Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 7, n. 1, 2020.

Trabalho apresentado no XXXIX CNMAC, Uberlândia - MG, 2019.

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Fractional Order Gradient Descent Algorithm

Eliana Contharteze Grigoletto [1]
Departamento de Bioprocessos e Biotecnologia, FCA, UNESP, Botucatu, SP
Aurelio Ribeiro Leite de Oliveira [2]
Departamento de Matemática Aplicada, IMECC, UNICAMP, SP

Abstract. We present the gradient descent algorithm for unconstrained minimization problems using tools of the fractional calculus, a field of mathematics with applications in widespread areas of science and engineering. At each iteration of the fractional order gradient descent algorithm, the search direction is determined by means of fractional gradient, being a new alternative to solve large scale minimization problems involving one useful class of functions.

Keywords. Fractional Calculus, Gradient Descent Algorithm, Unconstrained Minimization

1 Introduction

The gradient descent algorithm [1,3,11,21] has been employed for solving optimization problems, being commonly used to minimize an error function in the research of artificial neural networks [5–7,19,23], solve least-squares minimization problems involving phase retrieval [4,10,14,16], learning algorithm of fuzzy system [12,18], logistic regression [2,15, 20], just to name a few.

The so-called fractional calculus was conceptualized in connection with the infinitesimal calculus since 1695 [17]. Some of the areas of applications of fractional calculus include viscoelasticity, signal processing, probability, statistics, electrochemistry, diffusion in porous media, fluid flow, backpropagation training of neural networks, and so on [9,13].

Investigations of the fractional order gradient descent algorithm has shown interest, for example, in learning of backpropagation neural networks, however, such investigations began only recently [8,22]. Chen et al. [8] presented the fractional order gradient methods (FOGMs) by writing the Riemann-Liouville and Caputo fractional derivatives as Taylor series. Wang et al. [22] proposed a fractional gradient descent method, employing the Caputo derivative, for the backpropagation training of neural networks.

We present a new fractional order gradient descent algorithm, in the Caputo sense, by using a well-known particular property of this fractional operator for fractional derivative of powers function. We apply the fractional order gradient descent algorithm for minimizing an objective function which is written in terms of a sum of powers of $(x-a)^{\mu}$.

[1]eliana.contharteze@unesp.br.
[2]aurelio@ime.unicamp.br.

The paper is organized as follows. In Section 2 we describe the definition and a particular property of the Caputo fractional derivative. The fractional version of the gradient descent algorithm is reported in Section 3. In Section 4 some numerical results and illustration of the fractional algorithm will be given and compared to the classical algorithm.

2 Caputo fractional derivative

In this section we present the definition and a particular property of the Caputo fractional derivative [13].

For the definition below, $\left({}^{\mathrm{C}}\mathrm{D}^{\alpha}_{a+}f\right)(x)$ denotes Caputo left-sided fractional derivative of order $\alpha > 0$, with $\alpha \notin \mathbb{N}$, of functions on a subset $\Omega = [a, b]$ of real axis $\mathbb{R} = (-\infty, \infty)$, where $f \in \mathrm{C}^n[a, b]$ and[3] $n = [\alpha] + 1$.

$$\left({}^{\mathrm{C}}\mathrm{D}^{\alpha}_{a+}f\right)(x) := \frac{1}{\Gamma(n-\alpha)}\left(\int_a^x \frac{f^{(n)}(\tau)}{(x-\tau)^{1-n+\alpha}}\,\mathrm{d}\tau\right), \qquad \text{for } x \in \Omega, \tag{1}$$

where

$$\Gamma(\alpha) = \int_0^\infty t^{\alpha-1}e^{-t}\mathrm{d}t, \tag{2}$$

is the Gamma function. In particular, $\Gamma(n+1) = n!$ $(n \in \mathbb{N})$.

If $\alpha = n \in \mathbb{N}$, then $\left({}^{\mathrm{C}}\mathrm{D}^{n}_{a+}f\right)(x) = f^{(n)}(x)$.

Property 2.1. Let $\alpha > 0$, with $\alpha \notin \mathbb{N}$, and let n be given by $n = [\alpha]+1$. Also let $\mu > -1$, then for $x > a$ we have

$${}^{\mathrm{C}}\mathrm{D}^{\alpha}_{a+}(x-a)^{\mu} = \begin{cases} \dfrac{\Gamma(\mu+1)}{\Gamma(\mu-\alpha+1)}(x-a)^{\mu-\alpha}, & \text{for } \mu > [\alpha], \\ 0, & \text{for } \mu = 0, 1, 2, ..., [\alpha]. \end{cases} \tag{3}$$

In particular, if $\mu = 0$ and k is a constant, then $\left({}^{\mathrm{C}}\mathrm{D}^{\alpha}_{a+}\,k\right)(x) = 0$.

Example 2.1. Consider the quadratic function $\varphi_1(x) = (x-3)^2$. The fractional derivatives of φ_1, in terms of the Caputo fractional derivatives, in accordance with Property 2.1, takes the form

$${}^{\mathrm{C}}\mathrm{D}^{\alpha}_{3+}\varphi_1(x) = \frac{2\,(x-3)^{2-\alpha}}{\Gamma(3-\alpha)}, \qquad \text{for } x > 3 \text{ and } 0 < \alpha < 2, \tag{4}$$

and

$${}^{\mathrm{C}}\mathrm{D}^{\alpha}_{3+}\varphi_1(x) = 0, \qquad \text{for } x > 3 \text{ and } \alpha > 2. \tag{5}$$

[3] $[\mu]$ indicates the integer part of μ.

3 Fractional order gradient descent algorithm

The simplest gradient descent algorithm is a line search method and can be used to solve an unconstrained minimization problem of the form [11]:

$$\underset{\boldsymbol{x}\in\mathcal{D}}{\text{minimize}} f\left(\boldsymbol{x}\right), \tag{6}$$

where $f : \mathcal{D} \subseteq \mathbb{R}^n \to \mathbb{R}$ is a convex funtion, defined and continuously differentiable on open set $\mathcal{D}$ (neighborhood of the solution) in $\mathbb{R}^n$.

The iterative fractional order gradient descent algorithm for solving equation (6) start with an initialization $\boldsymbol{x}_0 \in \mathcal{D}$, and inductively update by the following iteration

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \eta_k \nabla^\alpha f\left(\boldsymbol{x}_k\right), \tag{7}$$

where $\eta_k > 0$ is an appropriately chosen step size parameter, $\alpha > 0$ and ∇^α is a vector whose components are partial fractional derivatives, determined by Property 2.1.

In the so-called fixed step size gradient descent algorithm, the step size η_k is fixed at each iteration. However, to refine the iteration, we can choose the appropriate step size η_k at kth iteration.

We define the fractional order gradient descent algorithm as described below.

Algorithm 1 Fractional order gradient descent algorithm

```
Input: $\boldsymbol{x}_0$, $\eta_0$, $\epsilon$, $s$
Output: $\boldsymbol{x}_k$
1: $k \leftarrow 0$
2: for $k = 1, 2, \ldots, s$ do
3:     $\boldsymbol{x}_k \leftarrow \boldsymbol{x}_{k-1} - \eta_{k-1} \nabla^\alpha f\left(\boldsymbol{x}_{k-1}\right)$
4:     Update the step size $\eta_k$
5:     $k \leftarrow k + 1$
6:     if $\|\nabla f\left(\boldsymbol{x}_k\right)\| < \epsilon$ then
7:         break
8:     end if
9: end for
```

4 Numerical experiments

Here we present some numerical experiments through the use of Algorithm 1 in illustrative examples. The numerical experiments were performed under Windows 10 and Matlab (R2016a) running on a desktop with 2.20 GHz Intel Core i5-5200 central processing unit (CPU) and 4G random-access memory (RAM).

We present our simulations for functions of the type:

$$\varphi(x) = (x - a)^\mu,$$

with fractional derivatives of order $0 < \alpha < 2$ and $\mu > [\alpha]$.

Example 4.1. We consider the quadratic function $f(x) = (x-3)^2$ and start with an initialization $x_0 > 3$. Thus, we set $x_0 = 5$, $\eta_0 = 0.1$, $\epsilon = 10^{-4}$ and $s = 10^5$.

The results of performance of Algorithm 1 with different fractional derivatives and fixed step size are shown in Figure 1, where one can observe that the sequence $\{x_k\}$ converges to the extreme point.

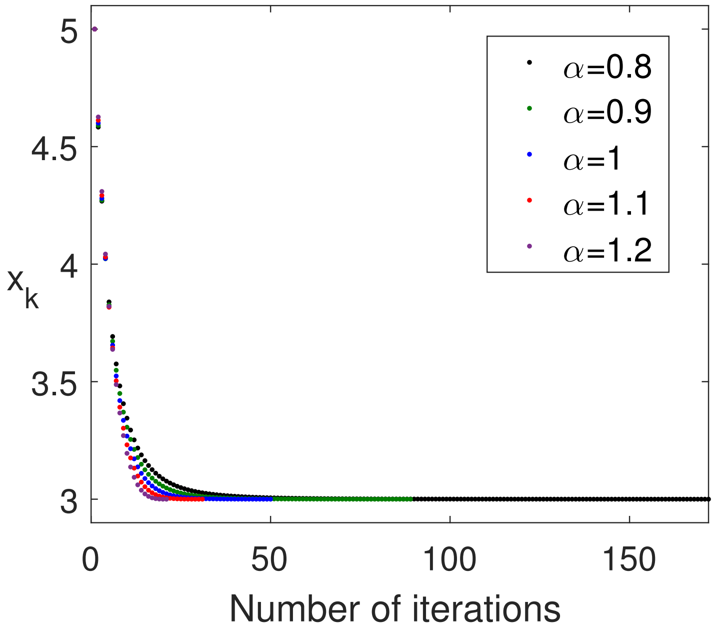

Figure 1: Performance of the Algorithm 1 for the function f.

Number of iterations, errors of approximation and CPU times in milliseconds (ms) consumed are shown below in Table 1.

Table 1: Numerical experiments for f with fractional order derivatives.

α	Number of iterations	Errors	Times (ms)
0.8	171	$4.772 \cdot 10^{-5}$	3.663
0.9	88	$4.335 \cdot 10^{-5}$	3.375
1.0	49	$3.568 \cdot 10^{-5}$	3.322
1.1	30	$1.608 \cdot 10^{-5}$	3.288
1.2	20	$2.598 \cdot 10^{-5}$	3.280

These results imply the numerical Algorithm 1 perform well in the right neighborhood around the point $a = 3$, that is the solution to the problem, when we consider $1 < \alpha < 1.2$.

In particular, when $\alpha = 1.3$, the algorithm stopped after $s = 10^5$ iterations, i.e., the algorithm stopped due to achieves the maximum number of iterations. The error of approximation was given by $1.425 \cdot 10^{-4}$.

Example 4.2. Now, we consider the minimizing of a function of 791 variables, given by

$$g\left(x_1, x_2, \ldots, x_{791}\right) = \sum_{i=1}^{791} \left(x_i - a_i\right)^2,$$

where $a_i = 1+0.1(i-1)$. We start with an initialization $\boldsymbol{x}_0 = (a_1 + 2, a_2 + 2, \ldots, a_{791} + 2)$. In this case, we set $\eta_0 = 0.1$, $\epsilon = 10^{-4}$ and $s = 10^5$. Results of this simulation, with fixed step size, are shown in Table 2.

Table 2: Numerical experiments for g with fractional order derivatives.

α	Number of iterations	Errors	Times (ms)
0.8	308	$9.950 \cdot 10^{-5}$	37.932
0.9	129	$9.565 \cdot 10^{-5}$	16.178
1.0	61	$8.620 \cdot 10^{-5}$	6.349
1.1	33	$5.193 \cdot 10^{-5}$	5.995

When $\alpha = 1.2$, the algorithm stopped after the maximum number of iterations and the error of approximation was given by $5.961 \cdot 10^{-4}$.

5 Conclusions

This paper has introduced a fractional approach to improve the classical gradient descent algorithm for minimizing an objective function which is written in terms of a sum of powers of $(x - a)^\mu$ by using a well-known property for fractional derivatives of power function. The comparative experimental results have been carried out between fractional and classical versions of the algorithm. The fractional algorithm ($\alpha \neq 1$) show a better performance than the classical algorithm ($\alpha = 1$). The future work would be to use a modified version of the algorithm choosing the appropriate step size and changes the fractional order along the method iterations.

References

[1] H. Akaike. On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method, *Ann. Inst. Statist. Math.*, 11:1–17, 1959.

[2] M. R. Amini, and P. Gallinari. Semi-supervised logistic regression. In *Proceedings of the 15th European Conference on Artificial Intelligence.* IOS Press, pages 390–394, 2002.

[3] J. Barzilai, and J. M. Borwein. Two-point step size gradient methods, *IMA J. Numer. Anal.*, 8:141–148, 1988.

[4] H. H. Bauschke, P. L. Combettes, and D. R. Luke. Phase retrieval, error reduction algorithm, and Fienup variants: A view from convex optimization, *J. Opt. Soc. Am. A*, 19:1334–1345, 2002.

[5] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model, *J. Mach. Learn.*, 3:1137–1155, 2003.

[6] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University, Clarendon Press, New York, 1995.

[7] O. Booij, and H. T. Nguyen. A gradient descent rule for spiking neurons emitting multiple spikes, *Inf. Process. Lett.*, 95:552–558, 2005.

[8] Y. Chen, Q. Gao, Y. Wei, and Y. Wang. Study on fractional order gradient methods, *Appl. Math. Comput.*, 314:310–321, 2017.

[9] M. Dalir, M. Bashour. Applications of fractional calculus, *Appl. Math. Sci.*, 4:1021–1032, 2010.

[10] J. R. Fienup. Phase retrieval algorithms: A comparison, *Appl. Opt.*, 21:2758–2769, 1982.

[11] R. Fletcher. *Practical Methods of Optimization.* John Wiley and Sons, New York, 1987.

[12] F. Guely, and P. Siarry. Gradient descent method for optimizing various fuzzy rule bases. In *Proc. 2nd IEEE lnternat. Conf. on Fuzzy System.* Proceedings, volume 2, pages 1241-1246, 1993. DOI: 10.1109/FUZZY.1993.327570.

[13] A. A. Kilbas, H. M. Srivastava, and J. J. Trujillo. *Theory and Applications of Fractional Differential Equations.* Elsevier, Amsterdam, 2006.

[14] T. I. Kuznetsova. On the phase retrieval problem in optics, *Sov. Phys. Usp.*, 31:364, 1988. DOI: 10.1070/PU1988v031n04ABEH005755.

[15] E. T. Lee. A computer program for linear logistic regression analysis, *Comput. Programs Biomed.*, 4:80–92, 1974.

[16] P. Netrapalli, P. Jain, and S. Sanghavi. Phase retrieval using alternating minimization, *IEEE Trans. Signal Process.*, 63:4814–4826, 2015.

[17] K. B. Oldham, and J. Spanier. *The Fractional Calculus.* Academic Press, New York, 1974.

[18] G. Y. Park, and P. H. Seong. Towards increasing the learning speed of gradient descent method in fuzzy system, *Fuzzy Sets Syst.*, 77:229–313, 1996.

[19] N. Qian. On the momentum term in gradient descent learning algorithms, *Neural Netw.*, 12:145–151, 1999.

[20] Y. Song, Q. Cai, F. Nie, and C. Zhang. Semi-Supervised Additive Logistic Regression: A Gradient Descent Solution, *Tsinghua Sci. Technol.*, 12:638–646, 2007.

[21] M. N. Vrahatis, G. S. Androulakis, J. N. Lambrinos, and G. D. Magoulas. A class of gradient unconstrained minimization algorithms with adaptive stepsize, *J. Comp. and Appl. Math.*, 114:367–386, 2000.

[22] J. Wang, Y. Wen, Y. Gou, Z. Ye, and H. Chen. Fractional-order gradient descent learning of BP neural networks with Caputo derivative, *Neural Netw.*, 89:19–30, 2017.

[23] Y. Xu, X. Zeng, L. Han, and J. Yang. A supervised multi-spike learning algorithm based on gradient descent for spiking neural networks, *Neural Netw.*, 43:99–113, 2013.