Computational Optimization and Applications
https://doi.org/10.1007/s10589-023-00502-2

SCORE: approximating curvature information under self-concordant regularization

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

Received: 21 April 2022 / Accepted: 17 June 2023

Abstract

Optimization problems that include regularization functions in their objectives are regularly solved in many applications. When one seeks second-order methods for such problems, it may be desirable to exploit specific properties of some of these regularization functions when accounting for curvature information in the solution steps to speed up convergence. In this paper, we propose the SCORE (self-concordant regularization) framework for unconstrained minimization problems which incorporates 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 minimization variables 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 computational overhead. GGN-SCORE demonstrates how to speed up convergence while also improving model generalization for problems that involve regularized minimization under the proposed SCORE framework. Numerical experiments show the efficiency of our method and its fast convergence, which compare favorably against baseline first-order and quasi-Newton methods. Additional experiments involving non-convex (overparameterized) neural network training problems show that the proposed method is promising for non-convex optimization.

Keywords Self-concordant functions · Gauss–Newton methods · Convex optimization · Overparameterized models

✉ Adeyemi D. Adeoye
adeyemi.adeoye@imtlucca.it

Alberto Bemporad
alberto.bemporad@imtlucca.it

[1] IMT School for Advanced Studies, Piazza San Francesco 19, Lucca 55100, Italy

Published online: 08 July 2023

1 Introduction

The results presented in this paper apply to a pseudo-online optimization algorithm based on solving a regularized unconstrained minimization problem under the assumption of strong convexity and Lipschitz continuity. Unlike first-order methods such as stochastic gradient descent (SGD) [1, 2] and its variants [3–6] that only make use of first-order information through the function gradients, second-order methods [7–13] 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, namely, inverting an $n_w \times n_w$ matrix at each iteration, where n_w is the number of optimization variables. In most commonly used second-order methods, the natural gradient, Gauss–Newton, and the sub-sampled Newton [15]—or its regularized version [16]—used for incorporating second-order information while maintaining desirable convergence properties, compute the Hessian matrix (or FIM) $\boldsymbol{H}$ by using the approximation $\boldsymbol{H} \approx \boldsymbol{J}^T \boldsymbol{J}$, where $\boldsymbol{J}$ is the Jacobian matrix. This approximation is still an $n_w \times n_w$ matrix and remains computationally demanding for large problems. In recent works [17–20], second-order methods for overparameterized neural network models are made to bypass this difficulty by applying a matrix identity, and instead only compute the matrix $\boldsymbol{J}\boldsymbol{J}^T$ which is a $d \cdot N \times d \cdot N$ matrix, where d is the model output dimension and N is the number of data points. This approach significantly reduces computational overhead in the case $d \cdot N$ is much smaller than n_w (overparameterized models) and helps to accelerate convergence [17]. Nevertheless, for an objective function with a differentiable (up to two times) convex regularizer, this simplification requires a closer attention and special modifications for a general problem with a large number of variables.

The idea of exploiting desirable regularization properties for improving the convergence of the (Gauss–)Newton scheme has been around for decades and most of the published works on the topic combine in different ways the idea of Levenberg–Marquardt (LM) regularization, line-search, and trust-region methods [21]. For example, the recent work [22] combines the idea of cubic regularization (originally proposed by [21]) and a particular variant of the adaptive LM penalty that uses the *Euclidean* gradient norm of the output-fit loss function (see [23] for a comprehensive list and complexity bounds). Their proposed scheme achieves a global $\mathcal{O}(k^{-2})$ rate, where k is the iteration number. A similar idea is considered in [24] using the Bregman distances, extending the idea to develop an accelerated variant of the scheme that achieves a $\mathcal{O}(k^{-3})$ convergence rate.

In this paper, we propose a new self-concordant regularization (SCORE) scheme for efficiently choosing optimal variables of the model involving smooth, strongly convex optimization objectives, where one of the objective functions regularizes the model's variable vector and hence avoids overfitting, ultimately improving the model's ability to generalize well. By an *extra* assumption that the regularization function is self-concordant, we propose the GGN-SCORE algorithm (see Algorithm 4 below) that updates 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 [25]. Our proposed scheme does not require that the output-fit loss function is self-concordant, which in many applications does not hold [22]. 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). Although this paper assumes a convex optimization problem, we also provide experiments that show promising results for non-convex problems that arise when training neural networks. Our experimental results provide an interesting opportunity for future investigation and scaling of the proposed method for large-scale machine learning problems, as one of the non-convex problems considered in the experiments involves training an overparameterized neural network. We remark that overparameterization is an interesting and desirable property, and a topic of interest in the machine learning community [26–29].

This paper is organized as follows: First, we introduce some notations, formulate the optimization problem with basic assumptions, and present an initial motivation for the optimization method in Sect. 2. In Sect. 3, we derive a new generalized method for reducing the computational overhead associated with the mini-batch Newton-type updates. The idea of SCORE is introduced in Sect. 4 and our GGN-SCORE algorithm is presented thereafter. Experimental results that show the efficiency and fast convergence of the proposed method are presented in Sect. 5.

2 Preliminaries

2.1 Notation and basic assumptions

Let $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$ be a sequence of N 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 $\boldsymbol{\theta} \in \mathbb{R}^{n_w}$. We denote by $\partial_{\boldsymbol{a}}\boldsymbol{b} \equiv \partial\boldsymbol{b}$ the gradient (or first derivative) of $\boldsymbol{b}$ (with respect to $\boldsymbol{a}$) and $\partial^2_{\boldsymbol{aa}}\boldsymbol{b} \equiv \partial^2\boldsymbol{b}$ the second derivative of $\boldsymbol{b}$ with respect to $\boldsymbol{a}$. We write $\|\cdot\|$ to denote the 2-norm. The set $\{\operatorname{diag}(\boldsymbol{v})\colon \boldsymbol{v} \in \mathbb{R}^n\}$, where $\operatorname{diag}\colon \mathbb{R}^n \to \mathbb{R}^{n\times n}$, denotes the set of all diagonal matrices in $\mathbb{R}^{n\times n}$. Throughout the paper, bold-face letters denote vectors and matrices.

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

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

where $\ell : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a (strongly) convex twice-differentiable output-fit loss function, $r_j : \mathbb{R} \to \mathbb{R}$, $j = 1, \ldots, n_w$, define a separable 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 twice differentiable and strongly convex. The following preliminary conditions define the regularity of the Hessian of $\mathcal{L}(\boldsymbol{\theta})$ and are assumed to hold only locally in this work:

Assumption 1 The functions g and h are twice-differentiable with respect to $\boldsymbol{\theta}$ and are respectively γ_l- and γ_a-strongly convex.

Assumption 2 $\exists \gamma_u, \gamma_b$ with $0 \leq \gamma_l \leq \gamma_u < \infty$, $0 < \gamma_a \leq \gamma_b < \infty$, such that the gradient of $\mathcal{L}(\boldsymbol{\theta})$ is $(\gamma_u + \lambda\gamma_b)$-Lipschitz continuous $\forall \boldsymbol{x} \in \mathbb{R}^{n_p}, \boldsymbol{y} \in \mathbb{R}^d$. That is, $\forall \boldsymbol{x} \in \mathbb{R}^{n_p}, \forall \boldsymbol{y} \in \mathbb{R}^d$, the gradient $\partial_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}) = \partial_{\boldsymbol{\theta}} g(\boldsymbol{\theta}) + \lambda \partial_{\boldsymbol{\theta}} h(\boldsymbol{\theta})$ satisfies

$$\|\partial_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{y}, f(\boldsymbol{\theta_1}; \boldsymbol{x})) - \partial_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{y}, f(\boldsymbol{\theta_2}; \boldsymbol{x}))\| \leq (\gamma_u + \lambda\gamma_b)\, \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|\,, \tag{2}$$

for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \mathbb{R}^{n_w}$.

Assumption 3 $\exists \gamma_g, \gamma_h$ with $0 < \gamma_g, \gamma_h < \infty$ such that $\forall \boldsymbol{x} \in \mathbb{R}^{n_p}, \forall \boldsymbol{y} \in \mathbb{R}^d$, the second derivatives of $g(\boldsymbol{\theta})$ and $h(\boldsymbol{\theta})$ respectively satisfy

$$\left\|\partial^2 g(\boldsymbol{y}, f(\boldsymbol{\theta_1}; \boldsymbol{x})) - \partial^2 g(\boldsymbol{y}, f(\boldsymbol{\theta_2}; \boldsymbol{x}))\right\| \leq \gamma_g\, \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|\,, \tag{3a}$$

$$\left\|\partial^2 h(\boldsymbol{y}, f(\boldsymbol{\theta_1}; \boldsymbol{x})) - \partial^2 h(\boldsymbol{y}, f(\boldsymbol{\theta_2}; \boldsymbol{x}))\right\| \leq \gamma_h\, \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|\,, \tag{3b}$$

for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \mathbb{R}^{n_w}$.

Commonly used loss functions such as the squared loss, and the sigmoidal cross-entropy loss are twice differentiable and (strongly) convex in the model variables. Certain smoothed sparsity-inducing penalties such as the (pseudo-)Huber function – presented later in this paper – constitute the class of functions that may be well-suited for $h(\boldsymbol{\theta})$ defined above.

The assumptions of strong convexity and smoothness about the objective $\mathcal{L}(\boldsymbol{\theta})$ are standard conventions in many optimization problems as they help to characterize the convergence properties of the underlying solution method [30, 31]. However, the smoothness assumption about the objective $\mathcal{L}(\boldsymbol{\theta})$ is sometimes not feasible for some multi-objective (or regularized) problems where a non-smooth (penalty-inducing) function $h(\boldsymbol{\theta})$ is used (see the recent work [32]). In such a case, and when the need to incorporate second-order information arise, a well-known approach in the optimization literature is generally either to approximate the non-smooth objectives by a smooth quadratic function (when such an approximation is available) or use a "proximal splitting" method and replace the 2-norm in this setting with the Q-norm, where Q is the Hessian matrix or its approximation [33]. In [33], the authors propose two techniques that help to avoid the complexity that is often introduced in subproblems when the latter approach is used. While proposing new approaches, [34] highlights some popular techniques to handle non-differentiability. Each of these works highlight the importance of incorporating second-order information in the solution techniques of optimization problems. By conveniently solving the optimization problem (1) where the assumptions made above are satisfied, our method ensures the full curvature information is captured while reducing computational overhead.

2.2 Approximate Newton scheme

Given the current value of $\boldsymbol{\theta}$, the (Gauss–)Newton method computes an update to $\boldsymbol{\theta}$ via

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \rho \boldsymbol{G}, \tag{4}$$

where ρ is the step size, $\boldsymbol{G} = \boldsymbol{H}^{-1} \partial \mathcal{L}(\boldsymbol{\theta})$ and $\boldsymbol{H}$ is the Hessian of $\mathcal{L}$ or its approximation. In this work, we consider the generalized Gauss–Newton (GGN) approximation of $\boldsymbol{H}$ which we now define in terms of the function $g(\boldsymbol{\theta})$. This approximation and its detailed expression motivates the modified version introduced in the next section to include the regularization function $h(\boldsymbol{\theta})$.

Definition 1 (Generalized Gauss–Newton Hessian) Let $\operatorname{diag}(\boldsymbol{q}_n) \in \mathbb{R}^{d \times d}$ be the second derivative of the loss function $\ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)$ with respect to the predictor $\hat{\boldsymbol{y}}_n$, $\boldsymbol{q}_n = \partial^2_{\hat{\boldsymbol{y}}_n \hat{\boldsymbol{y}}_n} \ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)$ for $n = 1, 2, \ldots, N$, and let $\boldsymbol{Q}_g \in \mathbb{R}^{dN \times dN}$ be a block diagonal matrix with $\boldsymbol{q}_n$ being the n-th diagonal block. Let $\boldsymbol{J}_n \in \mathbb{R}^{d \times n_w}$ denote the Jacobian of $\hat{\boldsymbol{y}}_n$ with respect to $\boldsymbol{\theta}$ for $n = 1, 2, \ldots, N$, and let $\boldsymbol{J}_g \in \mathbb{R}^{dN \times n_w}$ be the vertical concatenation of all $\boldsymbol{J}_n$'s. Then, the generalized Gauss–Newton (GGN) approximation of the Hessian matrix $\boldsymbol{H}_g \in \mathbb{R}^{n_w \times n_w}$ associated with the fit loss $\ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)$ with respect to $\boldsymbol{\theta}$ is defined by

$$\boldsymbol{H}_g \approx \boldsymbol{J}_g^T \boldsymbol{Q}_g \boldsymbol{J}_g = \sum_{n=1}^{N} \boldsymbol{J}_n^T \operatorname{diag}(\boldsymbol{q}_n) \boldsymbol{J}_n. \tag{5}$$

Let $\boldsymbol{e}_n \in \mathbb{R}^d$ be the Jacobian of the fit loss defined by $\boldsymbol{e}_n = \partial_{\hat{\boldsymbol{y}}_n} \ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)$ for $n = 1, 2, \ldots, N$. For example, in case of squared loss $\ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n) = \frac{1}{2}(\boldsymbol{y}_n - \hat{\boldsymbol{y}}_n)^2$ we get that $\boldsymbol{e}_n$ is the residual $\boldsymbol{e}_n = \hat{\boldsymbol{y}}_n - \boldsymbol{y}_n$. Let $\boldsymbol{e}_g \in \mathbb{R}^{dN}$ be the vertical concatenation of all $\boldsymbol{e}_n$'s. Then, using the chain rule, we write

$$\begin{aligned}
\boldsymbol{J}_n^T \boldsymbol{e}_n^T &= \begin{bmatrix} \partial_{\theta_1} \hat{y}^{(1)} & \partial_{\theta_1} \hat{y}^{(2))} & \cdots & \partial_{\theta_1} \hat{y}^{(d)} \\ \partial_{\theta_2} \hat{y}^{(1)} & \partial_{\theta_2} \hat{y}^{(2)} & \cdots & \partial_{\theta_2} \hat{y}^{(d)} \\ \vdots & \vdots & & \vdots \\ \partial_{\theta_{n_w}} \hat{y}^{(1)} & \partial_{\theta_{n_w}} \hat{y}^{(2)} & \cdots & \partial_{\theta_{n_w}} \hat{y}^{(d)} \end{bmatrix} \begin{bmatrix} \partial_{\hat{y}} \ell^{(1)} \\ \partial_{\hat{y}} \ell^{(2)} \\ \vdots \\ \partial_{\hat{y}} \ell^{(d)} \end{bmatrix} \\
&= \begin{bmatrix} \partial_{\theta_1} \ell^{(1)} + \partial_{\theta_1} \ell^{(2)} + \cdots + \partial_{\theta_1} \ell^{(d)} \\ \partial_{\theta_2} \ell^{(1)} + \partial_{\theta_2} \ell^{(2)} + \cdots + \partial_{\theta_2} \ell^{(d)} \\ \vdots \\ \partial_{\theta_{n_w}} \ell^{(1)} + \partial_{\theta_{n_w}} \ell^{(2)} + \cdots + \partial_{\theta_{n_w}} \ell^{(d)} \end{bmatrix} \\
&= \begin{bmatrix} \sum_{i=1}^{d} \partial_{\theta_1} \ell^{(i)} & \sum_{i=1}^{d} \partial_{\theta_2} \ell^{(i)} & \cdots & \sum_{i=1}^{d} \partial_{\theta_{n_w}} \ell^{(i)} \end{bmatrix}^T ,
\end{aligned} \tag{6}$$

and

$$\boldsymbol{g}_g(\boldsymbol{\theta}) = \boldsymbol{J}_g^T \boldsymbol{e}_g = \sum_{n=1}^{N} \boldsymbol{J}_n^T \boldsymbol{e}_n. \tag{7}$$

As noted in [35], the GGN approximation has the advantage of capturing the curvature information of ℓ in $g(\boldsymbol{\theta})$ through the term $\boldsymbol{Q}_g$ as opposed to the FIM, for example, which ignores the full second-order interactions. While it may become obvious, say when training a deep neural network with certain loss functions, how the GGN approximation can be exploited to simplify expressions for $\boldsymbol{H}_g$ (see e.g. in [36]), a modification is required to take account of a twice-differentiable convex regularization function to achieve some degree of simplicity and elegance. We derive a modification to the above for the mini-batch scheme presented in the next section that includes the derivatives of $h(\boldsymbol{\theta})$ in the GGN approximation of $\boldsymbol{H}_g$. This modification leads to our GGN-SCORE algorithm in Sect. 4.

3 Second-order (pseudo-online) optimization

Suppose that at each mini-batch step k we uniformly select a random index set $\mathcal{I}_k \subseteq \{1, 2, \ldots, N\}$, $|\mathcal{I}_k| = m \leq N$ (usually $m \ll N$) to access a mini-batch of m samples from the training set. The loss derivatives used for the recursive update of the variables $\boldsymbol{\theta}$ in this way is computed at each step k, and are estimated as running averages over the batch-wise computations. This leads to a stochastic approximation of the true derivatives at each iteration for which we assume unbiased estimations.

The problem of finding the optimal adjustment $\delta\boldsymbol{\theta}_{mk} := \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k$ that solves (1) results in solving either an *overdetermined* or an *underdetermined* linear system depending on whether $dm \geq n_w$ or $dm < n_w$, respectively. Consider, for example, the squared fit loss and the penalty-inducing square norm as the scalar-valued functions $g(\boldsymbol{\theta})$ and $h(\boldsymbol{\theta})$, respectively in (1). Then, $\boldsymbol{Q}_g$ will be the identity matrix, and the LM solution $\delta\boldsymbol{\theta}$ [37, 38] is estimated at each iteration k according to the rule[1]:

$$\delta\boldsymbol{\theta} = -(\boldsymbol{H}_g + \lambda \boldsymbol{I})^{-1} \boldsymbol{g}_g = -(\boldsymbol{J}_g^T \boldsymbol{J}_g + \lambda \boldsymbol{I})^{-1} \boldsymbol{J}_g^T \boldsymbol{e}_g. \tag{8}$$

If $dm < n_w$ (possibly $dm \ll n_w$), then by using the Searle identity $(\boldsymbol{A}\boldsymbol{B} + \lambda \boldsymbol{I})\boldsymbol{A} = \boldsymbol{A}(\boldsymbol{B}\boldsymbol{A} + \lambda \boldsymbol{I})$ [39], we can conveniently update the adjustment $\delta\boldsymbol{\theta}$ by

$$\delta\boldsymbol{\theta} = -\boldsymbol{J}_g^T (\boldsymbol{J}_g \boldsymbol{J}_g^T + \lambda \boldsymbol{I})^{-1} \boldsymbol{e}_g. \tag{9}$$

Clearly, this provides a more computationally efficient way of solving for $\delta\boldsymbol{\theta}$. In what follows, we formulate a generalized solution method for the regularized problem (1)

[1] For simplicity of notation, and unless where the full notations are explicitly required, we shall subsequently drop the subscripts m and k, and assume that each expression represents stochastic approximations performed at step k using randomly selected data batches each of size m.

which similarly exploits the Hessian matrix structure when solving the given optimization problem, thereby conveniently and efficiently computing the adjustment $\delta\boldsymbol{\theta}$.

Taking the second-order approximation of $\mathcal{L}(\boldsymbol{\theta})$, we have

$$\mathcal{L}(\boldsymbol{\theta}+\delta\boldsymbol{\theta}) \approx \mathcal{L}(\boldsymbol{\theta}) + \boldsymbol{g}^T\delta\boldsymbol{\theta} + \frac{1}{2}\delta\boldsymbol{\theta}^T\boldsymbol{H}\delta\boldsymbol{\theta}, \tag{10}$$

where $\boldsymbol{H} \in \mathbb{R}^{n_w\times n_w}$ is the Hessian of $\mathcal{L}(\boldsymbol{\theta})$ and $\boldsymbol{g} \in \mathbb{R}^{n_w}$ is its gradient. Let $M = dm+1$ and define the Jacobian $\boldsymbol{J} \in \mathbb{R}^{M\times n_w}$:

$$\boldsymbol{J}^T = \begin{bmatrix} \partial_{\theta_1}\hat{\boldsymbol{y}}_1 & \partial_{\theta_1}\hat{\boldsymbol{y}}_2 & \cdots & \partial_{\theta_1}\hat{\boldsymbol{y}}_m & \lambda\partial_{\theta_1}r_1 \\ \partial_{\theta_2}\hat{\boldsymbol{y}}_1 & \partial_{\theta_2}\hat{\boldsymbol{y}}_2 & \cdots & \partial_{\theta_2}\hat{\boldsymbol{y}}_m & \lambda\partial_{\theta_2}r_2 \\ \vdots & \vdots & & \vdots & \vdots \\ \partial_{\theta_{n_w}}\hat{\boldsymbol{y}}_1 & \partial_{\theta_{n_w}}\hat{\boldsymbol{y}}_2 & \cdots & \partial_{\theta_{n_w}}\hat{\boldsymbol{y}}_m & \lambda\partial_{\theta_{n_w}}r_{n_w} \end{bmatrix}. \tag{11}$$

Let $\boldsymbol{e}_M = \boldsymbol{1}$ and denote by $\boldsymbol{e} \in \mathbb{R}^M$ the vertical concatenation of all $\boldsymbol{e}_n$'s, $\boldsymbol{e}_n \in \mathbb{R}^d$, $n = 1, 2, \ldots, m+1$. Then by using the chain rule as in (6) and (7), we obtain

$$\boldsymbol{g}(\boldsymbol{\theta}) = \partial_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}) = \boldsymbol{J}^T\boldsymbol{e}. \tag{12}$$

Let $\boldsymbol{q}_n = \partial^2_{\hat{\boldsymbol{y}}_n\hat{\boldsymbol{y}}_n}\ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)$, $\boldsymbol{q}_n \in \mathbb{R}^d$ for $n = 1, 2, \ldots, m$ (clearly $\boldsymbol{q}_n = \boldsymbol{1}$ in case of squared fit loss terms) and let $\boldsymbol{q}_{m+1} = \boldsymbol{0}$. Define $\boldsymbol{Q} \in \mathbb{R}^{M\times M}$ as the diagonal matrix with diagonal elements $\boldsymbol{q}_n$ for $n = 1, 2, \ldots, m+1$, where $\boldsymbol{Q} = \boldsymbol{Q}^T \succeq \boldsymbol{0}$ by convexity of ℓ. Consider the following slightly modified GGN approximation of the Hessian $\boldsymbol{H} \in \mathbb{R}^{n_w\times n_w}$ associated with $\mathcal{L}(\boldsymbol{\theta})$:

$$\boldsymbol{H} \approx \boldsymbol{J}^T\boldsymbol{Q}\boldsymbol{J} + \lambda\boldsymbol{H}_h, \tag{13}$$

where $\boldsymbol{H}_h$ is the Hessian of the regularization term $h(\boldsymbol{\theta})$, $\boldsymbol{H}_h \in \mathbb{R}^{n_w\times n_w}$, and is a diagonal matrix whose diagonal terms are

$$\boldsymbol{H}_{h\,jj} = \frac{d^2r_j(\theta_j)}{d\theta_j^2},\ j = 1, \ldots, n_w.$$

We hold on to the notation $\boldsymbol{H}$ to represent the modified GGN approximation of the full Hessian matrix $\partial^2\mathcal{L}$. By differentiating (10) with respect to $\delta\boldsymbol{\theta}$ and equating to zero, we obtain the optimal adjustment

$$\delta\boldsymbol{\theta} = -(\boldsymbol{J}^T\boldsymbol{Q}\boldsymbol{J} + \lambda\boldsymbol{H}_h)^{-1}\boldsymbol{J}^T\boldsymbol{e}. \tag{14}$$

Remark 1 The inverse matrix in (14) exists due to the strong convexity assumption on the regularization function h which makes $\boldsymbol{H}_{\boldsymbol{r}} \succeq (\min_j \boldsymbol{H}_{h\,jj})\boldsymbol{I}$ and therefore matrix $\boldsymbol{J}^T\boldsymbol{Q}\boldsymbol{J} + \lambda\boldsymbol{H}_h$ is symmetric positive definite and hence invertible.

Let $\boldsymbol{U} = \boldsymbol{J}^T\boldsymbol{Q}$. Using the identity [40, 41]

$$(\boldsymbol{D} - \boldsymbol{V}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{V}\boldsymbol{A}^{-1} = \boldsymbol{D}^{-1}\boldsymbol{V}(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{V})^{-1} \tag{15}$$

with $\boldsymbol{D} = \lambda\boldsymbol{H_r}$, $\boldsymbol{V} = -\boldsymbol{J}^T$, $\boldsymbol{A} = \boldsymbol{I}_M$, and $\boldsymbol{B} = \boldsymbol{U}^T$, and recalling that $\boldsymbol{Q}$ is symmetric, from (14) we get

$$\begin{aligned}\delta\boldsymbol{\theta} &= \left(\lambda\boldsymbol{H}_h - (-\boldsymbol{J}^T)\boldsymbol{I}_M\boldsymbol{U}^T\right)^{-1}(-\boldsymbol{J}^T)\boldsymbol{I}_M^{-1}\boldsymbol{e}\\ &= -\frac{1}{\lambda}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\left(\boldsymbol{I}_M + \boldsymbol{U}^T\frac{1}{\lambda}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)^{-1}\boldsymbol{e}\\ &= -\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\left(\lambda\boldsymbol{I}_M + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)^{-1}\boldsymbol{e}.\end{aligned} \tag{16}$$

Remark 2 When combined with a second identity, namely $\boldsymbol{V}\boldsymbol{A}^{-1}(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{V}) = (\boldsymbol{D} - \boldsymbol{V}\boldsymbol{A}^{-1}\boldsymbol{B})\boldsymbol{D}^{-1}\boldsymbol{V}$, one can directly derive from (15) Woodbury identity defined as [42] $(\boldsymbol{A}+\boldsymbol{U}\boldsymbol{B}\boldsymbol{V})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}(\boldsymbol{B}+\boldsymbol{V}\boldsymbol{A}^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{A}^{-1}$, or in the special case $\boldsymbol{B} = -\boldsymbol{D}^{-1}$, as $(\boldsymbol{A} - \boldsymbol{U}\boldsymbol{D}^{-1}\boldsymbol{V})^{-1} = \boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{U}(\boldsymbol{D} - \boldsymbol{V}\boldsymbol{A}^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{A}^{-1}$. Using Woodbury identity would, in fact, *structurally* not result into the closed-form update step (16) in an exact sense. Our construction involves a more general regularization function than the commonly used square norm, where the Woodbury identity can be equally useful, as its Hessian yields a multiple of the identity matrix.

Compared to (14), the clear advantage of the form (16) is that it requires the factorization of an $M \times M$ matrix rather than an $n_w \times n_w$ matrix, where the term $\boldsymbol{H}_h^{-1}$ can be conveniently obtained by exploiting its diagonal structure. Given these modifications, we proceed by making an assumption that defines the residual $\boldsymbol{e}$ and the Jacobian $\boldsymbol{J}$ in the region of convergence where we assume the starting point $\boldsymbol{\theta}_0$ of the process (16) lies.

Let $\boldsymbol{\theta}^*$ be a nondegenerate minimizer of $\mathcal{L}$, and define $\mathcal{B}_\epsilon(\boldsymbol{\theta}^*) := \{\boldsymbol{\theta}_k \in \mathbb{R}^{n_w} : \|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\| \leq \epsilon\}$, a closed ball of a sufficiently small radius $\epsilon \geq 0$ about $\boldsymbol{\theta}^*$. We denote by $\mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$ an open neighbourhood of the sublevel set $\Gamma(\mathcal{L}) := \{\boldsymbol{\theta}_k : \mathcal{L}(\boldsymbol{\theta}_k) \leq \mathcal{L}(\boldsymbol{\theta}_0)\}$, so that $\mathcal{B}_\epsilon(\boldsymbol{\theta}^*) = \mathrm{cl}(\mathcal{N}_\epsilon(\boldsymbol{\theta}^*))$. We then have $\mathcal{N}_\epsilon(\boldsymbol{\theta}^*) := \{\boldsymbol{\theta}_k \in \mathbb{R}^{n_w} : \|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\| < \epsilon\}$.

Assumption 4 (i) Each $\boldsymbol{e}_n(\boldsymbol{\theta}_k)$ and each $\boldsymbol{q}_n(\boldsymbol{\theta}_k)$ is Lipschitz smooth, and $\forall\boldsymbol{\theta}_k \in \mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$ there exists $\nu > 0$ such that $\|\boldsymbol{J}(\boldsymbol{\theta}_k)\boldsymbol{z}\| \geq \nu\|\boldsymbol{z}\|$.

(ii) $\lim_{k\to\infty}\left\|\mathbb{E}_m[\boldsymbol{H}(\boldsymbol{\theta}_k)] - \partial^2\mathcal{L}(\boldsymbol{\theta}_k)\right\| = 0$ almost surely whenever $\lim_{k\to\infty}\|\boldsymbol{g}_{\mathcal{L}}(\boldsymbol{\theta}_k)\| = 0$, $\forall\boldsymbol{\theta}_k \in \mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$, where $\mathbb{E}_m[\cdot]$ denotes[2] expectation with respect to m.

Remark 3 Assumption 4(i) implies that the singular values of $\boldsymbol{J}$ are uniformly bounded away from zero and $\exists\beta, \tilde{\beta} > 0$ such that $\|\boldsymbol{e}\| \leq \beta$, $\|\boldsymbol{J}\| = \|\boldsymbol{J}^T\| \leq \tilde{\beta}$, then as $\boldsymbol{Q} \succeq \boldsymbol{0}$, we have $\exists K_1$ such that $\boldsymbol{Q} \leq K_1\boldsymbol{I}$, and hence $\|\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\| \leq \lambda + (K/\gamma_a)$, where $K = K_1\tilde{\beta}^2$. Note that although we use limits in Assumption 4(ii), the assumption

[2] Subsequently, we shall omit the notation for the Hessian and gradient estimates as we assume unbiasedness.

similarly holds in expectation by unbiasedness. Also, a sufficient sample size m may be required for Assumption 4(ii) to hold, by law of large numbers, see e.g. [43, Lemma 1, Lemma 2].

Remark 4 $\boldsymbol{H}_g(\boldsymbol{\theta})$ and $\boldsymbol{H}_h(\boldsymbol{\theta})$ satisfy

$$\begin{aligned} \gamma_l \boldsymbol{I}_{n_w} \preceq \boldsymbol{H}_g(\boldsymbol{\theta}_k) \preceq \gamma_u \boldsymbol{I}_{n_w}, \quad & \left\| \boldsymbol{H}_g(\boldsymbol{y}, f(\boldsymbol{\theta_1}; \boldsymbol{x})) - \boldsymbol{H}_g(\boldsymbol{y}, f(\boldsymbol{\theta_2}; \boldsymbol{x})) \right\| \leq \gamma_g \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\|, \\ \gamma_a \boldsymbol{I}_{n_w} \preceq \boldsymbol{H}_h(\boldsymbol{\theta}_k) \preceq \gamma_b \boldsymbol{I}_{n_w}, \quad & \left\| \boldsymbol{H}_h(\boldsymbol{y}, f(\boldsymbol{\theta_1}; \boldsymbol{x})) - \boldsymbol{H}_h(\boldsymbol{y}, f(\boldsymbol{\theta_2}; \boldsymbol{x})) \right\| \leq \gamma_h \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\|, \end{aligned}$$

at any point $\boldsymbol{\theta}_k \in \mathbb{R}^{n_w}$, for any $\boldsymbol{x} \in \mathbb{R}^{n_p}$, $\boldsymbol{y} \in \mathbb{R}^d$, and for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$ where Assumption 4 holds.

We now state a convergence result for the update type (14) (and hence (16)). First, we define the second-order optimality condition and state two useful lemmas.

Definition 2 (Second-order sufficiency condition (SOSC)) Let $\boldsymbol{\theta}^*$ be a local minimum of a twice-differentiable function $\mathcal{L}(\cdot)$. The second-order sufficiency condition (SOSC) holds if

$$\partial \mathcal{L}(\boldsymbol{\theta}^*) = 0, \quad \partial^2 \mathcal{L}(\boldsymbol{\theta}^*) \succ 0. \tag{SOSC}$$

Lemma 1 ([14, Theorem 1.2.3]) *Suppose that Assumption 3 holds. Let* $\mathbb{W} \subseteq \mathbb{R}^{n_w}$ *be a closed and convex set on which* $\mathcal{L}(\boldsymbol{\theta})$ *is twice-continuously differentiable. Let* $S \subset \mathbb{W}$ *be an open set containing some* $\boldsymbol{\theta}^*$*, and suppose that* $\mathcal{L}(\boldsymbol{\theta}^*)$ *satisfies* (SOSC)*. Then, there exists* $\mathcal{L}^* = \mathcal{L}(\boldsymbol{\theta}^*)$ *satisfying*

$$\mathcal{L}(\boldsymbol{\theta}_k) > \mathcal{L}^* \quad \forall \boldsymbol{\theta}_k \in S. \tag{18}$$

Lemma 2 *The adjustment* $\delta\boldsymbol{\theta}$ *given by* (14) *(and hence,* (16)*) provides a descent direction for the total loss* $\mathcal{L}(\boldsymbol{\theta}_k)$ *in* (1) *at the* k*th oracle call.*

Remark 5 Lemma 1, Lemma 2 and the first part of (SOSC) ensure that the second part of Assumption 4(ii) always holds. In essence, it holds at every point $\boldsymbol{\theta}_k$ of the sequence $\{\boldsymbol{\theta}_k\}$ generated by the process (16) as long as we choose a starting point $\boldsymbol{\theta_0} \in \mathcal{B}_\epsilon(\boldsymbol{\theta}^*)$.

Theorem 3 *Suppose that Assumptions 1, 2, 3 and 4 hold, and that* $\boldsymbol{\theta}^*$ *is a local minimizer of* $\mathcal{L}(\boldsymbol{\theta})$ *for which the assumptions in Lemma 1 hold. Let* $\{\boldsymbol{\theta}_k\}$ *be the sequence generated by the process* (16)*. Then starting from a point* $\boldsymbol{\theta}_0 \in \mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$*,* $\{\boldsymbol{\theta}_k\}$ *converges at a Q-quadratic rate. Namely:*

$$\left\| \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^* \right\| \leq \xi_k \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|^2,$$

where

$$\xi_k = \frac{1}{2} \frac{\gamma_g + b\gamma_h}{\left(\gamma_l + a\gamma_a - (\gamma_g + b\gamma_h) \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|\right)}.$$

The proofs of these results are reported in “Appendix A” and “Appendix B”.

4 Self-concordant regularization (SCORE)

In the case $dm < n_w$, one could deduce by mere visual inspection that the matrix $\boldsymbol{H}_h^{-1}$ in (16) plays an important role in the perturbation of $\boldsymbol{\theta}$ within the region of convergence due to its "dominating" structure in the equation. This may be true as it appears. But beyond this "naive" view, let us derive a more technical intuition about the update step (16) by using a similar analogy as that given in [14, Chapter 5]. We have

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \boldsymbol{H}_h^{-1}\boldsymbol{J}^T\left(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)^{-1}\boldsymbol{e}, \tag{19}$$

where $\boldsymbol{I}$ is the identity matrix of suitable dimension. By some simple algebra (see e.g., [14, Lemma 5.1.1]), one can show that indeed the update method (19) is affine-invariant. In other words, the region of convergence $\mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$ does not depend on the problem scaling but on the local topological structure of the minimization functions $g(\boldsymbol{\theta})$ and $h(\boldsymbol{\theta})$ [14].

Consider the non-augmented form of (19): $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \boldsymbol{Q}_g\boldsymbol{J}_g^T\boldsymbol{G}_g^{-1}\boldsymbol{e}_g$, where $\boldsymbol{G}_g = \boldsymbol{J}_g\boldsymbol{J}_g^T$ is the so-called *Gram matrix*. It has been shown that indeed when learning an overparameterized model (sample size smaller than number of variables), and as long as we start from an initial point $\boldsymbol{\theta}_0$ close enough to the minimizer $\boldsymbol{\theta}^*$ of $\mathcal{L}(\boldsymbol{\theta})$ (assuming that such a minimizer exists), both $\boldsymbol{J}_g$ and $\boldsymbol{G}_g$ remain stable throughout the learning process (see e.g., [17, 44, 45]). The term $\boldsymbol{Q}_g$ may have little to no effect on this notion, for example, in case of the squared fit loss, $\boldsymbol{Q}_g$ is just an identity term. However, the original Equation (19) involves the Hessian matrix $\boldsymbol{H}_h$ which, together with its bounds (namely, γ_a, γ_b characterizing the region of convergence), changes rapidly when we scale the problem [14]. It therefore suffices to impose an additional assumption on the function $h(\boldsymbol{\theta})$ that will help to control the rate at which its second-derivative $\boldsymbol{H}_h$ changes, thereby enabling us to characterize an affine-invariant structure for the region of convergence. Namely:

Assumption 5 (SCORE) The scaled regularization function $\lambda h(\boldsymbol{\theta})$ has a third derivative and is M_h-self-concordant. That is, the inequality

$$\left|\left\langle \boldsymbol{u}, \left(\partial^3 h(\boldsymbol{\theta})[\boldsymbol{u}]\right)\boldsymbol{u}\right\rangle\right| \le 2M_h\left\langle \boldsymbol{u}, \partial^2 h(\boldsymbol{\theta})\boldsymbol{u}\right\rangle^{3/2}, \tag{20}$$

where $M_h \geq 0$, holds for any $\boldsymbol{\theta}$ in the closed convex set $\mathbb{W} \subseteq \mathbb{R}^{n_w}$ and $\boldsymbol{u} \in \mathbb{R}^{n_w}$.

Here, $\partial^3 h(\boldsymbol{\theta})[\boldsymbol{u}] \in \mathbb{R}^{n_w \times n_w}$ denotes the limit

$$\partial^3 h(\boldsymbol{\theta})[\boldsymbol{u}] := \lim_{t\to 0}\frac{1}{t}\left(\partial^2(\boldsymbol{\theta} + t\boldsymbol{u}) - \partial^2(\boldsymbol{\theta})\right), \quad t \in \mathbb{R}.$$

For a detailed analysis of self-concordant functions, we refer the interested reader to [14, 25].

Given this *extra* assumption, and following the idea of Newton-decrement in [25], we propose to update $\boldsymbol{\theta}$ by

$$\delta\boldsymbol{\theta} = -\frac{\alpha}{1+M_h\eta_k}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\left(\lambda\boldsymbol{I}+\boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)^{-1}\boldsymbol{e}, \tag{21}$$

where $\alpha > 0$, $\eta_k = \left\langle \boldsymbol{g}_h, \boldsymbol{H}_h^{-1}\boldsymbol{g}_h\right\rangle^{1/2}$ and $\boldsymbol{g}_h$ is the gradient of $h(\boldsymbol{\theta})$. The proposed method which we call GGN-SCORE is summarized, for one oracle call, in Algorithm 4.

Algorithm 1 GGN-SCORE

```
Input: variables vector $\boldsymbol{\theta}_k$, data $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^m$, $\boldsymbol{H}_h$ (see (13)), $\boldsymbol{Q}$ (see (13)), $\boldsymbol{J}$ (see (13)), $\boldsymbol{e}$ (see (12)),
    parameters $\alpha$, $M_h$, $\lambda$
Output: variables vector $\boldsymbol{\theta}_{k+1}$
Compute $\boldsymbol{g}_h = \partial_{\boldsymbol{\theta}_k} h(\boldsymbol{\theta}_k)$
Choose $\eta_k = \left\langle \boldsymbol{g}_h, \boldsymbol{H}_h^{-1}\boldsymbol{g}_h\right\rangle^{1/2}$
Set $\rho_k = \frac{\alpha}{1+M_h\eta_k}$
Solve $\left(\lambda\boldsymbol{I}+\boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)\boldsymbol{b} = \boldsymbol{e}$ for $\boldsymbol{b}$
Set $\boldsymbol{G} = \boldsymbol{H}_h^{-1}\boldsymbol{J}^T\boldsymbol{b}$
Compute $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \rho_k\boldsymbol{G}$
```

There is a wide class of self-concordant functions that meet practical requirements, either in their scaled form [14, Corollary 5.1.3] or in their original form (see e.g., in [46] for a comprehensive list). Two of them are used in our experiments in Sect. 5.

In the following, we state a local convergence result for the process (21). We introduce the notation $\|\Delta\|_{\boldsymbol{\theta}}$ to represent the local norm of a direction Δ taken with respect to the local Hessian of a self-concordant function h, $\|\Delta\|_{\boldsymbol{\theta}} := \left\langle \Delta, \partial^2_{\boldsymbol{\theta}\boldsymbol{\theta}} h(\boldsymbol{\theta})\Delta\right\rangle^{1/2} = \left\| \left[\partial^2_{\boldsymbol{\theta}\boldsymbol{\theta}} h(\boldsymbol{\theta})\right]^{1/2}\Delta\right\|$. Hence, without loss of generality, a ball $\mathcal{B}_\epsilon(\boldsymbol{\theta}^*)$ of radius ϵ about $\boldsymbol{\theta}^*$ is taken with respect to the local norm for the function h in the result that follows.

Theorem 4 *Let Assumptions 1, 2, 3, 4 and 5 hold, and let $\boldsymbol{\theta}^*$ be a local minimizer of $\mathcal{L}(\boldsymbol{\theta})$ for which the assumptions in Lemma 1 hold. Let $\{\boldsymbol{\theta}_k\}$ be the sequence generated by Algorithm 4 with $\alpha = \frac{\sqrt{\gamma_a}}{\beta_1}(K+\lambda\gamma_a)$, $\beta_1 = \beta\tilde{\beta}$. Then starting from a point $\boldsymbol{\theta}_0 \in \mathcal{N}_{M_h^{-1}}(\boldsymbol{\theta}^*)$, $\{\boldsymbol{\theta}_k\}$ converges to $\boldsymbol{\theta}^*$ according to the following descent properties:*

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{k+1})] \le \mathcal{L}(\boldsymbol{\theta}_k) - \left(\frac{\lambda}{M_h^2}\omega(\zeta_k) + \frac{\gamma_l}{2\gamma_a}\omega''(\tilde{\zeta}_k) - \xi\right),$$

$$\mathbb{E}\|\boldsymbol{\theta}_{k+1}-\boldsymbol{\theta}^*\|_{\boldsymbol{\theta}_{k+1}} \le \vartheta\|\boldsymbol{\theta}_k-\boldsymbol{\theta}^*\|_{\boldsymbol{\theta}_k} + \frac{\gamma_u}{\beta_1}\|\boldsymbol{\theta}_k-\boldsymbol{\theta}^*\| + \frac{\gamma_g}{2}\|\boldsymbol{\theta}_k-\boldsymbol{\theta}^*\|^2,$$

Table 1 Real datasets: N = number of data samples, n_p = number of features, d = number of targets

Dataset	N	n_p before & after feature mapping	d
dis	3772	29	1
		3017	
hypothyroid	3163	25	1
		2530	
w2a	3470	300	1
		2776	
ijcnn1	35000	22	1
		28000	
coil2000	9822	85	1
		7857	

where $\omega(\cdot)$ is an auxiliary univariate function defined by $\omega(t) := t - \ln(1+t)$ and has a second derivative $\omega''(t) = 1/(1+t)^2$, and

$$\zeta_k := \frac{M_h}{1 + M_h \eta_k}, \quad \tilde{\zeta}_k := M_h \eta_k, \quad \xi := \frac{2(\gamma_u + \lambda \gamma_b)}{\sqrt{\gamma_a}},$$
$$\vartheta := 1 + \frac{\lambda}{\sqrt{\gamma_a}\beta_1 (1 - M_h \|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\|_{\boldsymbol{\theta}_k})}.$$

The proof is provided in "Appendix B". The results of Theorem 4 combine strong convexity and smoothness properties of both $g(\boldsymbol{\theta})$ and $h(\boldsymbol{\theta})$, and requires that only $h(\boldsymbol{\theta})$ is self-concordant.

5 Experiments

In this section, we validate the efficiency of GGN-SCORE (Algorithm 4) in solving the problem of minimizing a regularized strongly convex quadratic function and in solving binary classification tasks. For the binary classification tasks, we use real datasets: ijcnn1 and w2a from the LIBSVM repository [47] and dis, hypothyroid and coil2000 from the PMLB repository [48], with an 80:20 train:test split each. The datasets are summarized in Table 1. In each classification task, the model with a sigmoidal output $\hat{\boldsymbol{y}}_n$ is trained using the cross-entropy fit loss function $\ell(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n) = \frac{1}{2}\sum_{n=1}^{N} \boldsymbol{y}_n \log\left(\frac{1}{\hat{\boldsymbol{y}}_n}\right) + (\mathbf{1} - \boldsymbol{y}_n)\log\left(\frac{1}{\mathbf{1}-\hat{\boldsymbol{y}}_n}\right)$, and the "deviance" residual [49] $\boldsymbol{e}_n = (-1)^{\boldsymbol{y}_n+1}\sqrt{-2\left[\boldsymbol{y}_n \log(\hat{\boldsymbol{y}}_n) + (\mathbf{1} - \boldsymbol{y}_n)\log(\mathbf{1} - \hat{\boldsymbol{y}}_n)\right]}$, $\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n \in \{\mathbf{0}, \mathbf{1}\}$.

We map the input space to higher dimensional feature space using the radial basis function (RBF) kernel $K(\boldsymbol{x}_n, \boldsymbol{x}_n') = \exp(-\gamma\|\boldsymbol{x}_n - \boldsymbol{x}_n'\|^2)$ with $\gamma = 0.1$. In each experiment, we use the penalty value $\lambda = 0.1$ with both pseudo-Huber regularization $h_\mu(\boldsymbol{\theta})$ [50] parameterized by $\mu > 0$ [51, 52] and ℓ^2 regularization $h_2(\boldsymbol{\theta})$ defined respectively as

$$h_\mu(\boldsymbol{\theta}) := \sqrt{\mu^2 + \boldsymbol{\theta}^2} - \mu, \qquad h_2(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2 := \sum_{i=1}^{n_w} |\theta_i|^2 ,$$

with coefficient $\mu = 1.0$. Throughout, we choose a batch size, m of 512 for `w2a`, `dis` and `hypothyroid`, 2048 for `coil2000` and 4096 for `ijcnn1`, unless otherwise stated. We assume a scaled self-concordant regularization so that $M_h = 1$ [14, Corollary 5.1.3].

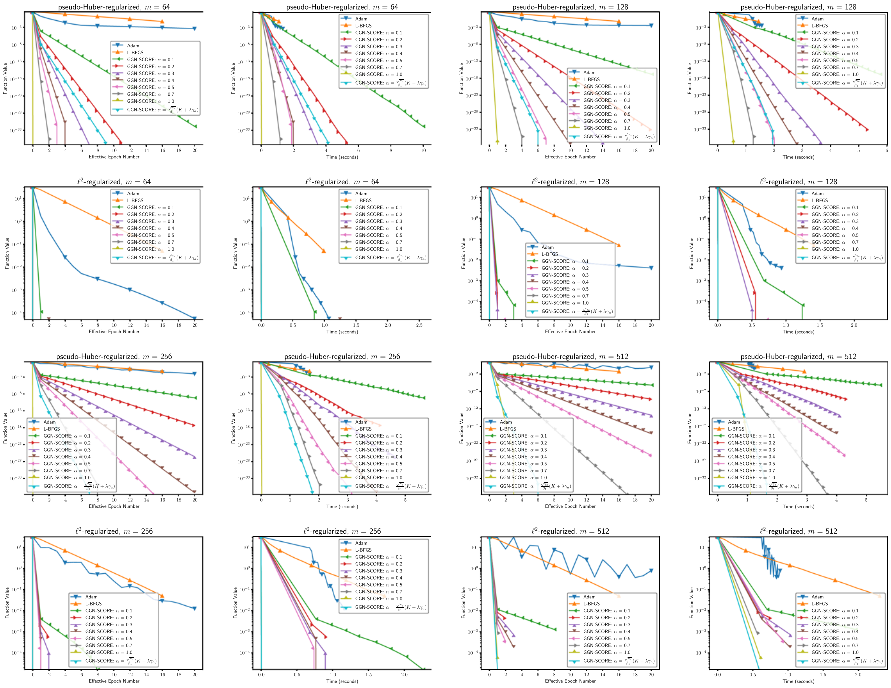
5.1 GGN-SCORE for different values of α

To illustrate the behaviour of GGN-SCORE for different values of α in Algorithm 4 versus its value indicated in Theorem 4, we consider the problem of minimizing a regularized strongly convex quadratic function:

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) := \frac{1}{2}\boldsymbol{\theta}^\top \hat{\boldsymbol{Q}}\boldsymbol{\theta} - \boldsymbol{p}^\top\boldsymbol{\theta} + \lambda h(\boldsymbol{\theta}) \equiv g(\boldsymbol{\theta}) + \lambda h(\boldsymbol{\theta}), \tag{22}$$

where $\hat{\boldsymbol{Q}} \in \mathbb{R}^{n_w \times n_w}$ is symmetric positive definite, $\boldsymbol{p} \in \mathbb{R}^{n_w}$, g is γ_a-strongly convex and has γ_u-Lipschitz gradient, with the smallest and largest eigenvalues of $\hat{\boldsymbol{Q}}$ corresponding to γ_a and γ_u, respectively. For this function, suppose the gradient and Hessian of $h(\boldsymbol{\theta})$ is known, for example when we choose $h = h_\mu$ or $h = h_2$, we have $\partial\mathcal{L}(\boldsymbol{\theta}) = \hat{\boldsymbol{Q}}\boldsymbol{\theta} - \boldsymbol{p} + \lambda\partial h(\boldsymbol{\theta})$ and $\partial^2\mathcal{L}(\boldsymbol{\theta}) = \hat{\boldsymbol{Q}} + \lambda\partial^2 h(\boldsymbol{\theta})$. The coefficients $\hat{\boldsymbol{Q}}$ and $\boldsymbol{p}$ form our data, and with $\hat{\boldsymbol{Q}} \equiv 0.1 \times \boldsymbol{M}^\top\boldsymbol{M}$, $N = n_w = 1000$, we generate the data $\boldsymbol{M} \in \mathbb{R}^{N\times N}$ randomly from a uniform distribution on $[0, 1]$ and consider the case $\mathcal{L}^* = \mathbf{0}$ in which $\boldsymbol{p}$ is the zero vector. The optimization variable $\boldsymbol{\theta}$ is initialized to a random value generated from a normal distribution with mean 0 and standard deviation 0.01. Figure 1 shows the behaviour of GGN-SCORE for this problem with different values of α in $(0, 1]$ and $\alpha = \frac{\sqrt{\gamma_a}}{\beta_1}(K + \lambda\gamma_a)$ indicated in Theorem 4. We experiment with different batch sizes $m \in \{64, 128, 256, 512\}$. One observes from Fig. 1 that larger batch size yields better convergence speed when we choose $\alpha = \frac{\sqrt{\gamma_a}}{\beta_1}(K + \lambda\gamma_a)$, validating the recommendation in Remark 3. Figure 1 also shows the comparison of GNN-SCORE with the first-order Adam [5] algorithm, and the quasi-Newton Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) [53] method using optimally tuned learning rates. While choosing $\alpha = \frac{\sqrt{\gamma_a}}{\beta_1}(K + \lambda\gamma_a)$ yields the kind of convergence shown in Theorem 4, Fig. 1 shows that by choosing α in $(0, 1]$, we can similarly achieve a great convergence that scale well with the problem.

Strictly speaking, the value of α indicated in Theorem 4 is not of practical interest, as it contains terms that may not be straightforward to retrieve in practice. In practice, we treat α as a hyperparameter that takes a *fixed* positive value in $(0, 1]$. For an adaptive step-size selection rule, such as that in Line 5 of Algorithm 4, choosing a suitable scaling constant such as α is often straightforward, as the main step-size selection task


pseudo-Huber-regularized, $m = 64$ — pseudo-Huber-regularized, $m = 64$ — pseudo-Huber-regularized, $m = 128$ — pseudo-Huber-regularized, $m = 128$

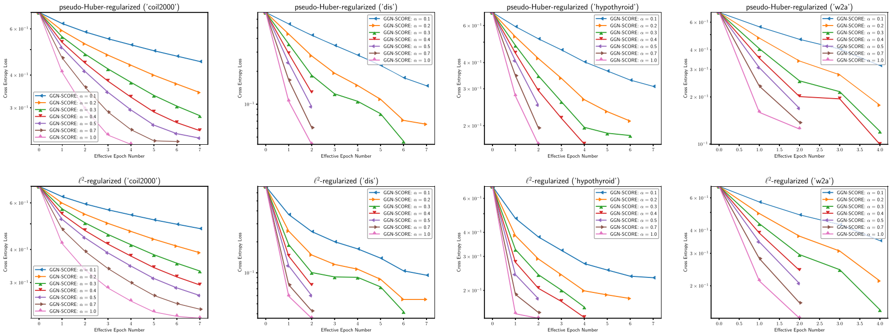
ℓ^2-regularized, $m = 64$ — ℓ^2-regularized, $m = 64$ — ℓ^2-regularized, $m = 128$ — ℓ^2-regularized, $m = 128$

pseudo-Huber-regularized, $m = 256$ — pseudo-Huber-regularized, $m = 256$ — pseudo-Huber-regularized, $m = 512$ — pseudo-Huber-regularized, $m = 512$

ℓ^2-regularized, $m = 256$ — ℓ^2-regularized, $m = 256$ — ℓ^2-regularized, $m = 512$ — ℓ^2-regularized, $m = 512$

Function Value
Effective Epoch Number
Time (seconds)
Adam
L-BFGS
GGN-SCORE: $\alpha = 0.1$
GGN-SCORE: $\alpha = 0.2$
GGN-SCORE: $\alpha = 0.3$
GGN-SCORE: $\alpha = 0.4$
GGN-SCORE: $\alpha = 0.5$
GGN-SCORE: $\alpha = 0.7$
GGN-SCORE: $\alpha = 1.0$
GGN-SCORE: $\alpha = \frac{\sqrt{\gamma_\alpha}}{\beta_1}(K + \lambda\gamma_\alpha)$

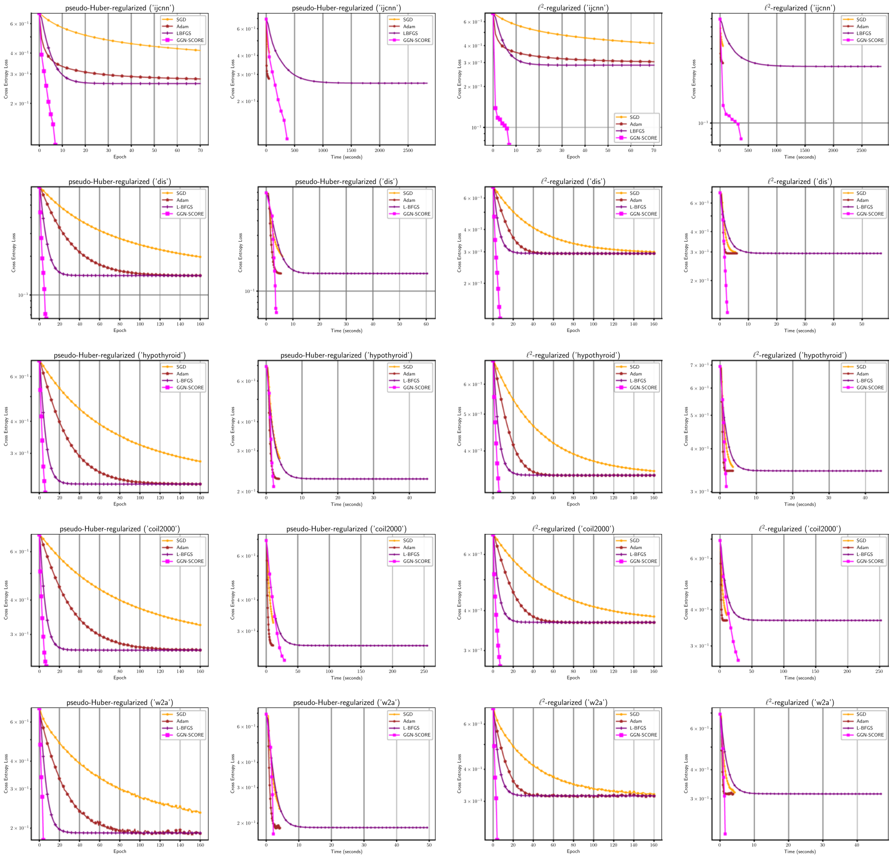
Fig. 1 Numerical behaviour of GGN-SCORE for different values of α in the strongly convex quadratic test problem (22)


pseudo-Huber-regularized ('coil2000') — pseudo-Huber-regularized ('dis') — pseudo-Huber-regularized ('hypothyroid') — pseudo-Huber-regularized ('w2a')

ℓ^2-regularized ('coil2000') — ℓ^2-regularized ('dis') — ℓ^2-regularized ('hypothyroid') — ℓ^2-regularized ('w2a')

Fig. 2 Numerical behaviour of GGN-SCORE for different values of α using real datasets. $m = 512$ for `w2a`, `dis` and `hypothyroid`, and $m = 2048$ for `coil2000`

is accomplished by the defined rule. We show the behaviour of GGN-SCORE on the real datasets for different values of α in $(0, 1]$ in Fig. 2. In general, a suitable scaling factor α should be selected based on the application demands.



pseudo-Huber-regularized ('ijcnn')
pseudo-Huber-regularized ('ijcnn')
ℓ^2-regularized ('ijcnn')
ℓ^2-regularized ('ijcnn')
pseudo-Huber-regularized ('dis')
pseudo-Huber-regularized ('dis')
ℓ^2-regularized ('dis')
ℓ^2-regularized ('dis')
pseudo-Huber-regularized ('hypothyroid')
pseudo-Huber-regularized ('hypothyroid')
ℓ^2-regularized ('hypothyroid')
ℓ^2-regularized ('hypothyroid')
pseudo-Huber-regularized ('coil2000')
pseudo-Huber-regularized ('coil2000')
ℓ^2-regularized ('coil2000')
ℓ^2-regularized ('coil2000')
pseudo-Huber-regularized ('w2a')
pseudo-Huber-regularized ('w2a')
ℓ^2-regularized ('w2a')
ℓ^2-regularized ('w2a')

Fig. 3 Convergence curves for GGN-SCORE, SGD, Adam and L-BFGS in the proposed convex problem. $m = 512$ for `w2a`, `dis` and `hypothyroid`, $m = 2048$ for `coil2000`, and $m = 4096$ for `ijcnn1`

5.2 Comparison with SGD, Adam, and L-BFGS methods on real datasets

Using the real datasets, we compare GGN-SCORE for solving (1) with results from the SGD, Adam, and the L-BFGS algorithms using optimally tuned learning rates. We also consider the training problem of a neural network with two hidden layers of dimensions (2, 128), respectively for the `coil2000` dataset, one hidden layer with dimension 1 for the `ijcnn1` dataset, and two hidden layers of dimensions (4, 128), respectively for the remaining datasets. We use ReLU activation functions in the hidden layers of the networks, and the network is overparameterized for `dis`, `hypothyroid`, `w2a` and `coil2000` with 25425, 21529, 23497 and 16229 trainable parameters, respectively. We choose $\alpha \in \{0.2, 0.5\}$ for GGN-SCORE. Minimization variables are initialized to the zero vector for all the methods. The neural network training problems are solved under the same settings. The results are respectively displayed in Fig. 3 and Fig. 4 for the convex and non-convex cases.

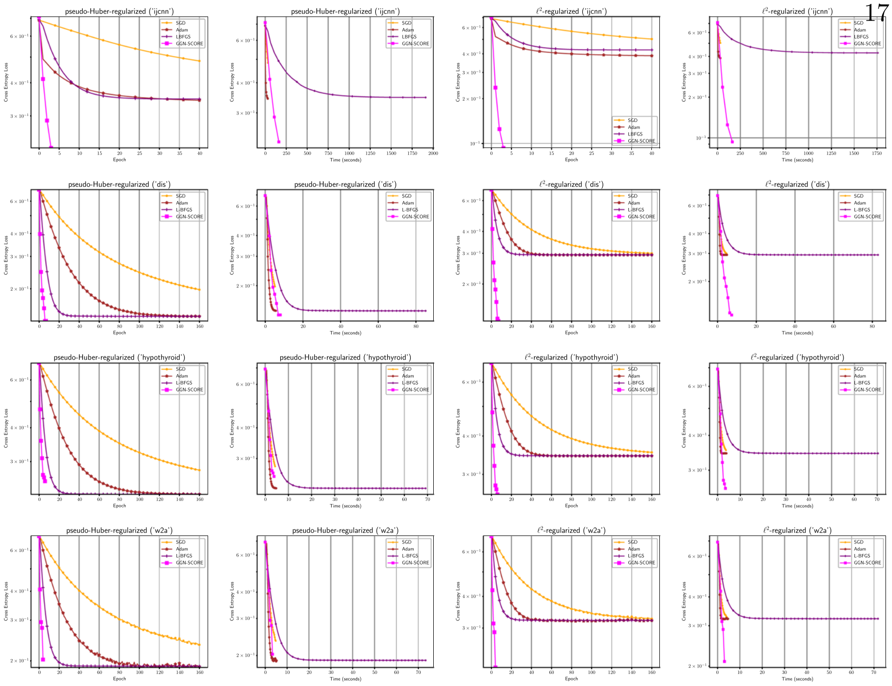
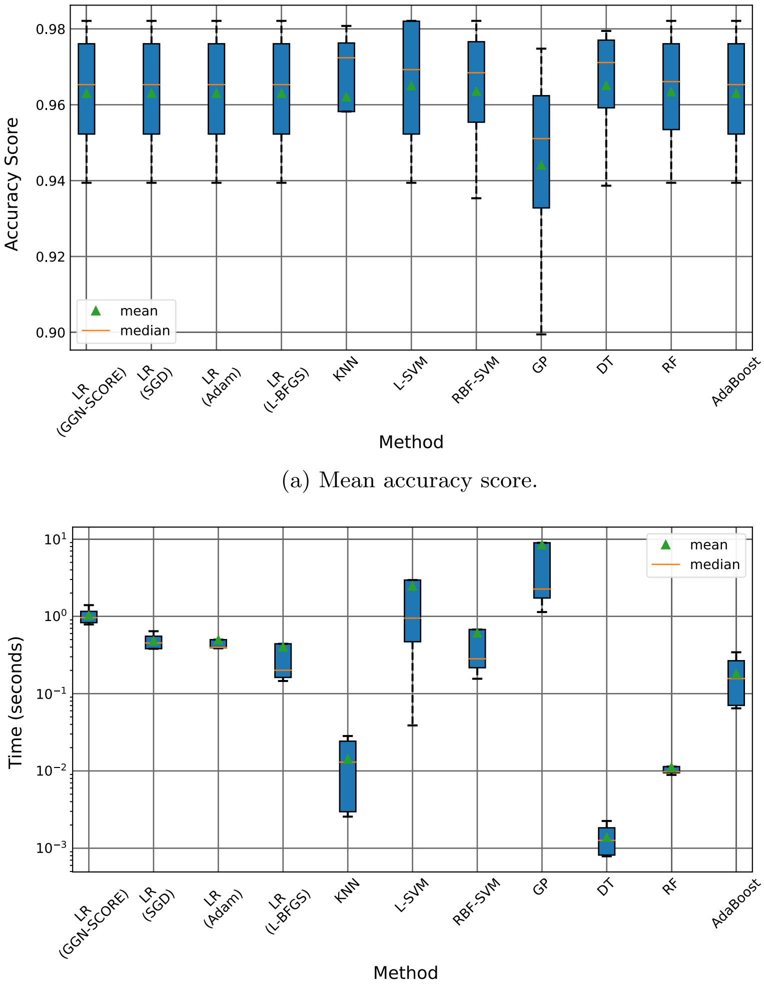


Fig. 4 Convergence curves for GGN-SCORE, SGD, Adam and L-BFGS in the non-convex neural network training problem: overparameterized for `dis`, `hypothyroid`, `w2a` and `coil2000`. $m = 512$ for `w2a`, `dis` and `hypothyroid`, $m = 2048$ for `coil2000`, and $m = 4096$ for `ijcnn1`

To investigate how well the learned model generalizes, we use the binary accuracy metric which measures how often the model predictions match the true labels when presented with new, previously unseen data: $Accuracy = \frac{1}{N}\sum_{n=1}^{N}\left(2\boldsymbol{y}_n\hat{\boldsymbol{y}}_n - \boldsymbol{y}_n - \hat{\boldsymbol{y}}_n + 1\right)$. While GGN-SCORE converges faster than SGD, Adam and L-BFGS methods, it generalizes comparatively well. The results are further compared with other known binary classification techniques to measure the quality of our solutions. The accuracy scores for `dis`, `hypothyroid`, `coil2000` and `w2a` datasets, with a 60:40 train:test split each, are computed on the test set. The mean scores are compared with those from the different classification techniques, and are shown in Fig. 5. The CPU runtimes are also compared where it is indicated that on average GGN-SCORE solves each of the problems within one second. This scales well with the other techniques, as we note that while GNN-SCORE solves each of the problems in high dimensions, the success of most of the other techniques are limited to relatively smaller dimensions of the problems. The obtained results from the classification techniques used for comparison are computed directly from the respective scikit-learn [54] functions.

The L-BFGS experiments are implemented with PyTorch [55] (v. 1.10.1+cu102) in full-batch mode. The GGN-SCORE, Adam and SGD methods are implemented using the open-source Keras API with TensorFlow [56] backend (v. 2.7.0). All experiments



(a) Mean accuracy score.

(b) Mean CPU time (in seconds).

Fig. 5 Classification task results using the method of this paper – logistic regression (LR), k-nearest neighbours (KNN), linear support vector machine (L-SVM), RBF-SVM, Gaussian process (GP), decision tree (DT), random forest (RF) and adaptive boosting (AdaBoost) techniques for `dis`, `hypothyroid`, `coil2000` and `w2a` datasets. LR and SVM methods use the ℓ_2 penalty function with parameter $\lambda = 1.0$. Solutions with GGN-SCORE, SGD, Adam, and L-BFGS are obtained over one data pass with $m = 128$

are performed on a laptop with dual (2.30GHz + 2.30GHz) Intel Core i7-11800 H CPU and 16GB RAM.

In summary, GGN-SCORE converges way faster (in terms of number of epochs) than SGD, Adam, and L-BFGS, and generalizes comparatively well. Experimental results show the computational convenience and elegance achieved by our "augmented" approach for including regularization functions in the GGN approximation. Although GGN-SCORE comes with a higher computational cost (in terms of wallclock time per iteration) than first-order methods on average, if per-iteration learning time is not provided as a bottleneck, this may not become an obvious issue as we need to pass the proposed optimizer on the dataset only a few times (epochs) to obtain superior function approximation and relatively high-quality solutions in our experiments.

6 Conclusion

In this paper, we have proposed GGN-SCORE, a generalized Gauss–Newton-type algorithm for solving unconstrained regularized minimization problems, where the regularization function is considered to be self-concordant. In this generalized setting, we employed a matrix approximation scheme that significantly reduces the computational overhead associated with the method. Unlike existing techniques that impose self-concordance on the problem's objective function, our analysis involves a less restrictive condition from a practical point of view but similarly benefits from the idea of self-concordance by considering scaled optimization step-lengths that depend on the self-concordant parameter of the regularization function. We proved a quadratic local convergence rate for our method under certain conditions, and validate its efficiency in numerical experiments that involve both strongly convex problems and the general non-convex problems that arise when training neural networks. In both cases, our method compare favourably against Adam, SGD, and L-BFGS methods in terms of per-iteration convergence speed, as well as some machine learning techniques used in binary classification, in terms of solution quality.

In future research, it would be interesting to relax some conditions on the problem and analyze a global convergence rate for the proposed method. We would also consider an analysis for our method in a general non-convex setting even though numerically we have observed a similar convergence speed as the strongly convex case.

Funding Open access funding provided by Scuola IMT Alti Studi Lucca within the CRUI-CARE Agreement

Data availability The datasets analysed during the current study are available in the LIBSVM repository [47], https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html and the PMLB repository [48], https://epistasislab.github.io/pmlb/.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

Appendix A useful results

Following Assumptions 1 – 3, we get that [14, Theorem 2.1.6]

$$\gamma_l \boldsymbol{I}_{n_w} \preceq \partial^2 g(\boldsymbol{\theta}) \preceq \gamma_u \boldsymbol{I}_{n_w}, \quad \gamma_a \boldsymbol{I}_{n_w} \preceq \partial^2 h(\boldsymbol{\theta}) \preceq \gamma_b \boldsymbol{I}_{n_w}, \tag{A1}$$

where $\boldsymbol{I}_{n_w} \in \mathbb{R}^{n_w \times n_w}$ is an identity matrix. Consequently,

$$(\gamma_l + \lambda\gamma_a)\boldsymbol{I}_{n_w} \preceq \partial^2 \mathcal{L}(\boldsymbol{\theta}) \preceq (\gamma_u + \lambda\gamma_b)\boldsymbol{I}_{n_w}. \tag{A2}$$

In addition, the second derivative of $\mathcal{L}(\boldsymbol{\theta})$ is $(\gamma_g + \lambda\gamma_h)$-Lipschitz continuous $\forall \boldsymbol{x} \in \mathbb{R}^{n_p}$, $\boldsymbol{y} \in \mathbb{R}^d$, that is,

$$\left\| \partial^2 \mathcal{L}(\boldsymbol{y}, f(\boldsymbol{\theta_1}; \boldsymbol{x})) - \partial^2 \mathcal{L}(\boldsymbol{y}, f(\boldsymbol{\theta_2}; \boldsymbol{x})) \right\| \leq (\gamma_g + \lambda\gamma_h) \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|. \tag{A3}$$

Lemma 5 *Let Assumptions 1, 2 and 3 hold. Let $\boldsymbol{\theta_1}, \boldsymbol{\theta_2}$ be any two points in $\mathbb{R}^{n_w}$. Then we have*

$$\|\boldsymbol{g}(\boldsymbol{\theta_1}) - \boldsymbol{g}(\boldsymbol{\theta_2}) - \langle \boldsymbol{H}(\boldsymbol{\theta_2}), \boldsymbol{\theta_1} - \boldsymbol{\theta_2}\rangle\| \leq \frac{\gamma_g + \lambda\gamma_h}{2} \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|^2, \tag{A4}$$

$$\left| \mathcal{L}(\boldsymbol{\theta_1}) - \mathcal{L}(\boldsymbol{\theta_2}) - \langle \boldsymbol{g}(\boldsymbol{\theta_2}), \boldsymbol{\theta_1} - \boldsymbol{\theta_2}\rangle - \frac{1}{2} \langle \boldsymbol{H}(\boldsymbol{\theta_2})(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}), \boldsymbol{\theta_1} - \boldsymbol{\theta_2}\rangle \right|$$
$$\leq \frac{\gamma_g + \lambda\gamma_h}{6} \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|^3. \tag{A5}$$

Proof Fix $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \mathbb{R}^{n_w}$. Then

$$\boldsymbol{g}(\boldsymbol{\theta_1}) - \boldsymbol{g}(\boldsymbol{\theta_2}) = \int_0^1 \partial^2 \mathcal{L}(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}))(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) d\tau.$$

As $\tau \in [0, 1]$, we have $\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) \in \mathbb{R}^{n_w}$. Hence writing $\partial^2 \mathcal{L} = \boldsymbol{H}$, we have

$$\begin{aligned}
&\boldsymbol{g}(\boldsymbol{\theta_1}) - \boldsymbol{g}(\boldsymbol{\theta_2}) \\
&\quad = \int_0^1 \boldsymbol{H}(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}))(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) d\tau \\
&\boldsymbol{g}(\boldsymbol{\theta_1}) - \boldsymbol{g}(\boldsymbol{\theta_2}) - \boldsymbol{H}(\boldsymbol{\theta_2})(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) \\
&\quad = \int_0^1 (\boldsymbol{H}(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2})) - \boldsymbol{H}(\boldsymbol{\theta_2}))(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) d\tau \\
&\|\boldsymbol{g}(\boldsymbol{\theta_1}) - \boldsymbol{g}(\boldsymbol{\theta_2}) - \boldsymbol{H}(\boldsymbol{\theta_2})(\boldsymbol{\theta_1} - \boldsymbol{\theta_2})\| \\
&\quad = \left\| \int_0^1 (\boldsymbol{H}(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2})) - \boldsymbol{H}(\boldsymbol{\theta_2}))(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) d\tau \right\| \\
&\quad \leq \int_0^1 \|(\boldsymbol{H}(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2})) - \boldsymbol{H}(\boldsymbol{\theta_2}))(\boldsymbol{\theta_1} - \boldsymbol{\theta_2})\| d\tau
\end{aligned}$$

$$\begin{aligned}
&\leq \int_0^1 \left\| \boldsymbol{H}(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2})) - \boldsymbol{H}(\boldsymbol{\theta_2}) \right\| \cdot \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\| d\tau \\
&\leq \int_0^1 \tau(\gamma_g + \lambda\gamma_h) \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\|^2 d\tau \\
&= \frac{\gamma_g + \lambda\gamma_h}{2} \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\|^2 .
\end{aligned}$$

The proof of (A5) follows immediately using a similar procedure (see e.g., [57]). □

Corollary 5.1 *Let Assumptions 1, 2 and 3 hold. Let $\boldsymbol{\theta_1}, \boldsymbol{\theta_2}$ be any two points in $\mathbb{R}^{n_w}$. Then by writing $\partial^2 \mathcal{L} = \boldsymbol{H}$,*

$$\boldsymbol{H}(\boldsymbol{\theta_2}) - (\gamma_g + \lambda\gamma_h) \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\| \leq \boldsymbol{H}(\boldsymbol{\theta_1}) \leq \boldsymbol{H}(\boldsymbol{\theta_2}) + (\gamma_g + \lambda\gamma_h) \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\| . \quad \text{(A6)}$$

Proof The proof follows immediately by recalling for any $\boldsymbol{\theta} \in \mathbb{R}^{n_w}$, $\boldsymbol{H}(\boldsymbol{\theta})$ is positive definite, and hence the eigenvalues s_j of the difference $\boldsymbol{H}(\boldsymbol{\theta_1}) - \boldsymbol{H}(\boldsymbol{\theta_2})$ satisfy

$$\left| s_j \right| \leq (\gamma_g + \lambda\gamma_h) \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\| , \quad j = 1, 2, \ldots, n_w,$$

so that we have

$$-(\gamma_g + \lambda\gamma_h) \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\| \leq \boldsymbol{H}(\boldsymbol{\theta_1}) - \boldsymbol{H}(\boldsymbol{\theta_2}) \leq (\gamma_g + \lambda\gamma_h) \left\| \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \right\| .$$

□

Lemma 6 ([14, Theorem 2.1.5]) *Let the first derivative of a function $\Phi(\cdot)$ be L-Lipschitz on* dom(Φ). *Then for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in$* dom$(\Phi)$, *we have*

$$0 \leq \Phi(\boldsymbol{\theta_1}) - \Phi(\boldsymbol{\theta_2}) - \langle \partial\Phi(\boldsymbol{\theta_2}), \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \rangle \leq \frac{L}{2} \|\boldsymbol{\theta_2} - \boldsymbol{\theta_1}\|^2.$$

Definition 3 ([14, Definition 2.1.3]) A continuously differentiable function $\psi(\cdot)$ is γ_{sc}-strongly convex on dom(ψ) if for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in$ dom(ψ), we have

$$\psi(\boldsymbol{\theta_1}) \geq \psi(\boldsymbol{\theta_2}) + \langle \partial\psi(\boldsymbol{\theta_2}), \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \rangle + \frac{\gamma_{sc}}{2} \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|^2, \quad \gamma_{sc} > 0.$$

We remark that if the first derivative of the function $\psi(\cdot)$ in the above definition is L-Lipschitz continuous, then by construction, for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in$ dom(ψ), ψ satisfies the Lipschitz constraint

$$|\psi(\boldsymbol{\theta_1}) - \psi(\boldsymbol{\theta_2})| \leq L\|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|. \quad \text{(A7)}$$

Lemma 7 ([14, Theorem 5.1.8]) *Let the function $\phi(\cdot)$ be M_ϕ-self-concordant. Then, for any $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \operatorname{dom}(\phi)$, we have*

$$\phi(\boldsymbol{\theta_1}) \geq \phi(\boldsymbol{\theta_2}) + \langle \partial\phi(\boldsymbol{\theta_2}), \boldsymbol{\theta_1} - \boldsymbol{\theta_2} \rangle + \frac{1}{M_\phi^2}\omega(M_\phi \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|_{\boldsymbol{\theta_2}}),$$

where $\omega(\cdot)$ is an auxiliary univariate function defined by $\omega(t) := t - \ln(1+t)$.

Lemma 8 ([14, Corollary 5.1.5]) *Let the function $\phi(\cdot)$ be M_ϕ-self-concordant. Let $\boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \operatorname{dom}(\phi)$ and $r = \|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\|_{\boldsymbol{\theta_1}} < \frac{1}{M_\phi}$. Then*

$$\left(1 - M_\phi r + \frac{1}{3}M_\phi^2 r^2\right)\partial^2\phi(\boldsymbol{\theta_1}) \preceq \int_0^1 \partial^2\phi(\boldsymbol{\theta_2} + \tau(\boldsymbol{\theta_1} - \boldsymbol{\theta_2}))d\tau \preceq \frac{1}{1 - M_\phi r}\partial^2\phi(\boldsymbol{\theta_1}).$$

Appendix B missing proofs

Proof of Lemma 2 By the arguments of Remark 1, we have that the matrix $(\boldsymbol{J}(\boldsymbol{\theta}_k)^T \boldsymbol{Q} \boldsymbol{J}(\boldsymbol{\theta}_k) + \lambda \boldsymbol{H}_h(\boldsymbol{\theta}_k))^{-1}$ is positive definite. Hence, with $\boldsymbol{g}(\boldsymbol{\theta}_k) \neq \boldsymbol{0}$, we have $\boldsymbol{g}\boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\boldsymbol{g}(\boldsymbol{\theta}_k) = -\delta\boldsymbol{\theta}^T\boldsymbol{g}(\boldsymbol{\theta}_k) > 0$ and $\delta\boldsymbol{\theta}^T\boldsymbol{g}(\boldsymbol{\theta}_k) < \boldsymbol{0}$. □

Proof of Theorem 3 The process formulated in (16) performs the update

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\boldsymbol{g}(\boldsymbol{\theta}_k).$$

As $\boldsymbol{g}(\boldsymbol{\theta}^*) = 0$ by mean value theorem and the first part of (SOSC), we have

$$\begin{aligned}
\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^* &= \boldsymbol{\theta}_k - \boldsymbol{\theta}^* - \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\int_0^1 \partial^2\mathcal{L}(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)d\tau \\
&= \left[\boldsymbol{I} - \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\int_0^1 \partial^2\mathcal{L}(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))d\tau\right](\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \\
&= \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\int_0^1 \Big(\boldsymbol{H}(\boldsymbol{\theta}_k) - \partial^2\mathcal{L}(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))\Big)d\tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \\
\left\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\right\| &= \left\|\boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\int_0^1 \Big(\boldsymbol{H}(\boldsymbol{\theta}_k) - \partial^2\mathcal{L}(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))\Big)d\tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)\right\| \\
&\leq \left\|\boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\right\|\int_0^1 \left\|\boldsymbol{H}(\boldsymbol{\theta}_k) - \partial^2\mathcal{L}(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))\right\|d\tau\left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|
\end{aligned}$$

Also, as $\tau \in [0, 1]$ we have that $\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \in \mathcal{B}_\epsilon(\boldsymbol{\theta}^*) \subseteq \mathbb{R}^{n_w}$. By taking the limit $\lim\limits_{k\to\infty}\left\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\right\|$, we have by the first part of (SOSC) and Assumption 4(ii)

$$\left\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\right\| \leq \left\|\boldsymbol{H}(\boldsymbol{\theta}_k)^{-1}\right\|\int_0^1 \left\|\boldsymbol{H}(\boldsymbol{\theta}_k) - \boldsymbol{H}(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))\right\|d\tau\left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|$$

$$\leq \left\| \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1} \right\| \int_0^1 (1-\tau)(\gamma_g + \lambda\gamma_h) \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| d\tau \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|$$
$$= \frac{\gamma_g + \lambda\gamma_h}{2} \left\| \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1} \right\| \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2.$$

By combining the claims in Corollary 5.1 and the bounds of $\boldsymbol{H}(\boldsymbol{\theta}_k)$ in (17), we obtain the relation

$$\gamma_l + a\gamma_a - (\gamma_g + \lambda\gamma_h) \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| \preceq \boldsymbol{H}(\boldsymbol{\theta}^*) - (\gamma_g + \lambda\gamma_h) \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| \preceq \boldsymbol{H}(\boldsymbol{\theta}_k).$$

Recall that $\boldsymbol{H}(\boldsymbol{\theta}_k)$ is positive definite, and hence invertible. We deduce that, indeed for all $\boldsymbol{\theta}_k$ satisfying $\left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| \leq \epsilon$, ϵ small enough, we have

$$\left\| \boldsymbol{H}(\boldsymbol{\theta}_k)^{-1} \right\| \leq \left(\gamma_l - \gamma_g - \lambda(\gamma_h - \gamma_a) \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|\right)^{-1}.$$

Therefore,

$$\left\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\right\| \leq \xi_k \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2,$$

where

$$\xi_k = \frac{1}{2} \frac{\gamma_g + \lambda\gamma_h}{\left(\gamma_l - \gamma_g - \lambda(\gamma_h - \gamma_a) \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|\right)}.$$

□

Proof of Theorem 4 First, we upper bound the norm $\|\delta\boldsymbol{\theta}\|_{\boldsymbol{\theta}_k} := \|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|_{\boldsymbol{\theta}_k} = \|\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_{k+_1} - \boldsymbol{\theta}_k)\|$. From Remark 3, we have

$$\|\boldsymbol{J}(\boldsymbol{\theta}_k)^T(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k))^{-1}\|$$
$$\leq \frac{\gamma_a \|\boldsymbol{J}(\boldsymbol{\theta}_k)^T\|}{K + \lambda\gamma_a} \leq \frac{\tilde{\beta}\gamma_a}{K + \lambda\gamma_a}.$$

Hence,

$$\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|_{\boldsymbol{\theta}_k} \leq \frac{\alpha}{1 + M_h\eta_k} \left\| \boldsymbol{H}_h^{1/2}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k)(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k))^{-1}\boldsymbol{e}(\boldsymbol{\theta}_k)\right\|$$
$$\leq \frac{\alpha}{1 + M_h\eta_k} \left\| \boldsymbol{H}_h(\boldsymbol{\theta}_k)^{1/2} \right\| \left\| \boldsymbol{H}_h(\boldsymbol{\theta}_k)^{-1} \right\| \left\| \boldsymbol{J}(\boldsymbol{\theta}_k)^T(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k))^{-1} \right\| \|\boldsymbol{e}(\boldsymbol{\theta}_k)\|$$
$$\leq \frac{\alpha\beta\tilde{\beta}\gamma_a^{-1/2}}{(K + \lambda\gamma_a)(1 + M_h\eta_k)}. \tag{B8}$$

Similarly, we have

$$\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\| \leq \frac{\alpha\beta\tilde{\beta}}{(K + \lambda\gamma_a)(1 + M_h\eta_k)}. \tag{B9}$$

By Lemma 6, the function $\mathcal{L}(\boldsymbol{\theta})$ satisfies

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}_{k+1}) &\leq \mathcal{L}(\boldsymbol{\theta}_k) + \langle \partial\mathcal{L}(\boldsymbol{\theta}_k), \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k \rangle + \frac{\gamma_u + \lambda\gamma_b}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2 \\ &= \mathcal{L}(\boldsymbol{\theta}_k) + \left\langle \boldsymbol{g}_g(\boldsymbol{\theta}_k), \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k \right\rangle + \lambda \left\langle \boldsymbol{g}_h(\boldsymbol{\theta}_k), \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k \right\rangle \frac{\gamma_u + \lambda\gamma_b}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2.\end{aligned}$$

By convexity of g and h, and self-concordance of h, we have (using Definition 3, Lemma 7 and (A7))

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}_{k+1}) &\leq \mathcal{L}(\boldsymbol{\theta}_k) + (\gamma_u + \lambda\gamma_b)\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\| + \frac{\gamma_u + \lambda\gamma_b - \gamma_l}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2 \\ &\quad - \frac{\lambda}{M_h^2}\omega(M_h\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|_{\boldsymbol{\theta}_k}) \\ &\overset{(B8)}{\leq} \mathcal{L}(\boldsymbol{\theta}_k) + (\gamma_u + \lambda\gamma_b)\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\| + \frac{\gamma_u + \lambda\gamma_b - \gamma_l}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2 \\ &\quad - \frac{\lambda}{M_h^2}\omega\left(\frac{\alpha\beta\tilde{\beta}\gamma_a^{-1/2}M_h}{(K + \lambda\gamma_a)(1 + M_h\eta_k)}\right).\end{aligned}$$

Substituting the choice $\alpha = (\beta\tilde{\beta})^{-1}(\gamma_a)^{1/2}(K + \lambda\gamma_a)$, we have

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}_{k+1}) \leq\ & \mathcal{L}(\boldsymbol{\theta}_k) + (\gamma_u + \lambda\gamma_b)\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\| + \frac{\gamma_u + \lambda\gamma_b - \gamma_l}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2 \\ & - \frac{\lambda}{M_h^2}\omega\left(\frac{M_h}{1 + M_h\eta_k}\right).\end{aligned}$$

Taking expectation on both sides with respect to m conditioned on $\boldsymbol{\theta}_k$, we get

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{k+1})] \leq\ & \mathcal{L}(\boldsymbol{\theta}_k) + (\gamma_u + \lambda\gamma_b)\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\| + \frac{\gamma_u + \lambda\gamma_b - \gamma_l}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2 \\ & - \mathbb{E}\left[\frac{\lambda}{M_h^2}\omega\left(\frac{M_h}{1 + M_h\eta_k}\right)\right],\end{aligned}$$

Note the second derivative ω'' of ω: $\omega''(t) = 1/(1+t)^2$. By the convexity of ω and using Jensen's inequality, also recalling unbiasedness of the derivatives,

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{k+1})] &\leq \mathcal{L}(\boldsymbol{\theta}_k) + \frac{\gamma_u + \lambda\gamma_b}{\sqrt{\gamma_a}(1 + M_h\eta_k)} + \frac{\gamma_u + \lambda\gamma_b}{2\gamma_a(1 + M_h\eta_k)^2} - \frac{\gamma_l}{2\gamma_a}\omega''(M_h\eta_k) \\ &\quad - \frac{\lambda}{M_h^2}\omega\left(\frac{M_h}{1 + M_h\eta_k}\right) \\ &\leq \mathcal{L}(\boldsymbol{\theta}_k) - \left[\frac{\lambda}{M_h^2}\omega\left(\frac{M_h}{1 + M_h\eta_k}\right) + \frac{\gamma_l}{2\gamma_a}\omega''(M_h\eta_k) - \frac{2(\gamma_u + \lambda\gamma_b)}{\sqrt{\gamma_a}}\right].\end{aligned}$$

In the above, we used the bounds of the norm in (B9), and again used the choice $\alpha = (\beta\tilde{\beta})^{-1}(\gamma_a)^{1/2}(K + \lambda\gamma_a)$.

To proceed, let us make a simple remark that is not explicitly stated in Remark 4: For any $\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1} \in \mathcal{N}_\epsilon(\boldsymbol{\theta}^*)$, we have

$$\begin{aligned}\left\|\boldsymbol{H}_g(\boldsymbol{\theta}_{k+1}) - \boldsymbol{H}_g(\boldsymbol{\theta}_k)\right\| &\leq \gamma_g \left\|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\right\| \to 0 \text{ as } k \to \infty,\\ \left\|\boldsymbol{H}_h(\boldsymbol{\theta}_{k+1}) - \boldsymbol{H}_h(\boldsymbol{\theta}_k)\right\| &\leq \gamma_h \left\|\boldsymbol{\theta_1} - \boldsymbol{\theta_2}\right\| \to 0 \text{ as } k \to \infty.\end{aligned}$$

Now, recall the proposed update step (21):

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \frac{\alpha}{1 + M_h\eta_k}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\left(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)^{-1}\boldsymbol{e}.$$

Then the above remark allows us to perform the following operation: Subtract $\boldsymbol{\theta}^*$ from both sides and pre-multiply by $\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_{k+1}) \approx \boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k) = \boldsymbol{H}_h^{1/2}$, we get the recursion

$$\begin{aligned}&\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*)\\ &\quad= \boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) - \frac{\alpha}{1 + M_h\eta_k}\boldsymbol{H}_h^{1/2}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k)\left(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k)\right)^{-1}\boldsymbol{e}(\boldsymbol{\theta}_k),\\ &\left\|\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*)\right\|\\ &\quad= \left\|\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) - \frac{\alpha}{1 + M_h\eta_k}\boldsymbol{H}_h^{1/2}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k)\left(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k)\right)^{-1}\boldsymbol{e}(\boldsymbol{\theta}_k)\right\|\\ &\quad\leq \left\|\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)\right\| + \frac{\alpha}{1 + M_h\eta_k}\left\|\boldsymbol{H}_h^{1/2}\right\|\left\|\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\left(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\right)^{-1}\boldsymbol{e}(\boldsymbol{\theta}_k)\right\|.\end{aligned}$$

Take expectation with respect to m on both sides conditioned on $\boldsymbol{\theta}_k$ and again consider unbiasedness of the derivatives. Further, recall the definition of the *local norm* $\|\cdot\|_{\boldsymbol{\theta}}$, and the bounds of $\boldsymbol{H}_h$, then

$$\begin{aligned}&\mathbb{E}\left\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^*\right\|_{\boldsymbol{\theta}_{k+1}}\\ &\quad\leq \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|_{\boldsymbol{\theta}_k} + \frac{\alpha}{\sqrt{\gamma_a}(1 + M_h\eta_k)}\left\|\left(\lambda\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{J}\boldsymbol{H}_h^{-1}\boldsymbol{J}^T(\boldsymbol{\theta}_k)\right)^{-1}\right\|\left\|\boldsymbol{H}_h^{-1}\boldsymbol{J}^T\boldsymbol{e}(\boldsymbol{\theta}_k)\right\|\\ &\quad\leq \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|_{\boldsymbol{\theta}_k} + \frac{\alpha\gamma_a}{\sqrt{\gamma_a}(1 + M_h\eta_k)(K + \lambda\gamma_a)}\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}(\boldsymbol{\theta}_k)\right\|\\ &\quad= \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|_{\boldsymbol{\theta}_k} + \frac{\gamma_a}{\beta\tilde{\beta}(1 + M_h\eta_k)}\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_g(\boldsymbol{\theta}_k) + \lambda\boldsymbol{H}_h^{-1}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\|\\ &\quad\leq \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|_{\boldsymbol{\theta}_k} + \frac{\gamma_a}{\beta\tilde{\beta}}\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_g(\boldsymbol{\theta}_k) + \lambda\boldsymbol{H}_h^{-1}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\|\\ &\quad\leq \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|_{\boldsymbol{\theta}_k} + \frac{\gamma_a}{\beta\tilde{\beta}}\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_g(\boldsymbol{\theta}_k)\right\| + \frac{\lambda\gamma_a}{\beta\tilde{\beta}}\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\|.\end{aligned}$$

By the mean value theorem and the first part of (SOSC),

$$\begin{aligned} \boldsymbol{g}_g(\boldsymbol{\theta}_k) &= \int_0^1 \boldsymbol{H}_g(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)d\tau \\ &= \boldsymbol{H}_g(\boldsymbol{\theta}^*)(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) + \int_0^1 \left(\boldsymbol{H}_g(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)) - \boldsymbol{H}_g(\boldsymbol{\theta}^*)\right)(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)d\tau, \end{aligned}$$

where $\boldsymbol{H}_g$ is the second derivative of g.

$$\begin{aligned} &\left\|\boldsymbol{g}_g(\boldsymbol{\theta}_k)\right\| = \left\|\boldsymbol{H}_g(\boldsymbol{\theta}^*)(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) + \int_0^1 \left(\boldsymbol{H}_g(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)) - \boldsymbol{H}_g(\boldsymbol{\theta}^*)\right)(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)d\tau\right\| \\ &\le \left\|\boldsymbol{H}_g(\boldsymbol{\theta}^*)\right\| \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| + \int_0^1 \left\|\boldsymbol{H}_g(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)) - \boldsymbol{H}_g(\boldsymbol{\theta}^*)\right\| \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| d\tau \\ &= \left\|\boldsymbol{H}_g(\boldsymbol{\theta}^*)\right\| \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| + \int_0^1 \tau\gamma_g \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2 d\tau \\ &= \left\|\boldsymbol{H}_g(\boldsymbol{\theta}^*)\right\| \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| + \frac{\gamma_g}{2} \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2 d\tau \\ &\le \gamma_u \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| + \frac{\gamma_g}{2} \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2. \end{aligned}$$

In the above steps, we have used Assumption 4 and the remarks that follow it. Further,

$$\begin{aligned} \left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_g(\boldsymbol{\theta}_k)\right\| &= \left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_g(\boldsymbol{\theta}_k)\right\| \\ &\le \left\|\boldsymbol{H}_h^{-1}\right\| \left\|\boldsymbol{g}_g(\boldsymbol{\theta}_k)\right\| \\ &\le \frac{1}{\gamma_a}\left(\gamma_u \left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| + \frac{\gamma_g}{2}\left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2\right) \\ &= \frac{\gamma_u}{\gamma_a}\left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\| + \frac{\gamma_g}{2\gamma_a}\left\|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*\right\|^2. \end{aligned}$$

Next, we analyze $\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\|$. We have

$$\begin{aligned} \left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\| &= \left\|\boldsymbol{H}_h^{-3/2}\boldsymbol{H}_h^{1/2}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\| \\ &\le \frac{1}{\gamma_a^{3/2}}\left\|\boldsymbol{H}_h^{1/2}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\| \\ &\overset{(SOSC)}{=} \frac{1}{\gamma_a^{3/2}}\left\|\boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k)\left(\boldsymbol{g}_h(\boldsymbol{\theta}_k) - \boldsymbol{g}_h(\boldsymbol{\theta}^*)\right)\right\|, \end{aligned}$$

and by the mean value theorem,

$$\left\|\boldsymbol{H}_h^{-1}\boldsymbol{g}_h(\boldsymbol{\theta}_k)\right\| = \frac{1}{\gamma_a^{3/2}}\left\|\int_0^1 \boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k)\boldsymbol{H}_h(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)d\tau\right\|$$

$$
\begin{aligned}
&\leq \frac{1}{\gamma_a^{3/2}} \left\| \boldsymbol{H}_h^{1/2}(\boldsymbol{\theta}_k)(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \right\| \left\| \int_0^1 \boldsymbol{H}_h(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))d\tau \right\| \\
&= \frac{1}{\gamma_a^{3/2}} \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_k} \left\| \int_0^1 \boldsymbol{H}_h(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))d\tau \right\| \\
&\leq \frac{1}{\gamma_a^{3/2}} \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_k} \left\| \int_0^1 \boldsymbol{H}_h^{-1/2}(\boldsymbol{\theta}_k)\boldsymbol{H}_h(\boldsymbol{\theta}^* + \tau(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*))\boldsymbol{H}_h^{-1/2}(\boldsymbol{\theta}_k)d\tau \right\| \\
&\overset{Lemma 8}{\leq} \frac{\left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_k}}{\gamma_a^{3/2}\left(1 - M_h \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_k}\right)}.
\end{aligned}
$$

In the above, we have used the fact $\boldsymbol{\theta}_0 \in \mathcal{N}_{M_h^{-1}}(\boldsymbol{\theta}^*) \implies \boldsymbol{\theta}_k \in \mathcal{N}_{M_h^{-1}}(\boldsymbol{\theta}^*)$ for all $\boldsymbol{\theta}_k$ generated by the process (21).

Combining the above results, we have

$$
\begin{aligned}
\mathbb{E} \left\| \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_{k+1}} \leq &\left[1 + \frac{\lambda \gamma_a^{-1/2}}{\beta\tilde{\beta}\left(1 - M_h \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_k}\right)} \right] \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|_{\boldsymbol{\theta}_k} \\
&+ \frac{\gamma_u}{\beta\tilde{\beta}} \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\| + \frac{\gamma_g}{2} \left\| \boldsymbol{\theta}_k - \boldsymbol{\theta}^* \right\|^2.
\end{aligned}
$$

□

References

1. Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Stat. 400–407 (1951)
2. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010, pp. 177–186. Springer, USA (2010)
3. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research **12**(7) (2011)
4. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
5. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
6. Johnson, R., Zhang, T.: Accelerating stochastic gradient descent using predictive variance reduction. Adv. Neural. Inf. Process. Syst. **26**, 315–323 (2013)
7. Becker, S., le Cun, Y.: Improving the convergence of back-propagation learning with second order methods (1988)
8. Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Math. Program. **45**(1), 503–528 (1989)
9. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Trans. Neural Netw. **5**(6), 989–993 (1994)
10. Amari, S.-I.: Natural gradient works efficiently in learning. Neural Comput. **10**(2), 251–276 (1998)
11. Martens, J., et al.: Deep learning via hessian-free optimization. In: ICML, vol. 27, pp. 735–742 (2010)
12. Pascanu, R., Bengio, Y.: Revisiting natural gradient for deep networks. arXiv preprint arXiv:1301.3584 (2013)
13. Martens, J., Grosse, R.: Optimizing neural networks with Kronecker-factored approximate curvature. In: International Conference on Machine Learning, pp. 2408–2417 (2015). PMLR
14. Nesterov, Y., et al.: Lectures on Convex Optimization, vol. 137. Springer, Switzerland (2018)
15. Byrd, R.H., Chin, G.M., Neveitt, W., Nocedal, J.: On the use of stochastic hessian information in optimization methods for machine learning. SIAM J. Optim. **21**(3), 977–995 (2011)

16. Erdogdu, M.A., Montanari, A.: Convergence rates of sub-sampled newton methods. arXiv preprint arXiv:1508.02810 (2015)
17. Cai, T., Gao, R., Hou, J., Chen, S., Wang, D., He, D., Zhang, Z., Wang, L.: Gram-gauss-newton method: Learning overparameterized neural networks for regression problems. arXiv preprint arXiv:1905.11675 (2019)
18. Zhang, G., Martens, J., Grosse, R.: Fast convergence of natural gradient descent for overparameterized neural networks. arXiv preprint arXiv:1905.10961 (2019)
19. Bernacchia, A., Lengyel, M., Hennequin, G.: Exact natural gradient in deep linear networks and application to the nonlinear case. NIPS (2019)
20. Karakida, R., Osawa, K.: Understanding approximate fisher information for fast convergence of natural gradient descent in wide neural networks. arXiv preprint arXiv:2010.00879 (2020)
21. Nesterov, Y., Polyak, B.T.: Cubic regularization of newton method and its global performance. Math. Program. **108**(1), 177–205 (2006)
22. Mishchenko, K.: Regularized newton method with global $O(1/k^2)$ convergence. arXiv preprint arXiv:2112.02089 (2021)
23. Marumo, N., Okuno, T., Takeda, A.: Constrained Levenberg-marquardt method with global complexity bound. arXiv preprint arXiv:2004.08259 (2020)
24. Doikov, N., Nesterov, Y.: Gradient regularization of newton method with Bregman distances. arXiv preprint arXiv:2112.02952 (2021)
25. Nesterov, Y., Nemirovskii, A.: Interior-point Polynomial Algorithms in Convex Programming. SIAM, Philadelphia (1994)
26. Bubeck, S., Sellke, M.: A universal law of robustness via isoperimetry. arXiv preprint arXiv:2105.12806 (2021)
27. Muthukumar, V., Vodrahalli, K., Subramanian, V., Sahai, A.: Harmless interpolation of noisy data in regression. IEEE J. Sel. Areas Inf. Theory **1**(1), 67–83 (2020)
28. Belkin, M., Hsu, D., Ma, S., Mandal, S.: Reconciling modern machine-learning practice and the classical bias-variance trade-off. Proc. Natl. Acad. Sci. **116**(32), 15849–15854 (2019)
29. Allen-Zhu, Z., Li, Y., Liang, Y.: Learning and generalization in overparameterized neural networks, going beyond two layers. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
30. Meng, S.Y., Vaswani, S., Laradji, I.H., Schmidt, M., Lacoste-Julien, S.: Fast and furious convergence: Stochastic second order methods under interpolation. In: International Conference on Artificial Intelligence and Statistics, pp. 1375–1386 (2020). PMLR
31. Ye, H., Luo, L., Zhang, Z.: Nesterov's acceleration for approximate newton. J. Mach. Learn. Res. **21**, 142–1 (2020)
32. Bieker, K., Gebken, B., Peitz, S.: On the treatment of optimization problems with l1 penalty terms via multiobjective continuation. arXiv preprint arXiv:2012.07483 (2020)
33. Patrinos, P., Stella, L., Bemporad, A.: Forward-backward truncated newton methods for convex composite optimization. arXiv preprint arXiv:1402.6655 (2014)
34. Schmidt, M., Fung, G., Rosales, R.: Fast optimization methods for l1 regularization: a comparative study and two new approaches. In: European Conference on Machine Learning, pp. 286–297. Springer (2007)
35. Schraudolph, N.N.: Fast curvature matrix-vector products for second-order gradient descent. Neural Comput. **14**(7), 1723–1738 (2002)
36. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. SIAM Rev. **60**(2), 223–311 (2018)
37. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Q. Appl. Math. **2**(2), 164–168 (1944)
38. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. J. Soc. Ind. Appl. Math. **11**(2), 431–441 (1963)
39. Searle, S.R.: Matrix Algebra Useful for Statistics. Wiley, Hoboken (1982)
40. Duncan, W.J.: Lxxviii. Some devices for the solution of large sets of simultaneous linear equations: with an appendix on the reciprocation of partitioned matrices. Lond. Edinb. Dublin Philos. Mag. J. Sci. **35**(249), 660–670 (1944)
41. Guttman, L.: Enlargement methods for computing the inverse matrix. Ann. Math. Stat. 336–343 (1946)
42. Higham, N.J.: Accuracy and Stability of Numerical Algorithms. SIAM, New York (2002)
43. Roosta-Khorasani, F., Mahoney, M.W.: Sub-sampled newton methods ii: Local convergence rates. arXiv preprint arXiv:1601.04738 (2016)

44. Du, S., Lee, J., Li, H., Wang, L., Zhai, X.: Gradient descent finds global minima of deep neural networks. In: International Conference on Machine Learning, pp. 1675–1685. PMLR (2019)
45. Du, S.S., Zhai, X., Poczos, B., Singh, A.: Gradient descent provably optimizes over-parameterized neural networks. arXiv preprint arXiv:1810.02054 (2018)
46. Sun, T., Tran-Dinh, Q.: Generalized self-concordant functions: a recipe for newton-type methods. Math. Program. **178**(1), 145–213 (2019)
47. Chang, C.-C., Lin, C.-J.: Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. (TIST) **2**(3), 1–27 (2011)
48. Romano, J.D., Le, T.T., La Cava, W., Gregg, J.T., Goldberg, D.J., Chakraborty, P., Ray, N.L., Himmelstein, D., Fu, W., Moore, J.H.: Pmlb v1.0: an open source dataset collection for benchmarking machine learning methods. arXiv preprint arXiv:2012.00058v2 (2021)
49. Dunn, P.K., Smyth, G.K.: Generalized Linear Models with Examples in R. Springer, New York (2018)
50. Ostrovskii, D.M., Bach, F.: Finite-sample analysis of m-estimators using self-concordance. Electron. J. Stat. **15**(1), 326–391 (2021)
51. Charbonnier, P., Blanc-Féraud, L., Aubert, G., Barlaud, M.: Deterministic edge-preserving regularization in computed imaging. IEEE Trans. Image Process. **6**(2), 298–311 (1997)
52. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2004)
53. Nocedal, J.: Updating quasi-newton matrices with limited storage. Math. Comput. **35**(151), 773–782 (1980)
54. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
55. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
56. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org (2015). https://www.tensorflow.org/
57. Fowkes, J.M., Gould, N.I., Farmer, C.L.: A branch and bound algorithm for the global optimization of hessian lipschitz continuous functions. J. Global Optim. **56**(4), 1791–1815 (2013)