Published in AJSE, Vol:19, Issue: 03
Received on 17th January 2020
Revised on 5th December 2020
Accepted on 30th December 2020

AIUB JOURNAL OF SCIENCE AND ENGINEERING
ISSN: 1608 – 3679 (print) 2520 – 4890 (Online)

On Gradient Descent and Coordinate Descent Methods and Its Variants

Sajjadul Bari, Md. Rajib Arefin, and Sohana Jahan

***Abstract*—This research is focused on Unconstrained Optimization problems. Among a number of methods that can be used to solve Unconstrained Optimization problems we have worked on Gradient and Coordinate Descent methods. Step size plays an important role for optimization. Here we have performed numerical experiment with Gradient and Coordinate Descent method for several step size choices. Comparison between different variants of Gradient and Coordinate Descent methods and their efficiency are demonstrated by implementing in loss functions minimization problem.**

***Index Terms*—Convex function, Coordinate descent, Differentiable function, Gradient descent, Lipschitz constant, L-smooth function, Unconstrained optimization.**

I. Introduction

UNCONSTRAINED optimization [12], [15] problem minimizes an objective function that depends on real variables with no restrictions on their values. Mathematically, if $x \in \mathbb{R}^n$ is a real vector with $n \geq 1$ components and if $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth function. Then, the unconstrained optimization problem is of the form

$$(P) \quad \min_x f(x) \\ s.t \;\; x \in \mathbb{R}^n$$

Thus we want to find an optimal decision, that is $x^* \in \mathbb{R}^n$ such that

$$f(x^*) \leq f(x), \quad \forall \;\; x \in \mathbb{R}^n.$$

Unconstrained optimization have been used in applications for many years, and their popularity continues to grow because of their usefulness in data analysis, machine learning, and other areas of current interest. Unconstrained optimization problems may arise directly in many applications or they may arise from reformulations of constrained optimization problems. Constraints of an optimization problem can be replaced in the objective function with penalized terms and the constrained optimization problem can be solved as an unconstrained problem. In this paper we have worked on iterative techniques for solving unconstrained optimization problem specifically on Gradient descent method and Coordinate Descent method.

Sajjadul Bari is with Department of Mathematics at American International University- Bangladesh as a Lecturer (e-mail: sajjadul@aiub.edu).

Md. Rajib Arefin is with Deaprtment of Mathematics at University of Dhaka as Assistant Professor. He is now on study leave for pursuing PhD at Kyushu University, Japan. (e-mail: arefin.math@gmail.com).

Sohana Jahan is with Deaprtment of Mathematics at University of Dhaka as Associate Professor. (e-mail: sjahan.mat@du.ac.bd).

A. Main Features of Algorithms

General structure of an iterative algorithm for solving unconstrained minimization problem is as follows

- Choose a starting point x_0.
- Beginning at x_0, generate a sequence of iterates $\{x_k\}_{k=0}^{\infty}$ with non-increasing function (f) value until a solution point with sufficient accuracy is found or until no further progress can be made.
  To generate the next iterate x_{k+1}, the algorithm uses information about the function at x_k and possibly earlier iterates.

Both the Gradient Descent method and Coordinate descent method follow the above two steps in minimizing a function.

- Step Length: A suitable step-length can help the initial guess to reach the goal in the fastest way. If the step-length is too long it may exceed the target and on the contrary, if it is too short the convergence will be slow.
- Descent Direction: At each iteration a descent direction has to determine. This direction is opposite to the direction of gradient of the function at the current point,

Here we will concentrate on the discussion of different aspects of Gradient Descent and Coordinate Descent method which will assist us to find a right descent direction.
The remainig part of this paper is organized as follows: In the next section we have discussed choice of step length and the respective algorithms of for Gradient descent method. The following section includes the discussion of co-ordinate descent method. Choice of co-ordinates to update at each itearion is also discussed briefly. A comparison bethween GD and CD for different choice of stepsize is shown numerically in section IV. In section V we have implemented the idea of GD abd CD in loss function minimization problem. We have concluded our results in section VI.

II. GD: Gradient Descent

The basic Gradient Descent Method [3], [10], [11], [4] is based on fixed step size. A variant of GD includes choice of differnt step size so that the algorithm performs efficiently. Moreover, step length can be chosen with backtracking armijo condition to get a better approximation. In this section we will discuss each of these variants of GD.
The following algorithms [3], [5], [4], [14] is on Gradient Descent Method of different variants. step size.

Algorithm 1.(Gradient Descent with Fixed Step Size)
The algorithm is initialized with a guess x_0, a maximum

iteration count N. It proceeds as follows:
Step 1: For $j = 1, 2, \ldots, N$.
repeat
Step 2: $x_{j+1} \leftarrow x_j - \alpha \nabla f(x_j)$.
Step 3: $x_j := x_{j+1}$
until termination test satisfied;

The termination criterion includes one of the followings.

- Whenever the maximum iteration N exceeds.
- Whenever there is no significant change in successive values of x. That is, whenever

$$\frac{\|x_{k+1} - x_k\|}{x_k} \leq \epsilon$$

for a small tolerance $\epsilon > 0$.

It should be noted that choice of fixed step size does not always perform properly for a particular situation. Step size adaptation during the algorithm plays an important role in finding a good approximation. For example:

- If the function value increases at some point after taking a step, that means we have chosen a large step. Decreasing the step size can fix the problem.

- If the function value decreases with a suitable choice of step size, then we have to verify the situation by increasing the length of the step.

Here we are ready to introduce two algorithms of Gradient Descent with Step Size adaptation

Algorithm 2. (Gradient Descent with Step Size adaptation) The algorithm is initialized with a guess x, a maximum iteration count N. It proceeds as follows:
Step 1: Repeat upto N iterations
Step 2: $y \leftarrow x - \alpha \nabla f(x)^T, \ f_y \leftarrow f(y)$
Step 3: *if* $f_y < f_x$, then
Step 4: $x \leftarrow y$
Step 5: $\alpha \leftarrow 1.2\alpha$
Step 6: *else* $\alpha \leftarrow 0.5\alpha$

Algorithm 3. (Gradient Descent with Backtracking Armijo) We start with some initial estimates: x, given step size α, β, τ, maximum iteration N.
step 1: For $j = 1, 2, \ldots, N$
step 2: $x_{new} = x - \alpha \nabla f(x)^T$
step 3: *if* $f(x_{new}) \leq f(x) - \beta\alpha \left\| \nabla f(x)^T \right\|^2$
step 4: Set $x = x_{new}$ and $\alpha =$ given value.
step 5: *else* $\alpha = \tau\alpha$. Then Go To Step 3.

A. Convergence of Gradient Descent Method

Different variants of gradient descent method depend on how the step size (α) is chosen. Smaller step size may lead to use huge computational time whereas, larger step size can over shoot the minimum point and therefore may fail to converge. Choice of the step sizes depends on the behavior of the function. In addition to that it can give an estimation on the number of iterations needed. Note that, gradient descent converge to a local minimum, even with the fixed step size. It is observed that as the iterates approach to a local minimum, gradient descent will automatically take smaller steps. Therefore, no need to decrease the step size over time.

The following result [1] gives an estimation of the number of iterations when the step size is constant.

Theorem 1: If ∇f is Lipschitz continuous with constant $L > 0$, then gradient descent with fixed step size $\alpha \leq \frac{1}{L}$ satisfies

$$f(x^k) - f(x^*) \leq \frac{\|x^0 - x^*\|^2}{2\alpha k}.$$

III. CD: Coordinate Descent

Coordinate descent [2], [6] algorithms solve optimization problems by successively performing approximate minimization along coordinate directions. They have been used in applications for many years. Recently they are being used in many research area such as data analysis, machine learning and so on. This paper describes the basic coordinate descent approach, together with variants. Coordinate descent (CD) algorithms are iterative methods in which each iterate is obtained by fixing most components of the variable vector x at particular values from the current iteration, and approximately minimizing the objective with respect to the remaining components. Each such subproblem is a lower dimensional minimization problem, and thus can be solved more easily than the original problem.
The goal is to solve

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

where f is convex and smooth (f is continuously differentiable and gradient is Lipschitz continuous).
When n is large, it becomes computationally expensive to calculate full gradients, which means gradient descent is not necessarily always efficient. Observe that for unconstrained problems, x^* is an optimal solution if $\nabla f(x^*) = 0$. To find the optimal solution, it makes sense to search along each coordinate direction. This motivates the so called Coordinate Descent Algorithms.

A. Rules for Selecting Coordinates

There are several ways and orders to decide which coordinate to update at each iteration.
Cyclic Order: Run all coordinates in cyclic order, that is $1 \to 2 \to \cdots \to n$.
Gauss-Southwell: At each iteration, pick coordinate i so that

$$i = \underset{1 \leq j \leq n}{\operatorname{argmax}} |\nabla_j f(x)^T|$$

Random Permutation: Run cyclic order on a permuted index(sample without replacement).
For example if $n = 3$ we could have the following:
Cyclic: 1^{st} iteration: $(1 \to 2 \to 3)$,
2^{nd} iteration: $(1 \to 2 \to 3)$,
3^{rd} iteration: $(1 \to 2 \to 3) \ldots$
Random Permutation: 1^{st} iteration: $(1 \to 2 \to 3)$,
2^{nd} iteration: $(3 \to 1 \to 2)$,
3^{rd} iteration:$(2 \to 1 \to 3) \ldots$

B. Iterative Notion for Coordinate Descent

Starting with some initial guess x^0, the successive approximations are calculated by repeating the following process for $k = 1, 2, 3, \ldots$

$$\begin{aligned} x_1^k &\in \underset{x_1}{\operatorname{argmin}}\ f(x_1, x_2^{k-1}, x_3^{k-1}, \ldots, x_n^{k-1}) \\ x_2^k &\in \underset{x_2}{\operatorname{argmin}}\ f(x_1^k, x_2, x_3^{k-1}, \ldots, x_n^{k-1}) \\ x_3^k &\in \underset{x_3}{\operatorname{argmin}}\ f(x_1^k, x_2^k, x_3, \ldots, x_n^{k-1}) \\ &\ldots \\ x_n^k &\in \underset{x_n}{\operatorname{argmin}}\ f(x_1^k, x_2^k, x_3^k, \ldots, x_n). \end{aligned}$$

Here the variables are updated in Gauss-Seidel style.

C. Coordinate Descent Algorithms and Convergences

Here we have discussed the alroithms of Coordinate descent methods. The convergence of these algorithms are followed from [9], [7], [8]

Algorithm 4. Gauss-Southwell Coordinate Descent
Set $t \leftarrow 0$ and choose $x^0 \in \mathbb{R}^n$;
repeat
Step 1: choose index $i_t = argmax_{1 \le j \le n} \left| \nabla_j f(x^{(t)}) \right|$;
Step 2: $x^{(t+1)} \leftarrow x^{(t)} - \frac{1}{L} U_{i_t} \nabla_{i_t} f(x^{(t)})$;
Step 3: $k \leftarrow k + 1$;
until termination test is satisfied;

Theorem 2: If f is convex and L-smooth, then

$$f(x^{(t)}) - f^* \le \frac{2Ln \left\| x_0 - x_* \right\|^2}{2\alpha k}$$

Algorithm 5. Cyclic Coordinate Descent:
Set $t \leftarrow 0$ and choose $x^0 \in \mathbb{R}^n$;
repeat
Step 1: at iteration t, for $i = 1, 2, \ldots, n$;
Step 2: $x_i^{(t)} \leftarrow x_{i-1}^{(t)} - \frac{1}{L} U_i \nabla_i f(x_{i-1}^{(t)})$;
Step 3: set $x^{(t+1)} = x_n^{(t+1)}$;
Step 4: $k \leftarrow k + 1$;
until termination test is satisfied;

Theorem 3: If f is convex and L-smooth, then

$$f(x^{(t)}) - f^* \le \frac{4L(n+1)R(x_0)^2}{t}.$$

Where $R(x_0) = max\left\{ \left\| x - x^* \right\| : f(x) \le f(x_0) \right\}$.

IV. Numerical Experiments with GD and CD

For numerical experiment we consider two functions

$$f_1(x, y) = x^4 + 2x^3 + 2x^2 + y^2 - 2xy \qquad (1)$$

$$f_2(x, y) = 4x^2 - 6xy + 5y^2 - 20x + 40 \qquad (2)$$

which are convex and differentiable at every point. The local minimal of the function f_1 is $(0, 0)$ and f_2 is $\left(\frac{50}{11}, \frac{30}{11}\right)$. We start with an initial point $(1, 3)$ and $(7, 6)$ for function f_1 and f_2 respectively. Besides we observe how different approaches work to get a good approximation of the exact result.

A. GD with Different Step Sizes

First we have applied Gradient Descent algorithm with step size variation i) fixed step size, ii) step size adaptation and iii) with Backtracking Armijo condition on both functions. We started with stepsize $\alpha_1 = 0.09$ for f_1 and $\alpha_2 = 0.05$ for f_2 and run our experiment upto 50 iteartions. Some successive approximations are shown in the following tables. Graphical representations demonstrates the improvement in each iteration of GD for different step sizes. In the next turn, the step size adaptation rule has been used to improve the approximation.

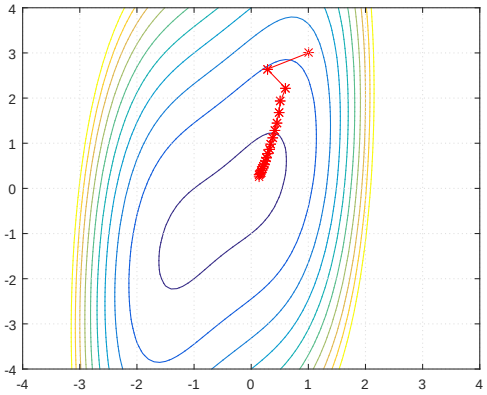


Fig. 1: GD with fixed step size on $f_1(x, y)$.

TABLE I: Some iterations of GD on $f_1(x, y)$ with fixed step size

Iteration No.	$\lvert f_1(x_{exact}) - f_1(x) \rvert$
1	5.698051
2	3.381916
⋮	⋮
47	0.000449
48	0.000387
49	0.000333
50	0.000287

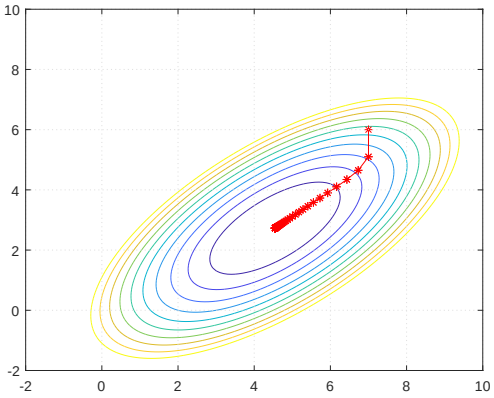


Fig. 2: GD with fixed step size on $f_2(x, y)$.

TABLE II: Some iterations of GD on $f_2(x, y)$ with fixed step size

Iteration No.	$\lvert f_2(x_{exact}) - f_2(x)\rvert$
1	17.30454545
2	12.37164545
⋮	⋮
47	0.0000085
48	0.0000062
49	0.0000045
50	0.0000033

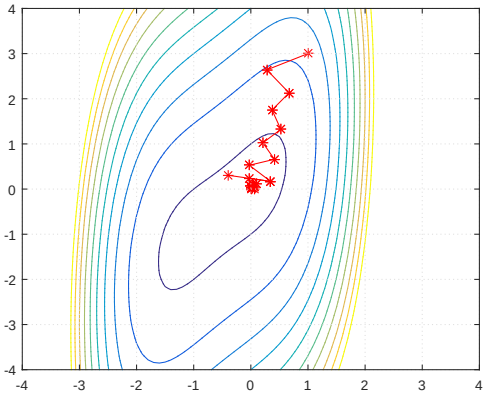


Fig. 3: GD with step size adaptation on $f_1(x, y)$.

TABLE III: Some iterations of GD with step size adaptation on $f_1(x, y)$

Iteration No.	$\lvert f_1(x_{exact}) - f_1(x)\rvert$	Adaptive step
1	5.698051	0.108
2	3.53474	0.1296
⋮	⋮	⋮
47	1.78×10^{-10}	0.430597
48	1.66×10^{-10}	0.516717
49	1.66×10^{-10}	0.258358
50	4.46×10^{-11}	0.31003

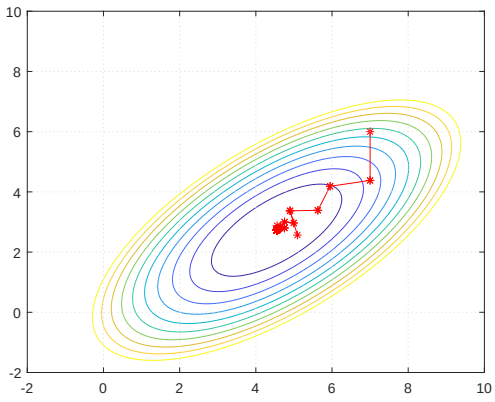


Fig. 4: GD with step size adaptation on $f_2(x, y)$.

TABLE IV: Some iterations of GD with step size adaptation on $f_2(x, y)$

Iteration No.	$\lvert f_2(x_{exact}) - f_2(x)\rvert$	Adaptive step
1	13.41655	0.108
2	6.235459	0.1296
⋮	⋮	⋮
47	2.32×10^{-12}	0.074756
48	2.84×10^{-13}	0.089708
49	1.35×10^{-13}	0.107649
50	7.82×10^{-14}	0.129179

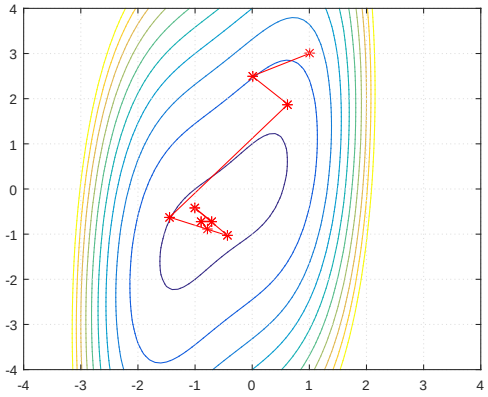


Fig. 5: GD with Backtracking Armijo on $f_1(x, y)$.

TABLE V: Some iterations of GD on $f_1(x, y)$ with Backtracking Armijo

Iteration No.	$\lvert f_1(x_{exact}) - f_1(x)\rvert$
1	8
2	8
⋮	⋮
47	6.29×10^{-06}
48	3.88×10^{-06}
49	3.88×10^{-06}
50	3.88×10^{-06}

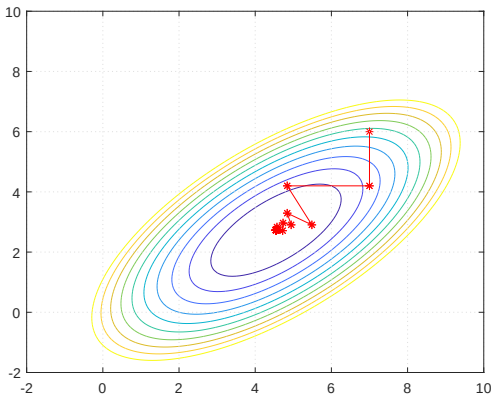
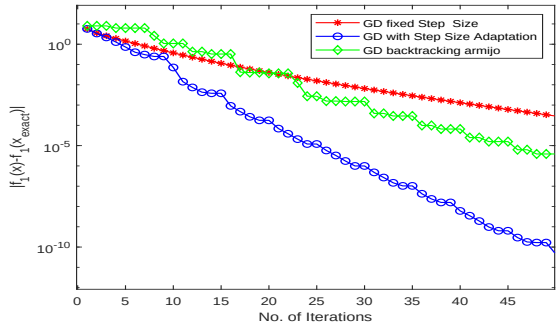


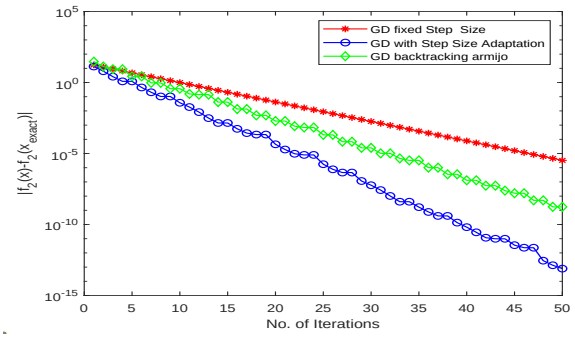
Fig. 6: GD with Backtracking Armijo on $f_2(x, y)$.

TABLE VI: Some iterations of GD on $f_2(x, y)$ with Backtracking Armijo

Iteration No.	$\lvert f_2(x_{exact}) - f_2(x)\rvert$
1	29.454545
2	13.254545
⋮	⋮
47	5.02×10^{-09}
48	5.02×10^{-09}
49	1.75×10^{-09}
50	1.75×10^{-09}



(a) on $f_1(x, y)$



(b) on $f_2(x, y)$

Fig. 7: Comparison on GD with different step sizes for $f_1(x, y)$ in (a) and $f_2(x, y)$ in (b)

In Figure 7(a) we have shown the performance of GD on $f_1(x, y)$ for different step size in same frame which demonstrates that that for gradient descent method, step size adaptation technique works better than others. Though upto 8th iteration all of them converges to same approximation but after 8th iteration error term decreases dramatically for step size adaptation technique because it pays careful attention to both increasing and decreasing the value of the function. On the other hand, backtracking armijo technique only decreases the value of the function against a specific condition. Again for the case of fixed step size, it only goes towards descent direction with a certain step length.

Whereas in figure 7(b) the experiment is showed on $f_2(x, y)$ and it also results the better performance of step size adaptation technique compared to fixed step size and backtracking armijo rule.

B. CD with Different Step Sizes

In this section we have applied the three techniques of choosing step size a on coordinate descent method. The tables shows that step size adaptation technique gives better approximations for coordinate descent method in 50 iteration.

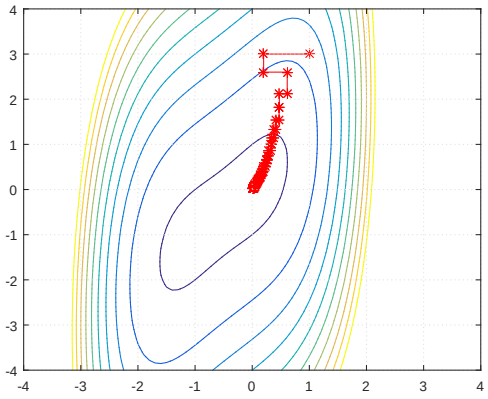


Fig. 8: CD with fixed step size on $f_1(x, y)$.

TABLE VII: Some iterations of CD on $f_1(x, y)$ with fixed step size

Iteration No.	$\lvert f_1(x_{exact}) - f_1(x)\rvert$
1	5.8176
2	3.2484366
⋮	⋮
47	0.000196
48	0.000166
49	0.000141
50	0.00012

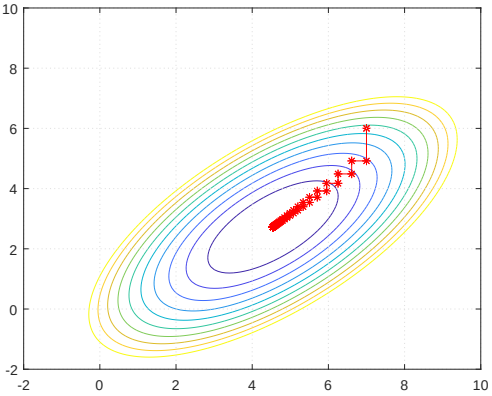


Fig. 9: CD with fixed step size on $f_2(x, y)$.

TABLE VIII: Some iterations of CD on $f_2(x, y)$ with fixed step size

Iteration No.	$\lvert f_2(x_{exact}) - f_2(x)\rvert$
1	15.846545
2	10.746733
⋮	⋮
47	3.24×10^{-7}
48	2.2×10^{-7}
49	1.5×10^{-7}
50	1.02×10^{-7}

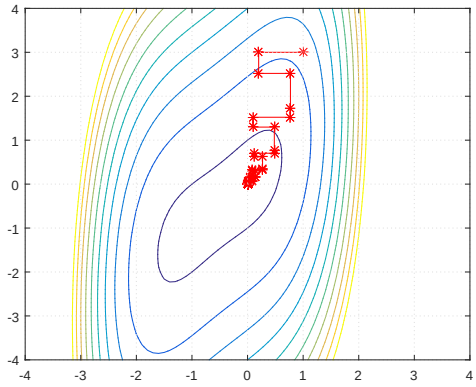


Fig. 10: CD with step size adaptation on $f_1(x, y)$.

TABLE IX: Some iterations of CD with step size adaptation on $f_1(x, y)$

Iteration No.	$\lvert f_1(x_{exact}) - f_1(x)\rvert$	Adaptive step
1	5.44	0.144
2	2.763612	0.20736
⋮	⋮	⋮
47	2.76×10^{-11}	0.397406
48	1.77×10^{-11}	0.238444
49	5.9×10^{-12}	0.343359
50	3.23×10^{-12}	0.494437

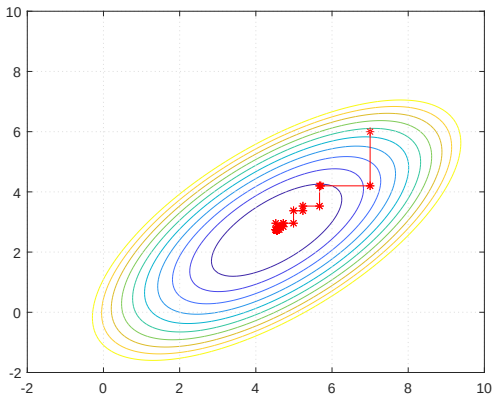


Fig. 11: CD with step size adaptation on $f_2(x, y)$.

TABLE X: Some iterations of CD with step size adaptation on $f_2(x, y)$

Iteration No.	$\lvert f_2(x_{exact}) - f_2(x)\rvert$	Adaptive step
1	13.254545	0.12
2	5.976209	0.072
⋮	⋮	⋮
47	1.42×10^{-14}	3.43×10^{-9}
48	1.42×10^{-14}	8.58×10^{-10}
49	1.42×10^{-14}	2.15×10^{-10}
50	1.42×10^{-14}	5.36×10^{-11}

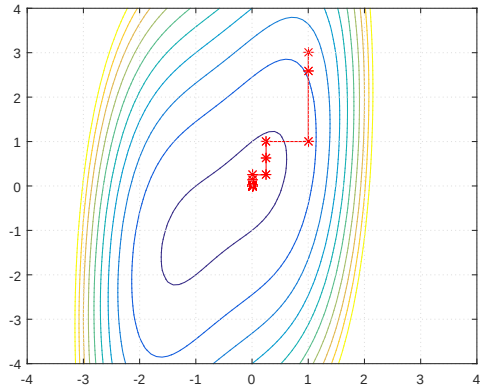


Fig. 12: CD with backtracking armijo on $f_1(x, y)$.

TABLE XI: Some iterations of CD on $f_1(x, y)$ with backtracking armijo condition

Iteration No.	$\lvert f_1(x_{exact}) - f_1(x)\rvert$
1	6.56
2	4
⋮	⋮
47	7.74×10^{-10}
48	7.74×10^{-10}
49	3.87×10^{-10}
50	2.42×10^{-10}

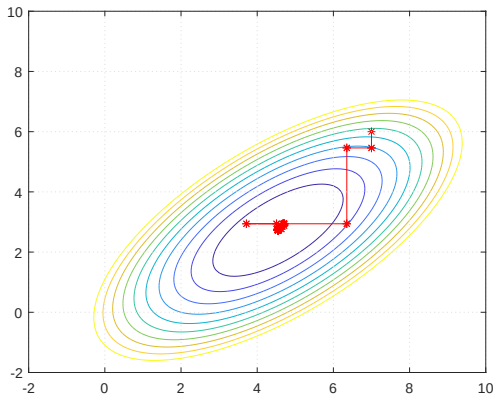
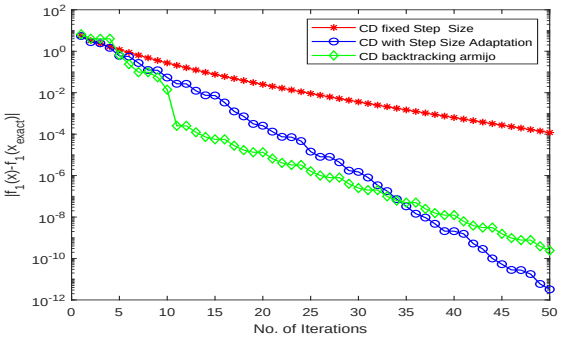


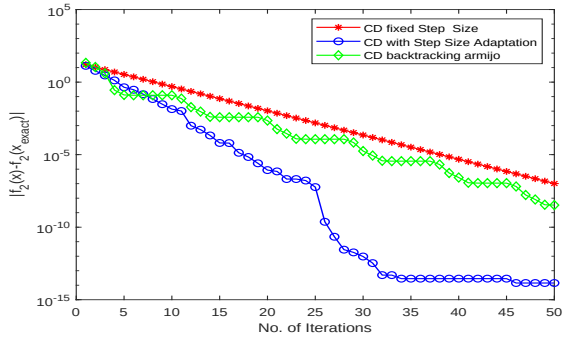
Fig. 13: CD with backtracking armijo on $f_2(x, y)$.

TABLE XII: Some iterations of CD on $f_2(x,y)$ with backtracking armijo condition

Iteration No.	$\lvert f_2(x_{exact}) - f_2(x)\rvert$
1	21.192545
2	10.974881
⋮	⋮
47	1.66×10^{-8}
48	8.11×10^{-9}
49	3.52×10^{-9}
50	3.34×10^{-9}



(a) on $f_1(x,y)$



(b) on $f_2(x,y)$

Fig. 14: Comparison on CD with different step sizes for $f_1(x,y)$ in (a) and $f_2(x,y)$ in (b)

Figure 14 (a) and (b) gives the comparison of three cases of CD on both $f_1(x,y)$ and $f_2(x,y)$ respectively in the same frame which implies that step size adaptation technique gives better approximations for coordinate descent method but if we stop before 35^{th} iteration (figure (a)) then backtracking armijo technique gives more accuracy than step size adaptation. For a better accuracy one can insert the stopping criterion.

$$\Big|f(x) - f(x_{exact})\Big| < \epsilon; \text{ suppose } \epsilon = 10^{-05}$$

The more the ϵ tends to zero, the more accuracy will be obtaind. After a certain number of iterations, step size adaptation technique will perform better than backtracking armijo.

Comment :
The resulting comparison on GD and CD may vary considering different choices of functions.

V. Loss Function Minimization

We are going to define loss function [16], [5], [17] as

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}(a_i^T x - b_i)^2$$

Where a_i is a row vector, x and b_i are column vectors.

This is the least-squares loss function that gives rise to the ordinary least squares regression model. The loss function is obviously convex function. Minimizing an arbitrary function is, in general, very difficult, but if the objective function to be minimized is convex then things become considerably simpler. The key advantage of dealing with convex function is that a local optima is also a global optima.

We will concentrate on numerical experiment with $L2-$regularized least square problem using gradient descent and coordinate descent algorithm. We consider,

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}(a_i^T x - b_i)^2 + \frac{\lambda}{2}\|x\|^2$$

The main goal is to predict x that minimizes the loss function

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

The function $f_1, f_2, f_3, ..., f_n$ are assumed to be $L-smooth$. Clearly, $(a_i^T x - b_i)^2$ is convex, therefore, $f(x)$ is strongly convex [1] with λ.

We can estimate the Lipschitz constant L_i for the function f_i as $(2\|a_i\|^2 + \lambda)$. Thus Lipschitz constant for $f(x)$ would be $\max_{1\le i\le n}\{L_i\}$. In this case, a training set of 50 examples are being considered. Each example comprises 30 features. That is $n = 50$ and $d = 30$. The entries of a_i are taken as random integers from 1 to 10. The constant λ is considered as $\frac{1}{n}$.

We have applied both GD and CD and run the algorithms for 100 iterations.

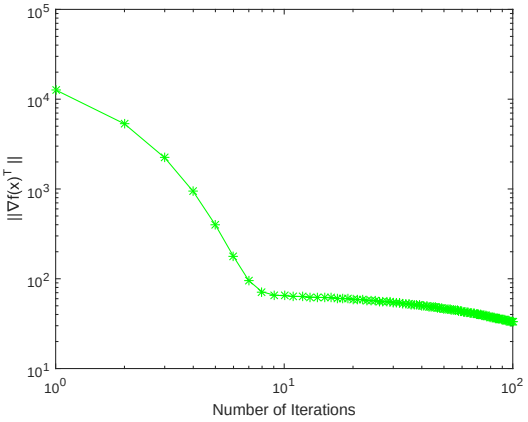


Fig. 15: Loss function minimization with GD.

[1] A convex function f is strongly convex if and only if, there exist a constant $\mu > 0$ such that the function $f(x) + \frac{\mu}{2}\|x\|^2$ is convex.

TABLE XIII: Some observation of GD for loss function minimization

Iteration No.	$\|\nabla f(x)\|$
1	12678.71
2	5314.989
⋮	⋮
97	33.5855
98	33.37241
99	33.16144
100	32.95256

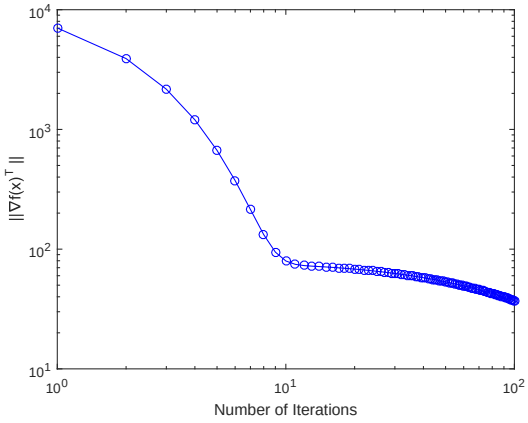


Fig. 16: Loss function minimization with CD.

TABLE XIV: Some observation of CD for loss function minimization

Iteration No.	$\|\nabla f(x)\|$
1	7038.337
2	3906.364
⋮	⋮
97	37.80994
98	37.55346
99	37.29952
100	37.04807

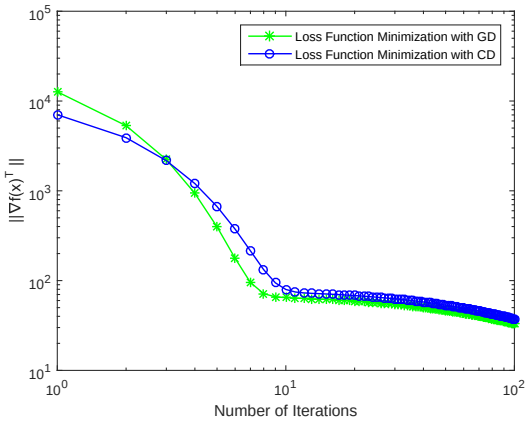


Fig. 17: GD vs CD for Loss function minimization.

Figure 17 represents the performance of GD and CD in loss function minimization for 100 iterations. It is clearly seen that GD works better than CD. But fo a large number of iterations the performance of two methods are quite similar.

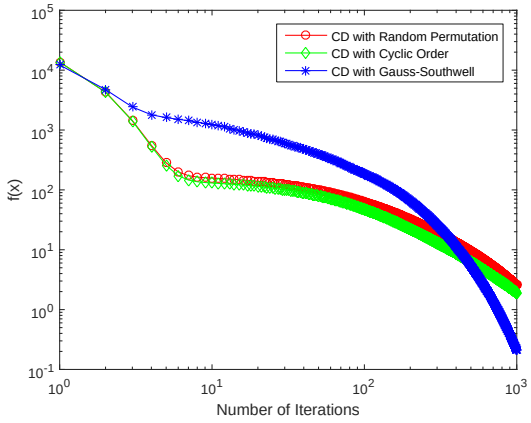


Fig. 18: GD vs CD for Loss function minimization.

We have applied CD on loss function minimization with different choice of selecting coordinates for the update. From figure 18 it can be concluded that though at the initial stage Gauss-Southwell was slow in convergence but after a certain iteration it works better then other two techniques.

VI. Conclusion

In this paper we have worked on unconstrained optimization problem. Specially we are focused on performance of iterative techniques Gradient Descent (GD) method and Coordinate Descent (CD) method in solving the unconstrained optimization problem. We have applied both GD and CD for different choice of step size and check their performance in minimizing a problem. Numerical results show that for both GD and CD method step size adaptation technique converges faster until a certain number of iterations. And also we may conclude that this comparison may vary for different choices of function and step size. A comparison between these two for solving loss function minimization is also demonstrated which shows that for loss function minimization problem GD works better than CD in general. In case of CD, Gauss-Southwell technique can be used for the updates of coordinates at different iteration for getting faster convergence.

References

[1] J. Désidéri. "Multiple-gradient descent algorithm (MGDA) for multiobjective optimization," *Comptes Rendus Mathematique*, vol. 350, no. 5, pp. 313-318, Mar. 2012. DOI: https://doi.org/10.1016/j.crma.2012.03.014

[2] M. Blondel *et al* "Block coordinate descent algorithms for large-scale sparse multiclass classification," *Machine Learning*, vol. 93, no. 1, pp. 31-52, May 2013. DOI: https://doi.org/10.1007/s10994-013-5367-2

[3] S. Ruder, "An overview of gradient descent optimization algorithms," Jun. 2017. https://arxiv.org/abs/1609.04747v2

[4] S. Wright, and J. Nocedal. "Numerical optimization," *Springer Science*, vol. 35, no. 7, pp. 35 (1999): 67-68.

[5] M. R. Arefin, and M. Asadujjaman., "Minimizing Average of Loss Functions Using Gradient Descent and Stochastic Gradient Descent," *The Dhaka University Journal of Science*, Vol. 64, no. 2, pp. 141-145, Jul. 2016.

[6] S. J. Wright, "Coordinate descent algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 3-34, Mar. 2015.
[7] P. Richtárik, and M. Takáč, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Mathematical Programming*, vol. 144, no.1-2, pp. 1-38, Apr. 2014.
[8] A. Beck, and L. Tetruashvili, "On the convergence of block coordinate descent type methods," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2037-2060, Oct. 2013.
[9] Y. Nesterov, "Efficiency of coordinate descent methods on huge-scale optimization problems," *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 341-362, Apr. 2012.
[10] Y. Nesterov, "Introductory Lectures on Convex Optimization, A Basic Course," Springer, 2003.
[11] S. Boyd *et al.*, "Unconstrained minimization," in *Convex Optimization*, 1st ed., Cambridge University Press, 2004, ch. 9, pp. 457-475. DOI: https://doi.org/10.1017/CBO9780511804441
[12] J. Nocedal, "Theory of algorithms for unconstrained optimization," *Acta Numerica*, vol. 1, pp. 199-242, Jan. 1992. DOI: https://doi.org/10.1017/S0962492900002270
[13] X. Wang, "Method of steepest descent and its applications," *IEEE Microwave and Wireless Components Letters*, vol. 12, pp. 24-26, Nov. 2008.
[14] D.P. Mandic, "A generalized normalized gradient descent algorithm," *IEEE signal processing letters*, vol. 11, no. 2, pp. 115-118, Feb. 2004. DOI: https://doi.org/10.1109/LSP.2003.821649
[15] Astolfi, A. "Optimization, An Introduction." Estados Unidos, Octubre del (2005).
[16] L. Armijo, "Minimization of functions having Lipschitz continuous first partial derivatives," *Pacific Journal of mathematics*, vol. 16, no. 1, pp. 1-3, Jan. 1966. DOI: https://doi.org/10.2140/pjm.1966.16.1
[17] D.P. Bertsekas, "Unconstrained Optimization," in *Nonlinear Programming,* 2nd ed. Belmont, Massachusetts, Athena scientific, 1999, ch. 1, sec. 1-8, pp. 4-160.

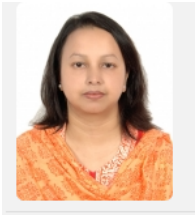
Sajjadul Bari completed his M.S and B.S(Hons) degree from University of Dhaka at Department of Mathematics. He became a life member in Bangladesh Mathematical Society in 2020. His research works focus on Operations Research. He also grew his interest on some topics with collaboration of other fields of study. In addition to all these, he is also studying books and articles related to Graph Theory to pursue his higher education later.

He is currently working as a Lecturer in Mathematics at American International University-Bangladesh. He has been working here since September 10, 2018. Prior to that, he was employed with Daffodil International University as a Lecturer. He participated several workshops, webinars and trainings to gather professional skills.

Md. Rajib Arefin completed his MSc in Financial Operational Research (FOR) (with Distinction), from University of Edinburgh, UK. He has received Dean's award from the faculty of Science, University of Dhaka for extraordinary performance in BSc (Hons.) examination. He also has been honored by A F Mujibur Rahman gold medal (twice) for securing first position in BSc and MS examinations. Besides these, he acquired University Grants Commission (UGC) of Bangladesh scholarship for excellence in Mathematics in BSc examination.

His Research interests lie in the field of Operations Research. He has also interest in applying modern optimization techniques in financial models. His another interest is evolving into the numerical simulations of Stochastic Differential Equations (SDEs).

He is currently working as an Assistant Professor (on study leave for pursuing PhD at Kyushu University, Japan) in Department of Mathematics at University of Dhaka. Before that, he had been working as a Lecturer in the same place since 2015. He had been teaching several undergraduate courses such as, Calculus, Differential equations, Mathematical Methods, Linear Algebra, Introduction to Mathematical Finance for the last couple of years. He taught an MSc level course: Operations Research as well. Besides all these, he supervised several undergraduate project students and also completed supervision of a few M.S (Thesis) students.



Sohana Jahan received the PhD in Operational Research from School of Mathematics, University of Southampton, UK in 2017, obtaining the Commonwealth Scholarship for that PhD program. She got A. F. Mujibur Rahman foundation gold medal for excellence in Pure Mathematics. She also acquired Mitra-Yusuf trust scholarship for the best result during third year of undergraduation.

Currently she is working as an Associate Professor at Department of mathematics, University of Dhaka. Before that she joined at University of Dhaka as a Lecturer in 2010 and promoted as an Assistant Professor in 2017. She also worked as a Lecturer at department of Mathematics, BUET in 2010.