SCORE: Approximating Curvature Information under Self-Concordant Regularization

Adeyemi D. Adeoye[1] and Alberto Bemporad[1]

[1] IMT School for Advanced Studies Lucca, Italy
Corresponding E-mail: adeyemi.adeoye@imtlucca.it
alberto.bemporad@imtlucca.it

Abstract— **In this work, we propose the SCORE (self-concordant regularization) framework for (overparameterized) unconstrained minimization problems by incorporating second-order information in the *Newton decrement* framework for convex optimization. We propose the generalized Gauss-Newton with Self-Concordant Regularization (GGN-SCORE) algorithm that updates the network parameters each time it receives a new input batch. The proposed algorithm exploits the structure of the second-order information in the Hessian matrix, thereby reducing the training computational overhead. Numerical experiments show the efficiency of our method and its fast convergence, which compare favorably against baseline first-order methods and a quasi-Newton method.**

Keywords— **Self-concordant functions, Overparamterized models, Optimization, Second-order methods, Quasi-Newton methods.**

I. Introduction

The results presented in this work apply to a pseudo-online optimization algorithm based on solving a regularized unconstrained minimization problem for machine learning. We propose a new self-concordant regularization (SCORE) scheme for efficiently choosing optimal variables of the model involving smooth optimization objectives, where one of the objective functions regularizes the model's variable vector and hence avoid overfitting, ultimately improving the model's ability to generalize well.

Unlike first-order methods such as stochastic gradient descent (SGD) [1, 2] and its variants [3, 4, 5, 6] that only make use of first-order information through the function gradients, second-order methods [7, 8, 9, 10, 11, 12, 13] attempt to incorporate, in some way, second-order information in their approach, through the Hessian matrix or the Fisher information matrix (FIM). It is well known that this generally provides second-order methods with better (quadratic) convergence than a typical first-order method which only converges linearly in the neighbourhood of the solution [14].

Despite their convergence advantage over first-order methods, second-order methods result into highly prohibitive computations. By an *extra* assumption that the regularization function is self-concordant, we propose GGN-SCORE algorithm (see Algorithm 1) that updates, while reducing computational overhead, the minimization variables in the framework of the *local norm* $\|\cdot\|_x$ (a.k.a., Newton decrement) of a self-concordant function $f(x)$ such as seen in [15]. Our proposed scheme does not require that the output-fit loss function is self-concordant, which in many applications does not hold [16]. Instead, we exploit the *greedy* descent provision of self-concordant functions, via regularization, to achieve a fast convergence rate while maintaining feasible assumptions on the combined objective function (from an application point of view). The experimental results provide an interesting opportunity for future investigation and scaling of the proposed method for large-scale machine learning problems, as the experiments involves training an overparameterized neural network: Overparameterization is an interesting and desirable property, and a topic of concern for machine learning [17, 18, 19, 20].

II. Problem Statement

Let $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$ be a sequence of input and output sample pairs, $\boldsymbol{x}_n \in \mathbb{R}^{n_p}, \boldsymbol{y}_n \in \mathbb{R}^d$, where n_p is the number of features and d is the number of targets. We assume a model $f(\boldsymbol{\theta}; \boldsymbol{x}_n)$, defined by $f\colon \mathbb{R}^{n_w} \times \mathbb{R}^{n_p} \to \mathbb{Y}$ and parameterized by the vector of variables $\boldsymbol{\theta} \in \mathbb{R}^{n_w}$. Suppose that $f(\boldsymbol{\theta}; \boldsymbol{x}_n)$ outputs the value $\hat{\boldsymbol{y}}_n \in \mathbb{R}^d$, the regularized minimization problem we want to solve is

$$\min_{\boldsymbol{\theta}} \mathscr{L}(\boldsymbol{\theta}) := \underbrace{\sum_{n=1}^{N} \ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)}_{g(\boldsymbol{\theta})} + \lambda \underbrace{\sum_{j=1}^{n_w} r_j(\boldsymbol{\theta}_j)}_{h(\boldsymbol{\theta})}, \tag{1}$$

where $\ell : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a twice-differentiable output-fit loss function, $r_j : \mathbb{R} \to \mathbb{R}$, $j = 1, \ldots, n_w$, define a regularization term on $\boldsymbol{\theta}$, $g(\boldsymbol{\theta})\colon \mathbb{R}^{n_w} \to \mathbb{R}$, $h(\boldsymbol{\theta})\colon \mathbb{R}^{n_w} \to \mathbb{R}$. We assume

that the regularization function $h(\boldsymbol{\theta})$, scaled by the parameter $\lambda > 0$, is M_h-self-concordant.

III. Algorithm

At each oracle call (Algorithm 1), we define

$$\boldsymbol{g}(\boldsymbol{\theta}_k) = \partial_{\boldsymbol{\theta}} \mathscr{L}(\boldsymbol{\theta}_k) = \boldsymbol{J}^T \boldsymbol{e} \tag{2}$$

$$\boldsymbol{H} \approx \boldsymbol{J}^T \boldsymbol{Q} \boldsymbol{J} + \lambda \boldsymbol{H}_h, \tag{3}$$

where $\boldsymbol{Q} \in \mathbb{R}^{M \times M}$ and $\boldsymbol{H}_h \in \mathbb{R}^{n_w \times n_w}$ are diagonal matrices with the second derivatives of the output-fit loss with respect to the outputs (augmented with a vector of zeros) and the second derivatives of h with respect to $\boldsymbol{\theta}$, respectively, as their diagonal entries. $M = dm + 1$, m is some mini-batch size, $\boldsymbol{J} \in \mathbb{R}^{M \times n_w}$ is an augmented Jacobian of the objective functions with respect to $\boldsymbol{\theta}$, $\boldsymbol{e} \in \mathbb{R}^M$ is the Jacobian of the output-fit loss function augmented with a vector of ones.

Algorithm 1 GGN-SCORE

1: **Input:** variables vector $\boldsymbol{\theta}_k$, data $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^m$, $\boldsymbol{H}_h$, $\boldsymbol{Q}$, $\boldsymbol{J}$, $\boldsymbol{e}$, parameters $\alpha_k > 0, M_h, \lambda$
2: **Output:** variables vector $\boldsymbol{\theta}_{k+1}$
3: Compute $\boldsymbol{g}_h = \partial_{\boldsymbol{\theta}_k} h(\boldsymbol{\theta}_k)$
4: Choose $\eta_k = \left\langle \boldsymbol{g}_h, \boldsymbol{H}_h^{-1} \boldsymbol{g}_h \right\rangle^{1/2}$
5: Set $\rho_k = \frac{\alpha_k}{1 + M_h \eta_k}$
6: Set $\boldsymbol{G} = \boldsymbol{H}_h^{-1} \boldsymbol{J}^T \left(\lambda \boldsymbol{I} + \boldsymbol{Q} \boldsymbol{J} \boldsymbol{H}_h^{-1} \boldsymbol{J}^T \right)^{-1} \boldsymbol{e}$
7: Compute $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \rho_k \boldsymbol{G}$

IV. Experiments

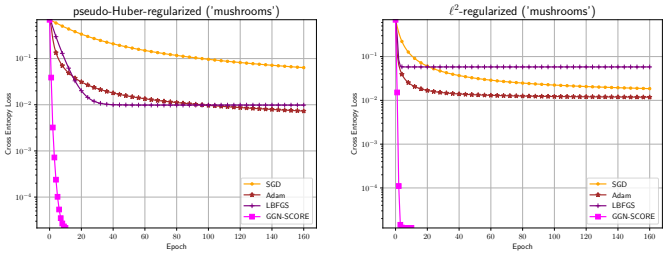

Fig. 1: Convergence curves for the non-convex neural network training

Figure 1 compares convergence results between our algorithm and three other literature algorithms, with $\ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n) = \frac{1}{2} \sum_{n=1}^N \boldsymbol{y}_n \log\left(\frac{1}{\hat{\boldsymbol{y}}_n}\right) + (\boldsymbol{1} - \boldsymbol{y}_n) \log\left(\frac{1}{\boldsymbol{1} - \hat{\boldsymbol{y}}_n}\right)$ and the pseudo-Huber regularizer $h_\mu(\boldsymbol{\theta})$ (left), $h_\mu(\boldsymbol{\theta}) := \sqrt{\mu^2 + \boldsymbol{\theta}^2} - \mu$, and the 2-norm regularizer $h_2(\boldsymbol{\theta})$ (right), $h_2(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2 := \sqrt{\sum_{i=1}^{n_w} |\theta_i|^2}$, using the `mushrooms` dataset from LIBSVM repository [21]. These results show the superiority of our algorithm to the other algorithms under the given assumptions.

References

1. Robbins Herbert, Monro Sutton. A stochastic approximation method *The annals of mathematical statistics.* 1951:400–407.
2. Bottou Léon. Large-scale machine learning with stochastic gradient descent in *Proceedings of COMPSTAT'2010*:177–186Springer 2010.
3. Duchi John, Hazan Elad, Singer Yoram. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research.* 2011;12.
4. Zeiler Matthew D. Adadelta: an adaptive learning rate method *arXiv preprint arXiv:1212.5701.* 2012.
5. Kingma Diederik P, Ba Jimmy. Adam: A method for stochastic optimization *arXiv preprint arXiv:1412.6980.* 2014.
6. Johnson Rie, Zhang Tong. Accelerating stochastic gradient descent using predictive variance reduction *Advances in neural information processing systems.* 2013;26:315–323.
7. Becker Sue, Cun Yann. Improving the Convergence of Back-Propagation Learning with Second Order Methods 1988.
8. Liu Dong C, Nocedal Jorge. On the limited memory BFGS method for large scale optimization *Mathematical programming.* 1989;45:503–528.
9. Hagan Martin T, Menhaj Mohammad B. Training feedforward networks with the Marquardt algorithm *IEEE transactions on Neural Networks.* 1994;5:989–993.
10. Amari Shun-Ichi. Natural gradient works efficiently in learning *Neural computation.* 1998;10:251–276.
11. Martens James, others . Deep learning via hessian-free optimization. in *ICML*;27:735–742 2010.
12. Pascanu Razvan, Bengio Yoshua. Revisiting natural gradient for deep networks *arXiv preprint arXiv:1301.3584.* 2013.
13. Martens James, Grosse Roger. Optimizing neural networks with kronecker-factored approximate curvature in *International conference on machine learning*:2408–2417PMLR 2015.
14. Nesterov Yurii, others . *Lectures on convex optimization*;137. Springer 2018.
15. Nesterov Yurii, Nemirovskii Arkadii. *Interior-point polynomial algorithms in convex programming.* SIAM 1994.
16. Mishchenko Konstantin. Regularized Newton Method with Global $O(1/k^2)$ Convergence *arXiv preprint arXiv:2112.02089.* 2021.
17. Bubeck Sébastien, Sellke Mark. A Universal Law of Robustness via Isoperimetry *arXiv preprint arXiv:2105.12806.* 2021.
18. Muthukumar Vidya, Vodrahalli Kailas, Subramanian Vignesh, Sahai Anant. Harmless interpolation of noisy data in regression *IEEE Journal on Selected Areas in Information Theory.* 2020;1:67–83.
19. Belkin Mikhail, Hsu Daniel, Ma Siyuan, Mandal Soumik. Reconciling modern machine-learning practice and the classical bias–variance trade-off *Proceedings of the National Academy of Sciences.* 2019;116:15849–15854.
20. Allen-Zhu Zeyuan, Li Yuanzhi, Liang Yingyu. Learning and generalization in overparameterized neural networks, going beyond two layers *Advances in neural information processing systems.* 2019;32.
21. Chang Chih-Chung, Lin Chih-Jen. LIBSVM: a library for support vector machines *ACM transactions on intelligent systems and technology (TIST).* 2011;2:1–27.