f(\cdot)$ is continuously differentiable and $\|\cdot\|$ is also continuous function, therefore

$$0 = \lim_{j\to\infty} \left\|f'(y_{i_j})\right\| = \left\|f'(\widetilde{y}\right\|.$$

From $0 = f'(\widetilde{y}) = f'(\widehat{y})$ follows $\widetilde{y} = \widehat{y}$. By virtue of arbitrarity of $\widetilde{y}$, every convergent subsequence of $\{y_i\}_{i=1}^{\infty}$ converges to $\widehat{y}$. But this means that $\{y_i\}_{i=1}^{\infty}$ itself converges to $\widehat{y}$. □

If besides the conditions of the proven theorem, the objective function is also strongly convex, then convergence of $\{y_i\}_{i=1}^{\infty}$ sequence is proved exactly like, for example, as in [3], so we will not repeat it.

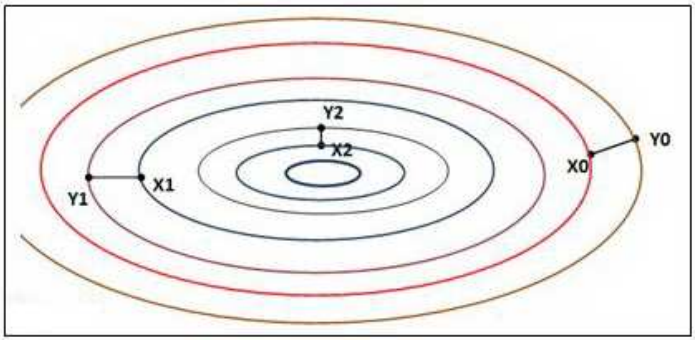

FIGURE 1

2. GEOMETRIC INTERPRETATION OF THE METHOD FOR A QUADRATIC OBJECTIVE FUNCTION

Trajectory of the modified heavy ball is shown in four different cells in figures coupled with trajectories of four prevalent minimization methods. $x^2 + 10y^2$ is taken as the objective function. Initial conditions are the same everywhere. The coefficients of standard and modified heavy balls are equal. The trajectories of the steepest descent and modified heavy ball are simultaneously shown in the first cell (see about the terminology pertaining to the other methods in the last point). In the second cell you can see where the modified heavy ball stops, unlike the standard heavy ball, and its further movement. In the third cell we compare our method to the standard gradient method with fixed step. In the fourth cell all the four trajectories are shown at the same time:

The trajectories resulting from minimization of Rosenbrock's function using the following methods are shown in the cells of the next table (Figure 3):

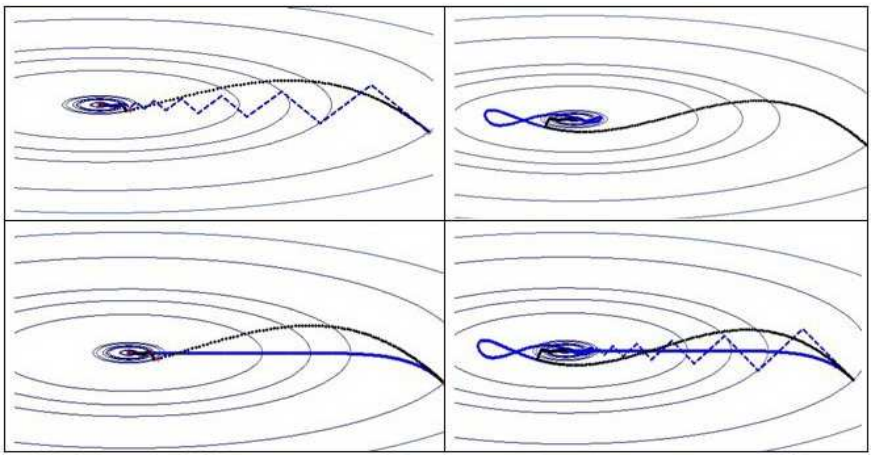

FIGURE 2

- the steepest descent;
- heavy ball;
- modified heavy ball;
- standard gradient method with fixed step;

(please find the detailed information about Rosenborck's function below).

As we can see, the same fact that we found during the comparison of the heavy and modified heavy balls in the vicinity of the solution for the quadratic function, shows up from the beginning of the trajectory in case of the Rosenbrock's function.

It is also interesting that the trajectory of the modified heavy ball is quite close to the one of standard gradient method, but the modified method is significantly faster.

The radically different trajectory of the steepest descent method is explained by the circumstance that this method, just as Fletcher-Reeves method, is significantly dependent on the used one-dimensional minimization problem algorithm. When one-dimensional algorithm quickly (and roughly) tries to identify the endpoints of the search interval, for a non-convex function there is a possibility that one-dimensional minimal is not a global or good local minimal for the given direction. Getting a different trajectory is possible with a different realization.

3. The Results of Testing the Modified Heavy Ball Algorithm

We rarely encounter scientific studies that compare and analyze programming codes of different algorithms. One reason could be that the conclusions of such works are sometimes subjective and contradictory. We tried to test

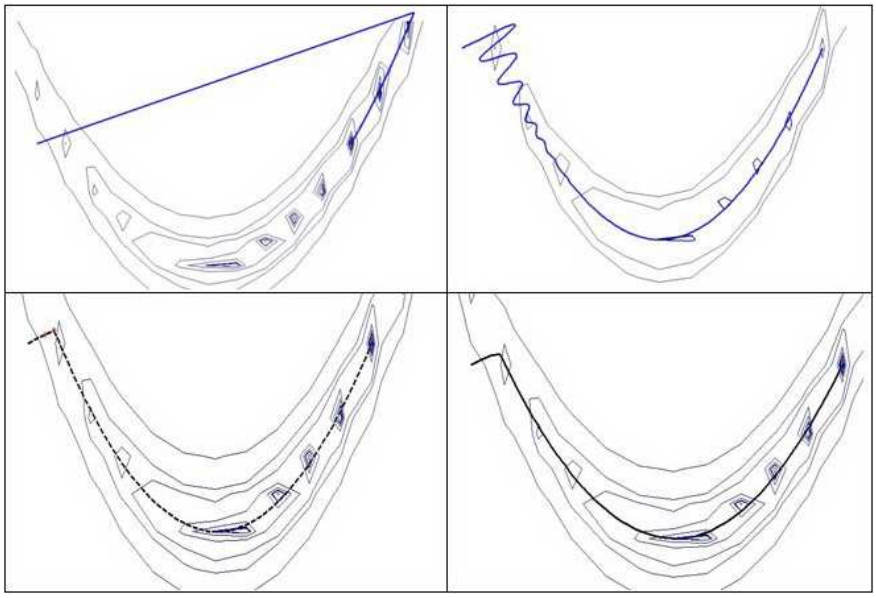

FIGURE 3

the algorithm using the same technologies used in [11], which is a kind of de facto standard. From a methodological perspective in [12], [13] we selected several very well-known minimization problems, for which the tests of other algorithms (except the modified heavy heavy ball) were known. However, we deemed necessary to try our implementation of several algorithms and compare to the modified heavy ball method.

Due to the small number of variables in the test tasks, we have also considered the such type of quadratic minimization problem, where the objective function is a penalty function for some system of linear equations.

The code for the modified heavy ball algorithm is extremely easy to write. Maybe it is exactly because of simplicity that the algorithm works extremely reliably. We will be assured in the algorithm's reliability after discussing the results of testing.

The description and implementation of the modified heavy ball algorithm are for simplicity placed together in "ModifiedHeavyBall" file. Let us note that thanks to using STL library capabilities there is no necessity to use indexed variables, therefore saving the needed amounts of points in the trajectory is easily regulated. Since due to limited space we provide only the modified heavy ball code, we do not place it separately in "namespace". Unlike [11], "ModifiedHeavyBall.h" file receives the necessary information

about the objective function from "data.h" file, which can be changed if needed, boosting the codes flexibility.

Let us take into account that in case of the standard heavy ball

double alpha=1.05;

line would definitely cause the method to overflow. This is one of the strong resources to accelerate the convergence in our method. Another different and interesting resource is the term "inertia", which always becomes null when a threat that the objective function will grow along the trajectory appears. This means that we cause the ball to brake in this moment and then it has to accelerate. Maintaining inertia at a certain level and turning it to a needed direction thus creating an effect of turning the ball in motion makes for an interesting perspective. For the sake of simplicity we do not dwell on this effect in this paper.

There is no need for more comments due to the simplicity of the code:

```
//file "ModifiedeavyBall.h"
# include"data.h" // addressing data (dimension, function, derivative)
# include<iostream>
# include<vector>
const double EPS = 1E-6;
double beta = 1E-4;
double alpha = 1.05;
vector<double> inertia(DIMENSIONALITY);
vector<vector<double>> trajectory;
template<typename T>
void getNextIteration( vector<T>& x0)
{
   vector<T> x1(DIMENSIONALITY,0.0);
   while(true)
   {
      objFunctionPrime(x0);
      for(int i=0; i < DIMENSIONALITY; ++i)
         x1[i] = x0[i] + alpha *inertia[i] - beta*yPrime[i];
      if(objFunction(x0)<= objFunction(x1))
         return;
      for(int i=0; i < DIMENSIONALITY; ++i)
         inertia[i] = x1[i]-x0[i];
      x0 = x1;
      if(dist(x0, trajectory[trajectory.size()-1])>0.2)
         trajectory.push_back(x0);
   }
```

```
}
template<typename T>
void getMinimal( vector<T>& x0)
{
   while(true)
   {
      getNextIteration(x0);
      solutionIterations.push_back(x0);
      for(int i=0; i < DIMENSIONALITY; ++i)
         inertia[i] = 0.0;
      if(norm(yPrime)<EPS )
         break;
   }
}
```

It is implied that "ModifiedHeavyBall.h"file will get the following information from "data.h" file (in case of Pawell's function):

```
const int DIMENSIONALITY = 4; //problem dimension
double array[]={3.0, -1.0, 0.0, 1.0}; //{0.0, 0.0};
vector<double> x0(array,array+4); //starting point
double y(0.0); //point for saving function values
//vector for saving function derivative
vector<double>yPrime(DIMENSIONALITY);
```

```
template<typename T>
T objFunction( vector<T>&x)
{
   y =(x[0]+10*x[1])*(x[0]+10*x[1]) + 5*(x[2]- x[3])*(x[2]- x[3])+
      pow((x[1]-2*x[2]), 4) + 10*pow((x[0] - x[3]), 4);
   return y;
}
template<typename T>
void objFunctionPrime(const vector<T>& x)
{
   yPrime[0]=2*(x[0] + 10*x[1]) + 40*pow((x[0] - x[3]),3);
   yPrime[1]=20*(x[0] + 10*x[1]) + 4*pow((x[1] - 2*x[2]), 3);
   yPrime[2]=10*(x[2]-x[3]) - 8*pow((x[1]-2*x[2]),3);
   yPrime[3]=-10*(x[2]-x[3]) - 40*pow((x[0]-x[3]), 3);
}
```

Since our main purpose is to attract the attention to this approach and not provide a full-scale research in one paper, we placed emphasis on the level of algorithm performance. The algorithm stops when the derivative satisfies given precision. Testing took the minimal value for which the

Fletcher-Reeves method worked properly as selection criteria for the precision parameter. Testing took place on standard personal computers with the following specifications: Intel(R) Pentium(R) Dual CPU E2200 2.20 GHz, 2.00 GB of RAM. Methods used for testing (indicating some parameters) are listed in the left column and the names of the functions are given in the top row.

The results of testing are given in the next table. In every cell there is the result of testing the method in the corresponding row on the corresponding function. Two numbers are given in every cell, (e.g. 0.001 /1E-12), from which the first number is time in seconds (average of several trials) and the second number is precision. First of all, we consider the precision which is common for all the methods (mostly it is 1E-6).

Both for the heavy and modified heavy balls the length of the step is taken as

double beta = 1E-4;

We are not trying to accelerate the convergence of the method using this parameter, which gives an advantage to the other three methods. However, in case of the steepest descent and the Fletcher-Reeves methods we use fast one-dimensional minimization method.

Let us also note that terminology is mixed in scientific literature. The method we call standard gradient method with fixed step is sometimes called the steepest descent method. We use the term "Steepest descent" according to [3].

As it appears from these tests (with the small number of variables), the effectiveness of modified Heavy ball method only falls behind the Fletcher-Reeves algorithm. But if the number of variables increases significantly, the situation changes even in case of quadratic objective function. More precisely, if one can compose the penalty function $f(x) = \sum_{i=1}^{n} g(A[i]x - a[i])$ for the system of linear equations $A(x) = a$ (with square random matrix) where $A[i]$ is a i-th row of the matrix, $a[i]$ is the corresponding constant term, and $g(z) = \max(0, z)^2$, then approximately beginning from $n = 20$, modified Heavy ball method works much more faster than Fletcher- Reeves method. It is explained by the fact that the latter is very sensitive to error accumulation. However, the situation may change if we take into consideration the special type of objective function (quadratic form). It is clear that for objective functions the degree of which is more than quadratic, in case of high dimension, the method offered by us is more effective than the other methods mentioned above.

	Eason'sfunction	Powell's function	Rosenbrock's function	Wood's function
Steepest-descent	0.000/1E-7 0.001/1E-12	1.9188 /1E-5 39.8662/1E-7	0.184 /1E-5 0.481/1E-7	0.297/1E-5 1.1047/1E-7
std_gradient	1.3996/1E-7 3.085/1E-11 — /1E-12	26.525/1E-5 122.453/1E-6 —- / 1E-7	0.849/1E-7 1.505/1E-12	2.53/1E-7 3.02/1E-11 —/1E-12
modified-HeavyBall alpha=1.05; beta=1E-4	0.005/1E-6 —/1E-7	0.006/1E-6 0.011/1E-7 0.025/1E-12	0.002 /1E-6 0.002/1E-7 0.004/1E-12	0.009 /1E-6 0.007/1E-7 0.01/1E-12
modified-HeavyBall alpha=1.25; beta=1E-4	0.001/1E-6 —/1E-7	0.017/1E-6 0.038/1E-7 0.208/1E-12	0.001/1E-6 0.003/1E-7 0.003/1E-12	0.023/1E-7 0.023/1E-11 —/1E-12
modified-HeavyBall alpha=2.05; beta=1E-4	0.001/1E-5 —/1E-7	3.994/1E-6 18.488 /1E-7 —/1E-8	0.011/1E-7 0.013/1E-7	0.035 /1E-7 0.046/1E-11 —/1E-12
HeavyBall alpha=0.96 beta=1E-4	0.395/1E-7 0.429/1E-12	10.556/1E-7 —/1E-8	0.03/1E-7 0.056/1E-12	—
Fletcher-Reeves	0.000/1E-7	0.0045/1E-7	0.0016/1E-7	0.0015/1E-7

References

1. G. W. Brown and J. von Neumann, Solutions of games by differential equations. Contributions to the Theory of Games, pp. 73–79. *Annals of Mathematics Studies*, No. 24. *Princeton University Press, Princeton, N. J.*, 1950.
2. G. W. Brown, Iterative solution of games by fictitious play. *Activity Analysis of Production and Allocation*, Cowles Commission Monograph No. 13, 374–376, *John Wiley & Sons, Inc., New York, N. Y.; Chapman & Hall, Ltd., London*, 1951.
3. B. T. Polyak, Introduction to optimization. Translated from the Russian. With a foreword by Dimitri P. Bertsekas. Translations Series in Mathematics and Engineering. *Optimization Software, Inc., Publications Division, New York*, 1987.
4. H. A. Eiselt and C.-L. Sandblom, The bounce algorithm for mathematical programming. *Math. Methods Oper. Res.* **52** (2000), No. 2, 173–183.
5. S. Y. Chang and K. G. Murty, The steepest descent gravitational method for linear programming. *Discrete Appl. Math.* **25** (1989), No. 3, 211–239.
6. X. Goudou and J. Munier, The gradient and heavy ball with friction dynamical systems: the quasiconvex case. *Math. Program.* **116** (2009), No. 1-2, Ser. B, 173–191.
7. Tamas Hajba, Optimizing second-order differential equation systems. *Electron. J. Differential Equations* **2011**, No. 44, 16 pp.
8. A. Cabot, H. Engler and S. Gadat, On the long time behavior of second order differential equations with asymptotically small dissipation. *Trans. Amer. Math. Soc.* **361** (2009), No. 11, 5983–6017.

9. A. Bhaya, F. Pazos and E. Kaszkurewicz, The controlled conjugate gradient type trajectory-following neural net for minimization of nonconvex functions. *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN),* 18–23 *July, Spain, Barcelona,* 1–8, 2010.
10. Paul-Emile, Mainge, Asymptotic convergence of an inertial proximal method for unconstrained quasiconvex minimization. *J. Global Optim.* **45** (2009), No. 4, 631–644.
11. W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, Numerical recipes. The art of scientific computing. Third edition. *Cambridge University Press, Cambridge*, 2007.
12. Jorge J. More, Burton S. Garbow and Kenneth E. Hillstrom, Testing unconstrained optimization software. *ACM Trans. Math. Software* **7** (1981), No. 1, 17–41.
13. A. Ravindran, K. M. Ragsdell and G. V. Reklaitis, Engineering Optimization: Methods and Applications, Second Edition. *John Wiley & Sons, Inc., Hoboken, New Jersey*, 2006.

(Received 29.06.2012; revised 11.12.2012)

Authors' address:

Department of Computer Sciences
I. Javakhishvili Tbilisi State University
13 University St., Tbilisi 0186, Georgia
E-mail: koba.gelashvili@tsu.ge
lela.alkhazishvili@tsu.ge
irina.khutsishvili@gmail.com