Journal of Scientific Computing (2024) 100:17
https://doi.org/10.1007/s10915-024-02535-z

Gauss Newton Method for Solving Variational Problems of PDEs with Neural Network Discretizaitons

Wenrui Hao[1] · Qingguo Hong[2] · Xianlin Jin[3]

Received: 7 July 2023 / Revised: 24 January 2024 / Accepted: 12 March 2024 /
Published online: 3 June 2024

Abstract
The numerical solution of differential equations using machine learning-based approaches has gained significant popularity. Neural network-based discretization has emerged as a powerful tool for solving differential equations by parameterizing a set of functions. Various approaches, such as the deep Ritz method and physics-informed neural networks, have been developed for numerical solutions. Training algorithms, including gradient descent and greedy algorithms, have been proposed to solve the resulting optimization problems. In this paper, we focus on the variational formulation of the problem and propose a Gauss–Newton method for computing the numerical solution. We provide a comprehensive analysis of the superlinear convergence properties of this method, along with a discussion on semi-regular zeros of the vanishing gradient. Numerical examples are presented to demonstrate the efficiency of the proposed Gauss–Newton method.

Keywords Partial differential equations · Neural network discretization · Variational form · Gauss–Newton method · Convergence analysis

1 Introduction

The use of machine learning-based approaches in the computational mathematics community has witnessed significant growth in recent years, particularly in the numerical solution of differential equations. Neural network-based discretization has emerged as a revolutionary tool for solving differential equations [10, 15, 32] and for discovering the underlying physics from experimental data [26]. This approach has been successfully applied to a wide range of practical problems with remarkable success [5, 13, 17, 24]. One of the key advantages of this approach is that neural networks can alleviate, and in some cases overcome, the curse of dimensionality associated with high-dimensional problems [15, 18, 21]. This can be attributed to the dimension-independent approximation properties of neural networks [20],

✉ Wenrui Hao
wxh64@psu.edu

1 Department of Mathematics, Pennsylvania State University, State College, USA

2 Department of Mathematics and Statistics, Missouri University of Science and Technology, Rolla, USA

3 School of Mathematical Sciences, Peking University, Beijing, China

which have been compared to traditional methods such as finite element methods (FEMs) and other approximation techniques in the field of approximation theory [22, 27, 30, 34].

Discretizing PDEs through neural networks involves parameterizing a set of functions aimed at solving these PDEs. To illustrate this concept, we consider the following Laplace's equation:

$$\begin{cases} -\Delta v = f(v) & \text{in } \Omega \\ \frac{\partial v}{\partial n} = 0 & \text{on } \partial\Omega \end{cases}$$

Here, $\Omega \subset \mathbb{R}^d$ and $\partial\Omega$ is the boundary of the domain. The associated energy functional is expressed as:

$$\min_w J(w) = \int_\Omega |\nabla w|^2 - G(w),\ dx$$

where $G'(w) = f(w)$. Three primary approaches exist for solving numerical solutions based on the neural network discretization:

- *Variational energy minimization (deep Ritz method):* The first approach, known as the deep Ritz method [35], involves minimizing the variational energy:

$$\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \int_\Omega |\nabla u(x;\boldsymbol{\theta})|^2 - G(u(x;\boldsymbol{\theta}))\, dx$$

- L_2 *Residual minimization* The second approach, widely applied in physics-informed neural networks (PINNs) [26], minimizes the L_2 residual of PDEs and boundary conditions:

$$\min_{\boldsymbol{\theta}} \sum_i \|\Delta u(x_i;\boldsymbol{\theta}) + f(u(x_i;\boldsymbol{\theta}))\|^2 + \sum_j \left\| \frac{\partial u(x_j;\boldsymbol{\theta})}{\partial n} \right\|^2$$

- *System of nonlinear equations* The third approach involves solving a discretized system of nonlinear equations [8]:

$$\boldsymbol{F}(\boldsymbol{\theta}) = \begin{cases} \Delta u(x_i;\boldsymbol{\theta}) + f(u(x_i;\boldsymbol{\theta})), & i = 1, \cdots, N \\ \frac{\partial u(x_j;\boldsymbol{\theta})}{\partial \mathrm{n}}, & j = 1, \cdots, n \end{cases} = \mathbf{0}$$

  In this context, the second approach is related to the third approach and can be expressed as $\min_{\boldsymbol{\theta}} \|\boldsymbol{F}(\boldsymbol{\theta})\|_2^2$.

Researchers have made efforts to bound the errors associated with these approaches. For instance, studies have shown that gradient descent applied to a wide enough network can reach a global minimum [1, 2, 37]. The convergence of stochastic gradient descent (SGD) and Adam optimizer [19] has also been analyzed in Fourier space, revealing that the error converges more rapidly in the lowest frequency modes. This observation is known as the frequency principle or spectral bias of neural network training [25]. Furthermore, a novel greedy training algorithm has been devised for shallow neural networks in order to numerically determine their theoretical convergence rate [28].

In practice, Adam or SGD are commonly used to solve the resulting optimization problems in both variational and L2-minimization approaches. Additionally, randomized Newton's method has been developed to achieve faster local convergence by solving the system of nonlinear equations [8]. Gauss–Newton method has also been employed to speed up the computation of the L2-minimization approach [6, 23]. However, these training algorithms cannot be directly applied to the variational form of the problem. In this paper, we will develop a Gauss–Newton method for the variational form and provide a convergence analysis for this proposed method.

The remaining sections of the paper are organized as follows: In Sect. 2, we introduce the problem setup and discuss the class of elliptic PDEs we will be solving. Section 3 provides an overview of Gauss–Newton methods for solving PDEs in both variational and L2-minimization forms. The property of local minimizes, namely, semi-regular zeros of the vanishing gradient, is discussed in Sect. 4. In Sect. 5, we present a comprehensive analysis of the convergence properties of the Gauss–Newton method for the variational form, as well as its variant, the random Gauss–Newton method. Several numerical examples are presented in Sect. 6 to illustrate the efficiency of the proposed Gauss–Newton method. Finally, in Sect. 7, we conclude the paper, by summarizing the key findings and discussing potential avenues for future research.

2 Problem Setup

We consider the following second-order elliptic equation

$$\mathcal{L}v = f(v) \text{ in } \Omega, \tag{2.1}$$

where $\Omega \in \mathbb{R}^d$ is an open subset, $d \geq 1$. Here $\mathcal{L}$ is the second-order elliptic operator defined by

$$\mathcal{L} = -\sum_{i=1}^{d}\sum_{j=1}^{d} a_{i,j}\frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j} + \sum_{k=1}^{d} b_k \frac{\partial}{\partial x_k} + c. \tag{2.2}$$

To ensure the existence and uniqueness of the weak solution for problem (2.1), certain assumptions need to be made on the operator $\mathcal{L}$. Specifically, the following conditions should be satisfied [11]:

$$a_{i,j} \in L^{\infty}(\Omega), \quad 1 \leq i, j \leq d, \tag{2.3}$$

which means that the coefficients of the operator are bounded in $L^{\infty}(\Omega)$.
Moreover, there should exist positive constants λ and Λ such that

$$\lambda|\boldsymbol{\xi}|^2 \leq \sum_{i=1}^{d}\sum_{j=1}^{d} a_{i,j}\xi_i\xi_j \leq \Lambda|\boldsymbol{\xi}|^2, \quad \forall \boldsymbol{\xi} \in \mathbb{R}^d, \quad \forall x \in \Omega. \tag{2.4}$$

This means that the operator is uniformly elliptic, i.e., it satisfies a strong ellipticity condition, with ellipticity constants λ and Λ. These assumptions ensure that the weak solution to the problem (2.1) exists and is unique. To simplify notation, we consider the following second-order partial differential equation:

$$-a\Delta v(x) + cv(x) = f(x), \text{ in } \Omega \tag{2.5}$$

$$\frac{\partial v}{\partial n} = 0, \text{ on } \partial\Omega \tag{2.6}$$

where $a > 0$ and $c \geq 0$ are constants such that the conditions (2.3–2.4) are satisfied. It should be noted that all the results presented in this paper can be extended to the more general form of the operator L defined in (2.2), with Dirichlet boundary conditions.

We define the admissible set V on Ω as $V = H^1(\Omega)$ and transform the PDE (2.5) into an energy minimization problem, given by:

$$\min_{w \in V} \mathcal{J}(w) = \int_\Omega \left(\frac{a}{2}|\nabla w(x)|^2 + \frac{c}{2}w(x)^2 - f(x)w(x\right) dx. \tag{2.7}$$

Here, $\mathcal{J}(w)$ is the energy functional, and the minimizer $v = \arg\min_{w \in V} \mathcal{J}(w)$ of (2.7) satisfies the PDE (2.5). It is important to note that minimizing (2.7) is equivalent to solving the PDE (2.5).

The minimization problem can be solved using various learning algorithms, such as the Deep Ritz method [35], which utilizes deep neural networks (DNNs). In practice, the energy integral in (2.7) can be computed using numerical quadrature methods, such as the Gauss quadrature and the Monte Carlo method [14].

3 Gauss–Newton Method for the Variational Problem

In this section, without loss of generality, we will introduce the Gauss–Newton method for solving the energy minimization problem (2.7) with $a = 1$ and $c = 1$. Let $u(x, \theta)$ be a learning model to approximate the exact solution $w(x)$, for example, $u(x, \theta)$ can be the finite element function or the neural network function. More specifically, we consider a neural network discretization, consisting of a sequence of fully connected layers from the input $x \in \mathbb{R}^d$ to the output $u \in \mathbb{R}$. Mathematically, a neural network with J hidden layers can be expressed as follows:

$$u(x;\theta) = W_J h_{J-1} + b_J, \quad h_i = \sigma(W_i h_{i-1} + b_i), \quad i \in \{1, \ldots, J-1\}, \quad \text{and } h_0 = x, \tag{3.1}$$

where $W_i \in \mathbb{R}^{d_i \times d_{i-1}}$ is the weight, $b_i \in \mathbb{R}^{d_i}$ is the bias, d_i is the width of the i-th hidden layer, and σ is the activation function (e.g., ReLU or the sigmoid activation functions). For simplicity, we denote $m = d_1 + d_2 + \cdots + d_{J-1}$ and all the parameters of the neural network, including weights and biases, as θ. Then we define deep finite neuron functions with the activation function $\sigma = \text{ReLU}^k$ as DNN_J with J layers. In this paper, we will focus on a smaller admissible set, i.e., the DNN function space, where the loss function takes the form:

$$L(\theta) = \int_\Omega \frac{1}{2}|\nabla u(x,\theta)|^2 + \frac{1}{2}u(x,\theta)^2 - f(x)u(x,\theta)dx. \tag{3.2}$$

The gradient of $L(\theta)$ is

$$\nabla_\theta L(\theta) = \int_\Omega \nabla_\theta \nabla u(x,\theta)\nabla u(x,\theta) + u(x,\theta)\nabla_\theta u(x,\theta) - f(x)\nabla_\theta u(x,\theta)dx \tag{3.3}$$

The Hessian of $L(\theta)$ is as follows:

$$\mathbf{H}(\theta) = \underbrace{\int_\Omega \nabla_\theta \nabla u(x,\theta) \cdot \nabla_\theta \nabla u(x,\theta)^T + \nabla_\theta u(x,\theta) \cdot \nabla_\theta u(x,\theta)^T dx}_{\boldsymbol{J}(\theta)} \tag{3.4}$$

$$+ \underbrace{\int_\Omega \nabla_\theta^2 \nabla u(x,\theta) \cdot \nabla u(x,\theta) + u(x,\theta)\nabla_\theta^2 u(x,\theta) - f(x)\nabla_\theta^2 u(x,\theta)dx}_{\boldsymbol{Q}(\theta)}, \tag{3.5}$$

where $\boldsymbol{J}(\theta)$ and $\boldsymbol{Q}(\theta)$ denote terms involving the first- and second-order derivative information of θ, respectively.

Then we estimate the second derivative matrix by the following lemma [3, 29, 31, 34].

Lemma 1 *For $\forall\epsilon > 0$, there exist $\delta > 0$, $J \in \mathbb{N}^+$ and $m \in \mathbb{N}^+$, such that $\theta \in \mathbb{R}^m$, $DNN_J \subset H^1(\Omega)$, and for $\|\theta - \theta^*\| < \delta$, $\theta^* = \arg\min L(\theta)$, if $D^2_\theta u(x,\theta) \in H^1(\Omega)$, then it holds $\|\boldsymbol{Q}(\theta)\| < \epsilon$.*

Proof First, we can rewrite $\boldsymbol{Q}(\theta)$ using integration by parts as

$$\boldsymbol{Q}(\theta) = \int_\Omega \nabla^2_\theta \nabla u(x,\theta)\cdot\nabla u(x,\theta) + u(x,\theta)\nabla^2_\theta u(x,\theta) - f(x)\nabla^2_\theta u(x,\theta)dx \tag{3.6}$$

$$= \int_\Omega \nabla^2_\theta u(x,\theta)(-\Delta u(x,\theta) + u(x,\theta) - f(x))dx + \int_{\partial\Omega} \nabla^2_\theta u(x,\theta)\frac{\partial u(x,\theta)}{\partial n}ds. \tag{3.7}$$

Given that the true solution $v(x)$ satisfies:

$$-\Delta v(x) + v(x) = f(x), \text{ in } \Omega \tag{3.8}$$

$$\frac{\partial v}{\partial n} = 0, \text{ on } \partial\Omega. \tag{3.9}$$

We can rewrite $\boldsymbol{Q}(\theta)$ as follows:

$$\boldsymbol{Q}(\theta) = \int_\Omega \nabla^2_\theta u(x,\theta)(-\Delta u(x,\theta) + u(x,\theta) - f(x))dx + \int_{\partial\Omega} \nabla^2_\theta u(x,\theta)\frac{\partial u(x,\theta)}{\partial n}ds \tag{3.10}$$

$$= \int_\Omega \nabla^2_\theta u(x,\theta)(-\Delta u(x,\theta) + u(x,\theta) + \Delta v(x) - v(x))dx \tag{3.11}$$

$$+ \int_{\partial\Omega} \nabla^2_\theta u(x,\theta)\left(\frac{\partial u(x,\theta)}{\partial n} - \frac{\partial v(x)}{\partial n}\right)ds, \tag{3.12}$$

which implies

$$\|\boldsymbol{Q}(\theta)\| \le \|\nabla^2_\theta u(x,\theta)\|_{H^1(\Omega)}\big(\|\Delta u(x,\theta) - \Delta v(x)\|_{H^{-1}(\Omega)} + \|u(x,\theta) - v(x)\|_{H^{-1}(\Omega)}\big) \tag{3.13}$$

$$+ \|\nabla^2_\theta u(x,\theta)\|_{H^{\frac{1}{2}}(\partial\Omega)}\left\|\frac{\partial u(x,\theta)}{\partial n} - \frac{\partial v(x)}{\partial n}\right\|_{H^{-\frac{1}{2}}(\partial\Omega)}. \tag{3.14}$$

Hence, by trace theorem, we have

$$\|\boldsymbol{Q}(\theta)\| \le C\|\nabla^2_\theta u(x,\theta)\|_{H^1(\Omega)}\|u(x,\theta) - v(x)\|_{H^1(\Omega)}. \tag{3.15}$$

Given that $\theta^* = \arg\min L(\theta)$ and $u(x,\theta^*)$ is the approximation solution for $v(x)$, we have the following error estimates [34]:

$$\|u(x,\theta^*) - v(x)\|_{H^1(\Omega)} \le \inf_{u(x,\theta)\in DNN_J} \|u(x,\theta) - v(x)\|_{H^1(\Omega)}.$$

This leads to

$$\begin{aligned}\|\boldsymbol{Q}(\theta^*)\| &\le C\|\nabla^2_\theta u(x,\theta^*)\|_{H^1(\Omega)}\|u(x,\theta^*) - v(x)\|_{H^1(\Omega)}\\ &\le C\|\nabla^2_\theta u(x,\theta^*)\|_{H^1(\Omega)}\inf_{u(x,\theta)\in DNN_J}\|u(x,\theta) - v(x)\|_{H^1(\Omega)}.\end{aligned}$$

Furthermore, using the approximation results of neural networks from [3, 29, 31] and considering the assumption $D_\theta^2 u(x,\theta^*) \in H^1(\Omega)$, we have for any $\epsilon > 0$, there exist $J \in \mathbb{N}^+$ and $m \in \mathbb{N}^+$, such that $\theta \in \mathbb{R}^m$, $DNN_J \subset H^1(\Omega)$ and

$$\inf_{u(x,\theta)\in DNN_J} \|u(x,\theta) - v(x)\|_{H^1(\Omega)} < \frac{\epsilon}{2C\|\nabla_\theta^2 u(x,\theta^*)\|_{H^1(\Omega)}},$$

implying $\|\boldsymbol{Q}(\theta^*)\| < \frac{\epsilon}{2}$. Finally, leveraging the continuity of $|\boldsymbol{Q}(\theta)|$ concerning θ, we establish that for every $\epsilon > 0$, there exists $\delta > 0$ such that for $\|\theta - \theta^*\| < \delta$, we have $|\boldsymbol{Q}(\theta) - \boldsymbol{Q}(\theta^*)| < \frac{\epsilon}{2}$. Consequently, the desired result is attained through the application of the triangle inequality. $\square$

Hence, it is reasonable to consider the first-order approximation of the Hessian, i.e., $\boldsymbol{J}(\theta) \approx \mathbf{H}(\theta)$. The Gauss–Newton method for the variational problem is then given by

$$\theta_{k+1} = \theta_k - \boldsymbol{J}(\theta_k)^\dagger \nabla_\theta L(\theta_k), \quad k = 0, 1, 2, \cdots. \tag{3.16}$$

3.1 Gauss–Newton Method for Solving the L2 Minimization Problem

In this subsection, we will review the Gauss–Newton method for solving the L2 minimization problem, namely,

$$\min_\theta \frac{1}{2}\|\boldsymbol{F}(\theta)\|_2^2, \tag{3.17}$$

where

$$\boldsymbol{F}(\theta) = \begin{pmatrix} -\Delta u(x_1,\theta) + u(x_1,\theta) - f(x_1) \\ \vdots \\ -\Delta u(x_N,\theta) + u(x_N,\theta) - f(x_N) \\ \nabla u(x_1^b,\theta)\cdot \boldsymbol{n} \\ \vdots \\ \nabla u(x_n^b,\theta)\cdot \boldsymbol{n} \end{pmatrix}, \tag{3.18}$$

with collocation points $\{x_1, x_2, \cdots, x_N\} \subset \Omega$ and $\{x_1^b, x_2^b, \cdots, x_n^b\} \subset \partial\Omega$. Here, $\boldsymbol{n}$ is the outer normal unit vector of $\partial\Omega$, and $\theta = \{\theta_1, \theta_2, \cdots, \theta_m\}$ represents the parameters in the learning model $u(x,\theta)$. Therefore, an optimal choice of θ is computed by the Gauss–Newton method as

$$\theta_{k+1} = \theta_k - (\mathbf{JF}(\theta_k))^\dagger \boldsymbol{F}(\theta_k), \quad k = 0, 1, 2, \cdots \tag{3.19}$$

where † denotes the Moore–Penrose inverse and $\mathbf{JF}(\theta)$ is the Jacobi matrix defined as

$$\mathbf{JF}(\theta) = \begin{bmatrix} -\nabla_\theta \Delta u(\boldsymbol{x},\theta)^T + \nabla_\theta u(\boldsymbol{x},\theta)^T \\ \left(\nabla_\theta \nabla u(\boldsymbol{x}^b,\theta)\cdot \boldsymbol{n}\right)^T \end{bmatrix} \in \mathbb{R}^{(N+n)\times m} \text{ and } \nabla_\theta \nabla = \begin{bmatrix} \partial_{x_1}\partial_{\theta_1} & \cdots & \partial_{x_d}\partial_{\theta_1} \\ \vdots & \ddots & \vdots \\ \partial_{x_1}\partial_{\theta_m} & \cdots & \partial_{x_d}\partial_{\theta_m} \end{bmatrix}. \tag{3.20}$$

3.2 The Consistency Between Gauss–Newton Methods for L2 Minimization and Variational Problems

By applying the divergence theorem to $\boldsymbol{J}(\theta)$, we obtain:

$$\boldsymbol{J}(\theta) = \int_\Omega \nabla_\theta u(x,\theta) \cdot (-\nabla_\theta \Delta u(x,\theta) + \nabla_\theta u(x,\theta))^T \, dx \\ + \int_{\partial\Omega} \nabla_\theta u(x,\theta) \cdot \frac{\partial \nabla_\theta u(x,\theta)}{\partial \boldsymbol{n}}^T \, dS. \tag{3.21}$$

If we compute all integrals using numerical methods, e.g., the Gaussian quadrature rule, we can derive the condition for the consistency of the Gauss–Newton method in (3.16) with the Gauss–Newton method for the L2 minimization problem in (3.19). Denote the grid points in the domain Ω as $\boldsymbol{x} = (x_1, x_2, \cdots, x_N)^T$ and the corresponding weights as $\boldsymbol{w} = (w_1, w_2, \ldots, w_N)^T$. Then, we can write the first part of (3.21) as:

$$\int_\Omega \nabla_\theta u(x,\theta) \cdot (-\nabla_\theta \Delta u(x,\theta) + \nabla_\theta u(x,\theta))^T \, dx \\ = \sum_{i=1}^N w_i \nabla_\theta u(x_i,\theta) \cdot (-\nabla_\theta \Delta u(x_i,\theta) + \nabla_\theta u(x_i,\theta))^T . \tag{3.22}$$

Similarly, with grid points on the boundary $\boldsymbol{x^b} = (x_1^b, x_2^b, \cdots, x_n^b)^T$ and weights $\boldsymbol{w^b} = (w_1^b, w_2^b, \cdots, w_n^b)^T$, we can write the second part of (3.21) as:

$$\int_{\partial\Omega} \nabla_\theta u(x,\theta) \cdot \frac{\partial \nabla_\theta u(x,\theta)}{\partial \boldsymbol{n}}^T \, dS \\ = \sum_{j=1}^n w_j^b \nabla_\theta u(x_j,\theta) \cdot \frac{\partial \nabla_\theta u(x_j,\theta)}{\partial \boldsymbol{n}}^T . \tag{3.23}$$

Thus, we have

$$\boldsymbol{J}(\theta) = \boldsymbol{G} \cdot \mathbf{JF}(\boldsymbol{\theta}) \text{ where } \boldsymbol{G} \\ = \left[w_1 \nabla_\theta u(x_1,\theta) \cdots w_N \nabla_\theta u(x_N,\theta) \; w_1^b \nabla_\theta u(x_1^b,\theta) \cdots w_n^b \nabla_\theta u(x_n^b,\theta) \right] \tag{3.24}$$

Similarly, we can rewrite the gradient defined in (3.3) as

$$\nabla_\theta L(\theta) = \int_\Omega \nabla_\theta u(x,\theta) \, (-\Delta u(x,\theta) + u(x,\theta) - f(x)) \, dx \\ + \int_{\partial\Omega} \nabla_\theta u(x,\theta) \frac{\partial \nabla_\theta u(x,\theta)}{\partial \boldsymbol{n}} dS = \boldsymbol{G} \cdot \boldsymbol{F}(\boldsymbol{x},\theta) \tag{3.25}$$

If $\boldsymbol{G}$ has linearly independent columns and $\mathbf{JF}(\theta)$ has linearly independent rows, then we have $(\boldsymbol{G} \cdot \mathbf{JF}(\theta))^\dagger = (\mathbf{JF}(\theta))^\dagger \cdot \boldsymbol{G}^\dagger$ and $\boldsymbol{G}^\dagger \boldsymbol{G} = \boldsymbol{I} \in \mathbb{R}^{(N+n)\times(N+n)}$ [12]. This is possible if the number of grid points $N + n$ is less than or equal to the number of parameters θ. In this case, the Gauss–Newton method that we proposed for the variational problem (3.16) is identical to the Gauss–Newton method for the L2 minimization (3.19). More specifically, we have

$$(\boldsymbol{J}(\theta))^\dagger \cdot \nabla_\theta L(\theta) = (\boldsymbol{G} \cdot \mathbf{JF}(\theta))^\dagger \cdot \boldsymbol{G} \cdot \boldsymbol{F}(\boldsymbol{x},\theta) \\ = (\mathbf{JF}(\theta))^\dagger \cdot (\boldsymbol{G}^\dagger \boldsymbol{G}) \cdot \boldsymbol{F}(\boldsymbol{x},\theta) = (\mathbf{JF}(\theta))^\dagger \cdot \boldsymbol{F}(\boldsymbol{x},\theta). \tag{3.26}$$

4 Semiregular Zeros of $\nabla L(\boldsymbol{\theta}) = 0$

We will consider the semiregular zeros of $\nabla L(\theta^*) = 0$ by the following two definitions.

Definition 1 (Dimension of a Zero [4, 36]) Let θ^* be a zero of a smooth mapping $\nabla L : \Omega \subset \mathbb{R}^m \to \mathbb{R}^{N+n}$. If there is an open neighborhood $\Omega_z \subset \Omega$ of θ_* in $\mathbb{R}^m$ such that $\Omega_z \cap (\nabla L)^{-1}(\mathbf{0}) = \phi(\Lambda)$ where $\mathbf{z} \mapsto \phi(\mathbf{z})$ is a differentiable injective mapping defined in a connected open set Λ in $\mathbb{R}^k$ for a certain $k > 0$ with $\phi(\mathbf{z}_*) = \theta_*$ and $rank\,(\phi_{\mathbf{z}}(\mathbf{z}_*)) = k$, then the dimension of θ_* as a zero of ∇L is defined as

$$dim_{\nabla L}(\theta_*) := dim\,(Range\,(\phi_{\mathbf{z}}(\mathbf{z}_*))) \equiv rank\,(\phi_{\mathbf{z}}(\mathbf{z}_*)) = k$$

Definition 2 (Semiregular Zero [4, 36]) A zero $\theta^* \in \mathbb{R}^m$ of a smooth mapping $\theta \mapsto \nabla L(\theta)$ is semiregular if $dim_{\nabla L}(\theta^*)$ is well-defined and identical to *nullity* $(\mathbf{H}(\theta^*))$. Namely

$$dim_{\nabla L}\left(\theta^*\right) + rank\left(\mathbf{H}\left(\theta^*\right)\right) = m.$$

We have the following properties of semiregular zeros in our setup.

Lemma 2 *Let $\theta \mapsto \nabla L(\theta)$ be a smooth mapping with a semiregular zero θ^* and $Range(Q(\theta^*)) \subseteq Range(J(\theta^*))$. Then there is an open neighborhood Ω_* of θ^*, such that, for any $\hat{\theta} \in \Omega_*$, the equality $\mathbf{J}(\hat{\theta})^\dagger \nabla L(\hat{\theta}) = \mathbf{0}$ holds if and only if $\hat{\theta}$ is a semiregular zero of ∇L in the same branch of θ^*.*

Proof We follow the proof of Lemma 4 in [36] and note that $|\mathbf{Q}| \le \epsilon$ for any small ϵ. First, we claim that there exists a neighborhood Ω_1 of θ^* such that for every $\hat{\theta} \in \Omega_1$, we have $\nabla L(\hat{\theta}) = \mathbf{0}$ and $\mathbf{J}(\hat{\theta})^\dagger \nabla L(\hat{\theta}) = \mathbf{0}$. Assume that this assertion is false. Then there exists a sequence $\left\{\theta_j\right\}_{j=1}^{\infty}$ converging to θ^* such that $\mathbf{J}\left(\theta_j\right)^\dagger \nabla L\left(\theta_j\right) = \mathbf{0}$ but $\nabla L\left(\theta_j\right) \neq \mathbf{0}$ for all $j = 1, 2, \ldots$. Let $\mathbf{z} \mapsto \phi(\mathbf{z})$ be the parameterization of the solution branch containing θ^* as defined in Definition 1, with $\phi(\mathbf{z}) = \theta^*$.

From Lemma 9, for any sufficiently large j, there exists a $\check{\theta} j \in \Omega \cap (\nabla L)^{-1}(\mathbf{0}) = \phi(\Delta)$ such that

$$\left\|\theta_j - \check{\theta}_j\right\|_2 = \min_{\mathbf{z}\in\Delta} \left\|\theta_j - \phi(\mathbf{z})\right\|_2 = \left\|\theta_j - \phi\left(\mathbf{z}_j\right)\right\|_2 \tag{4.1}$$

at a certain $\mathbf{z}_j$ with $\phi\left(\mathbf{z}_j\right) = \check{\mathbf{x}}_j$, implying

$$\phi_{\mathbf{z}}\left(\mathbf{z}_j\right)\phi_{\mathbf{z}}\left(\mathbf{z}_j\right)^\dagger \frac{\theta_j - \phi\left(\mathbf{z}_j\right)}{\left\|\theta_j - \phi\left(\mathbf{z}_j\right)\right\|_2} = \frac{\phi_{\mathbf{z}}\left(\mathbf{z}_j\right)}{\left\|\theta_j - \phi\left(\mathbf{z}_j\right)\right\|_2}\left(\phi_{\mathbf{z}}\left(\mathbf{z}_j\right)^\dagger\left(\theta_j - \phi\left(\mathbf{z}_j\right)\right)\right) = \mathbf{0}. \tag{4.2}$$

We claim that $\check{\theta}_j$ converges to θ^* as j approaches infinity. To see why, assume otherwise. Namely, suppose there exists an $\varepsilon > 0$ such that for any $N > 0$, there is a $j > N$ with $\left\|\check{\theta}_j - \theta^*\right\|_2 \ge 2\varepsilon$. However, we know that $\left\|\theta_j - \theta^*\right\|_2 < \varepsilon$ for all j larger than some fixed N. This implies that

$$\left\|\check{\theta}_j - \theta_j\right\| \ge \left\|\check{\theta}_j - \theta^*\right\| - \left\|\theta_* - \theta_j\right\| > \varepsilon > \left\|\theta_j - \theta_*\right\|_2$$

which contradicts (4.1). Therefore, we conclude that $\check{\theta}_j$ converges to θ^* as j approaches infinity.

Since $\nabla L\left(\theta_j\right) \neq \mathbf{0}$, we have $\theta_j \neq \check{\theta}_j$. Therefore, we can consider the unit vector $\mathbf{v}_j = \left(\theta_j - \check{\theta} j\right) / \left\|\theta_j - \check{\theta} j\right\|_2$ for each j. By compactness, there exists a subsequence $\mathbf{v}_{j_k}$ that converges to some unit vector $\mathbf{v}$. That is, $\lim_{k\to\infty} \mathbf{v}_{j_k} = \mathbf{v}$ for some unit vector $\mathbf{v}$. Thus

$$\begin{aligned} 0 &= \lim_{j\to\infty} \frac{\mathbf{J}(\theta_j)^\dagger \left(\nabla L\left(\check{\theta}_j\right) - \nabla L\left(\theta_j\right)\right)}{\left\|\theta_j - \check{\theta}_j\right\|_2} \\ &= \lim_{j\to\infty} \frac{\mathbf{J}(\theta_j)^\dagger \mathbf{H}\left(\theta_j\right)\left(\check{\theta}_j - \theta_j\right)}{\left\|\check{\theta}_j - \theta_j\right\|_2} = \mathbf{J}(\theta^*)^\dagger \mathbf{H}\left(\theta^*\right)\mathbf{v} \end{aligned} . \tag{4.3}$$

By the assumption $Range(Q(\theta^*)) \subseteq Range(J(\theta^*))$ and noting that $\mathbf{H} = \mathbf{J} + \mathbf{Q}$ and $\|\mathbf{Q}\| \le \epsilon_2$ for any small ϵ_2, we have $\mathbf{v} \in Kernel\left(\mathbf{H}\left(\theta^*\right)\right)$. As a result,

$$span\{\mathbf{v}\} \oplus Range\left(\phi_{\mathbf{z}}\left(\mathbf{z}_*\right)\right) \subset Kernel\left(\mathbf{H}\left(\theta^*\right)\right)$$

since $\mathbf{H}\left(\theta^*\right)\phi_{\mathbf{z}}\left(\mathbf{z}_*\right) = O$ due to $\nabla L(\phi(\mathbf{z})) \equiv \mathbf{0}$ in a neighborhood of $\mathbf{z}_*$. From the limit of (4.2) for $j \to \infty$, the vector $\mathbf{v}$ is orthogonal to $Range\left(\phi_{\mathbf{z}}\left(\mathbf{z}_*\right)\right)$ and thus

$$nullity\left(\mathbf{H}\left(\theta^*\right)\right) \ge rank\left(\phi_{\mathbf{z}}\left(\mathbf{z}_*\right)\right) + 1$$

which is a contradiction to the semiregularity of θ^*. For the special case where θ^* is an isolated semiregular zero of $\nabla L(\theta)$ with dimension 0, the above proof applies with $\check{\theta}_j = \theta_j$ so that (4.3) holds, implying a contradiction to $nullity\left(\mathbf{H}\left(\theta^*\right)\right) = 0$.

Lemma 11 implies that there exists a neighborhood Ω_2 of θ^* such that for every $\theta \in \Omega_2 \cap (\nabla L)^{-1}(\mathbf{0})$, θ is a semiregular zero of ∇L in the same branch as θ^*. Therefore, the lemma holds for $\Omega_* = \Omega_1 \cap \Omega_2$. □

Subsequently, we will provide justification for the semi-regular case within our setup, beginning with the finite element method. Let us consider the finite element space and define V_N as the set of functions that can be represented as a linear combination of the basis functions $\phi_i(x)$, where $a_i \in \mathbb{R}$ and $i = 1, \cdots, m$. Here, $\phi_i(x)$ is a frame defined according to the partition $\mathcal{T}_h$. Thus we can express V_N as

$$V_N = \left\{\sum_{i=1}^{m} a_i \phi_i(x)\right\}. \tag{4.4}$$

If $u(x, \theta) \in V_N$, we obtain a linear system $\nabla L(\theta) = A\theta - g$, where A is a square matrix and $\theta = (a_1, a_2, \cdots, a_m)$ represents the coefficients of the basis functions in V_N.

- Regular zero: if A is full rank, θ^* is unique and therefore an isolated and regular zero.
- Semiregular zero: if A is not full rank, then we denote $rank(A) = r$. We assume that $Kerl(A) = span\{\theta_1, \theta_2, \cdots, \theta_{m-r}\}$ with $\|\theta_i\| = 1$ and set

  $$\Delta = \{\theta : \|\theta - \theta^*\| < \delta\}.$$

  For any $\theta \in \Delta \cap (\nabla L)^{-1}(0)$, we have

  $$\theta = \theta^* + \delta_1\theta_1 + \cdots + \delta_{m-r}\theta_{m-r}$$

  with $\|\delta_i\| < \frac{\delta}{m-r}$ and $\mathbf{z} = (\delta_1, \delta_2, \cdots, \delta_{m-r})$. Then we can construct ϕ as $\phi(\mathbf{z}) = \theta = \theta^* + \delta_1\theta_1 + \cdots + \delta_{m-r}\theta_{m-r}$ and $\phi_{\mathbf{z}}(\mathbf{z}) = (\theta_1, \theta_2, \cdots, \theta_{m-r}) \in \mathcal{R}^{m\times(m-r)}$. Moreover,

  $$dim_{\nabla L}\left(\theta^*\right) + rank\left(\mathbf{H}\left(\theta^*\right)\right) = rank(\phi_z(\mathbf{z})) + rank(A) = m$$

which means that θ^* is a semirrgular zero.

Next, we proceed to explain the occurrence of semiregular zeros in neural networks with the $ReLU^k$ activation function. Let's consider a simple neural network in 1D with domain $\Omega = (-1, 1)$:

$$V_N^k = \left\{ \sum_{i=1}^{m} a_i \mathrm{ReLU}^k \left(w_i x + b_i\right), a_i \in \mathbb{R}, w_i \in \{-1, 1\}, b_i \in [-1-\delta, 1+\delta] \right\}. \quad (4.5)$$

Because of the scaling property of the ReLU function, we can rewrite V_N^k as

$$V_N^k = \left\{ \sum_{i=1}^{m} a_i \mathrm{ReLU}^k \left(x + b_i\right), a_i \in \mathbb{R}, b_i \in [-1-\delta, 1+\delta] \right\}. \quad (4.6)$$

We consider the following semi-regular zero cases:

- Let us consider a simple case where $u = a_1 \mathrm{ReLU}^k(x + b_1)$ and $L(\theta)$ is computed with two grid points x_1 and x_1^b satisfying $x_1 + b_1^* < 0$ and $B(x_1^b, \theta^*) = u'(x_1^b, \theta^*) = 0$. Then we can write

$$\boldsymbol{G} = \begin{bmatrix} w_1 \frac{\partial u(x_1,\theta)}{\partial a_1} & w_1^b \frac{\partial u(x_1^b,\theta)}{\partial a_1} \\ w_1 \frac{\partial u(x_1,\theta)}{\partial b_1} & w_1^b \frac{\partial u(x_1^b,\theta)}{\partial b_1} \end{bmatrix} = \begin{bmatrix} 0 & w_1^b \frac{\partial u(x_1^b,\theta)}{\partial a_1} \\ 0 & w_1^b \frac{\partial u(x_1^b,\theta)}{\partial b_1} \end{bmatrix}, \mathbf{JF} = \begin{bmatrix} 0 & 0 \\ \frac{\partial B(x_1^b,\theta)}{\partial a_1} & \frac{\partial B(x_1^b,\theta)}{\partial b_1} \end{bmatrix},$$

and

$$\nabla_\theta L(\theta) = \begin{bmatrix} 0 & w_1^b \frac{\partial u(x_1^b,\theta)}{\partial a_1} \\ 0 & w_1^b \frac{\partial u(x_1^b,\theta)}{\partial b_1} \end{bmatrix} \begin{bmatrix} F(x_1, \theta) \\ F(x_1^b, \theta) \end{bmatrix} = w_1^b \begin{bmatrix} \frac{\partial u(x_1^b,\theta)}{\partial a_1} F(x_1^b, \theta) \\ \frac{\partial u(x_1^b,\theta)}{\partial b_1} F(x_1^b, \theta) \end{bmatrix}.$$

Let us define

$$\theta = (a_1, b_1)^T, \quad \theta^* = (a_1^*, b_1^*)^T, \quad \theta_1 = (0, 1)^T$$

and

$$\phi(\mathbf{z}) = \theta^* + \delta_1 \theta_1 = \theta^* + \delta_1 \theta_1 = \theta^* + \delta_1 (0, 1)^T \quad \text{with} \quad \mathbf{z} = \delta_1.$$

Then, there exists a δ such that $\Delta = \{\theta : \|\theta - \theta^*\| < \delta\}$, which means b_1 is near b_1^*. We have $\Lambda = (-\delta, \delta)$ such that $\phi(\Lambda) = \Delta \cap (\nabla L)^{-1}(0)$ since if $\theta^* = (a_1^*, b_1^*)^T$ satisfies $\nabla L(\theta^*) = 0$ then for some δ, for any $\theta = (a_1^*, b_1)$ with $\|b_1 - b_1^*\| < \delta$ satisfies $\nabla L(\theta) = 0$, which implies $\Delta \cap (\nabla L)^{-1}(0) = \Delta$.
We can see that $\phi_z(\mathbf{z}) = (0, 1)^T$ and $\mathrm{rank}(\phi_z(\mathbf{z})) = 1$. Furthermore, we observe that

$$\boldsymbol{H}(\theta) = w_1^b \begin{bmatrix} \frac{\partial u(x_1^b,\theta)}{\partial a_1} F_{a_1}(x_1^b, \theta) & \frac{\partial^2 u(x_1^b,\theta)}{\partial a_1 \partial b_1} F(x_1^b, \theta) + \frac{\partial u(x_1^b,\theta)}{\partial a_1} F_{b_1}(x_1^b, \theta) \\ \frac{\partial^2 u(x_1^b,\theta)}{\partial b_1 \partial a_1} F(x_1^b, \theta) + \frac{\partial u(x_1^b,\theta)}{\partial b_1} F_{a_1}(x_1^b, \theta) & \frac{\partial^2 u(x_1^b,\theta)}{\partial b_1^2} F(x_1^b, \theta) + \frac{\partial u(x_1^b,\theta)}{\partial b_1} F_{b_1}(x_1^b, \theta) \end{bmatrix},$$

which implies that $\mathrm{rank}(\boldsymbol{H}(\theta^*)) = 1$ due to the fact that $F(x_1^b, \theta^*) = 0$. Therefore, we have $\mathrm{rank}(\phi_z(\mathbf{z})) + \mathrm{rank}(\mathbf{H}(\theta^*)) = 2$, which is the dimension of the domain of $\nabla_\theta L(\theta)$. This implies that in the simple case $u = a_1 \mathrm{ReLU}(x + b_1)$, θ^* is a semiregular zero of $\nabla_\theta L(\theta) = 0$.
- We next consider the general case of $m > 1$ and assume $\mathrm{rank}(\mathbf{G}(\theta^*)) = r < 2m$. By denoting

$$\theta = (a_1, a_2, \cdots, a_m, b_1, b_2, \cdots, b_m)^T, \quad \theta^* = (a_1^*, a_2^*, \cdots, a_m^*, b_1^*, b_2^*, \cdots, b_m^*),$$

we assume there exists $2m-r$ sample points x_i such that $x_i+b_j^* < 0$ for all $j = 1, \cdots, m$ and define

$$\begin{aligned}
\theta_1 &= (0, 0, \cdots, 0, 1, 0, \cdots, 0, 0, 0, \cdots, 0)^T,\\
\theta_2 &= (0, 0, \cdots, 0, 0, 1, \cdots, 0, 0, 0, \cdots, 0)^T,\\
&\cdots\\
\theta_{2m-r} &= (0, 0, \cdots, 0, 0, 0, \cdots, 0, 1, 0, \cdots, 0)^T.
\end{aligned}$$

We observe that if $\theta^* = (a_1^*, \cdots, a_m^*, b_1^*, \cdots, b_m^*)$ satisfies $\nabla L(\theta^*) = 0$, then b_i is a function of b_i^* and the remaining parameters $a_1^*, \ldots, a_m^*$, and hence, if $\theta = (a_1, a_2, \cdots, a_m, b_1, b_2, \cdots, b_m)^T$, then $\nabla L(\theta) = 0$ as well. This implies that there exist $\delta_1, \delta_2, \cdots, \delta_{2m-r}$ such that

$$\phi(\mathbf{z}) = \theta^* + \delta_1\theta_1 + \delta_2\theta_2 + \cdots + \delta_{2m-r}\theta_{2m-r} \text{ with } \mathbf{z} = (\delta_1, \delta_2, \cdots, \delta_{2m-r})$$

and $\Delta = \{\theta : \|\theta - \theta^*\| < \delta\}$ such that $\Delta \cap (\nabla L)^{-1}(0) = \Delta$. We define

$$\Lambda = (-\delta_1, \delta_1) \times (-\delta_2, \delta_2) \times \cdots \times (-\delta_{2m-r}, \delta_{2m-r}),$$

and have

$$\phi_z(\mathbf{z}) = (\theta_1, \theta_2, \cdots, \theta_{2m-r}).$$

Clearly, we have $\text{rank}(\phi_z(\mathbf{z})) = 2m - r$. If $F(x_i; \theta^*) = 0$ when $x_i + b_j^* > 0$ for $i = 1, \cdots, N+n$ and $j = 1, \cdots, m$, then we have $\text{rank}(\mathbf{H}(\theta^*)) = r$ which implies that $\text{rank}(\phi_z(\mathbf{z})) + \text{rank}(\mathbf{J}(\theta^*)) = 2m - r + r = 2m$. Hence, in this case, θ^* is a semiregular zero of $\nabla_\theta L(\theta) = 0$.

5 Convergence Analysis

5.1 Gauss–Newton Method

In this section, we will analyze the convergence properties of the Gauss–Newton method shown in (3.16). We assume that the variational problem has a semiregular zero θ^* such that $\nabla_\theta L(\theta^*) = 0$, and we also assume that rank $\boldsymbol{J}(\theta^*) = r \le m$. Using singular value decomposition, we can write $J(\theta^*) = U\Sigma V^T$, where $J(\theta^*)$ is an r-rank semi-positive definite matrix. In particular, we have [33]

$$\Sigma = diag\left([\sigma_1, \cdots, \sigma_r, 0, \cdots, 0]\right), \text{ with } \sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_r > 0, r \le m. \tag{5.1}$$

Thus, the pseudo-inverse can be represented as

$$\boldsymbol{J}(\theta^*)^\dagger = V\Sigma^\dagger U^T, \text{ with } \Sigma^\dagger = diag\left(\left[\frac{1}{\sigma_1}, \cdots, \frac{1}{\sigma_r}, 0, \cdots, 0\right]\right). \tag{5.2}$$

Here, the pseudo-inverse is based on the rank-r projection of $\boldsymbol{J}(\theta)$ denoted as $\boldsymbol{J}_{\text{rank-r}}(\theta)$ in [36]. However, for simplicity, in our paper, we use the term $\boldsymbol{J}(\theta)$ to represent the rank-r projection.

Lemma 3 *Let $\boldsymbol{J}(\theta)$ be the approximated Hessian defined in (3.4) and θ^* be the stationary point. Then there exists a small open set $\Omega_* \ni \theta^*$ and constants $\epsilon, \zeta, \alpha > 0$ such that for* *every $\theta_1, \theta_2 \in \Omega_*$, the following inequalities holds:*

$$\|\boldsymbol{J}(\theta_2)\boldsymbol{J}(\theta_2)^{\dagger} - \boldsymbol{J}(\theta_1)\boldsymbol{J}(\theta_1)^{\dagger}\| \leq \zeta\|\theta_2 - \theta_1\|, \tag{5.3}$$

$$\begin{aligned}&\|\nabla_\theta L(\theta_2) - \nabla_\theta L(\theta_1) - \boldsymbol{J}(\theta_1)\,(\theta_2 - \theta_1)\,\| \\ &\quad \leq \epsilon\|\theta_2 - \theta_1\| + \alpha\|\theta_2 - \theta_1\|^2.\end{aligned} \tag{5.4}$$

Proof From Taylor's expansion, we have

$$\begin{aligned}&\nabla_\theta L(\theta_2) - \nabla_\theta L(\theta_1) - \boldsymbol{J}(\theta_1)\,(\theta_2 - \theta_1) \\ &\quad = \mathbf{H}(\theta_1)\,(\theta_2 - \theta_1) - \boldsymbol{J}(\theta_1)\,(\theta_2 - \theta_1) + \mathcal{O}(\|\theta_2 - \theta_1\|^2) \\ &\quad = \boldsymbol{Q}(\theta_1)\,(\theta_2 - \theta_1) + \mathcal{O}(\|\theta_2 - \theta_1\|^2).\end{aligned}$$

Therefore, due to the smoothness of the Hessian with respect to θ and Lemma 1, there exists a small neighborhood Ω_* of θ^* and a constant $\alpha > 0$ such that for every $\theta_1, \theta_2 \in \Omega_*$, we have $\|\boldsymbol{Q}(\theta_1)\| \leq \epsilon$, and the following inequality holds:

$$\|\nabla_\theta L(\theta_2) - \nabla_\theta L(\theta_1) - \boldsymbol{J}(\theta_1)\,(\theta_2 - \theta_1)\,\| \leq \epsilon\|\theta_2 - \theta_1\| + \alpha\|\theta_2 - \theta_1\|^2. \tag{5.5}$$

for every $\theta_1, \theta_2 \in \Omega_*$. On the other hand, Weyl's Theorem guarantees the singular value to be continuous with respect to the matrix entries. Therefore, as the open set Ω_* being sufficiently small, it holds

$$\sup_{\theta\in\Omega_*}\|\boldsymbol{J}(\theta)\| = \sup_{\theta\in\Omega_*}\sigma_1(\boldsymbol{J}(\theta)) \leq C\|\boldsymbol{J}(\theta^*)\|$$

$$\sup_{\theta\in\Omega_*}\|\boldsymbol{J}(\theta)^{\dagger}\| = \frac{1}{\sup_{\theta\in\Omega_*}\sigma_r(\boldsymbol{J}(\theta))} \leq C\|\boldsymbol{J}(\theta^*)^{\dagger}\|$$

for all $\theta \in \Omega_*$. By the smoothness of $\boldsymbol{J}(\theta)$ with respect to θ and the estimation in Lemma 8, we have

$$\begin{aligned}&\|\boldsymbol{J}(\theta_2)\boldsymbol{J}(\theta_2)^{\dagger} - \boldsymbol{J}(\theta_1)\boldsymbol{J}(\theta_1)^{\dagger}\| \\ &\quad \leq \|\boldsymbol{J}(\theta_2)^{\dagger}\|\|\boldsymbol{J}(\theta_2) - \boldsymbol{J}(\theta_1)\| \\ &\qquad + \|\boldsymbol{J}(\theta_1)\|\|\boldsymbol{J}(\theta_2)^{\dagger} - \boldsymbol{J}(\theta_1)^{\dagger}\| \\ &\quad \leq \zeta\|\theta_2 - \theta_1\|.\end{aligned}$$

□

Theorem 1 *(Convergence Theorem) Let $L(\theta)$ be a sufficiently smooth target function of θ and $\boldsymbol{J}(\theta)$ be the approximated Hessian of $L(\theta)$ defined in (3.4). Then for every open neighborhood Ω_1 of θ^*, there exists another neighborhood $\Omega_2 \ni \theta^*$ such that, from every initial guess $\theta_0 \in \Omega_2$, the sequence $\{\theta_k\}_{k=1}^{\infty}$ generated by the iteration (3.16) converges in Ω_1. Furthermore, $\{\theta_k\}_{k=1}^{\infty}$ has at least linear convergence with a coefficient $\gamma \leq 2\epsilon$ when k is large enough, where ϵ is the given constant in Lemma 1.*

Proof Firstly, let Ω_* be the small open neighbourhood in Lemma 3 such that $\|\boldsymbol{Q}(\theta)\| \leq \epsilon < \frac{1}{2}$. For any open neighbourhood Ω_1 of θ^*, there exists a constant $0 < \delta < 2$ such that $B(\theta^*, \delta) \subset \Omega_1 \cap \Omega_*$ and for any $\theta_1, \theta_2 \in B(\theta^*, \delta)$, it holds

$$\|\boldsymbol{J}(\theta_2)^{\dagger}\|\,(\alpha\|\theta_2 - \theta_1\| + \zeta\|\nabla_\theta L(\theta_1)\|) \leq h < 1, \tag{5.6}$$

where α and ζ are constants introduced in Lemma 3.

On the other hand, there also exists $0 < \tau < \frac{\delta}{2}$ such that

$$\|\boldsymbol{J}(\theta)^{\dagger}\|\|\nabla_{\theta}L(\theta)\| \le \frac{1-h}{2}\delta < \frac{\delta}{2} < 1 \tag{5.7}$$

holds for any $\theta \in B(\theta^*, \tau)$. Let $\Omega_2 = B(\theta^*, \tau)$, then by (5.7), we have for any $\theta_0 \in \Omega_2$, the iteration implies

$$\begin{aligned}\|\theta_1 - \theta^*\| &\le \|\theta_1 - \theta_0\| + \|\theta_0 - \theta^*\| \\ &\le \|\boldsymbol{J}(\theta_0)^{\dagger}\|\|\nabla_{\theta}L(\theta_0)\| + \tau < \delta.\end{aligned} \tag{5.8}$$

Next we assume that $\theta_k, \theta_{k-1} \in B(\theta^*, \delta)$ for some integer $k \ge 1$. From Lemma 3 we have the following estimation

$$\begin{aligned}&\|\theta_{k+1} - \theta_k\| = \|\boldsymbol{J}(\theta_k)^{\dagger}\nabla_{\theta}L(\theta_k)\| \\ &= \|\boldsymbol{J}(\theta_k)^{\dagger}\left(\nabla_{\theta}L(\theta_k) - \boldsymbol{J}(\theta_{k-1})\left(\theta_k - \theta_{k-1} + \boldsymbol{J}(\theta_{k-1})^{\dagger}\nabla_{\theta}L(\theta_{k-1})\right)\right)\| \\ &\quad \le \|\boldsymbol{J}(\theta_k)^{\dagger} \\ &\quad (\nabla_{\theta}L(\theta_k) - \nabla_{\theta}L(\theta_{k-1}) - \boldsymbol{J}(\theta_{k-1})(\theta_k - \theta_{k-1}))\| \\ &\quad + \|\boldsymbol{J}(\theta_k)^{\dagger}\left(\boldsymbol{J}(\theta_k)\boldsymbol{J}(\theta_k)^{\dagger} - \boldsymbol{J}(\theta_{k-1})\boldsymbol{J}(\theta_{k-1})^{\dagger}\right) \\ &\quad \nabla_{\theta}L(\theta_{k-1})\| \\ &\le \|\boldsymbol{J}(\theta_k)^{\dagger}\|\left(\alpha\|\theta_k - \theta_{k-1}\| + \zeta\|\nabla_{\theta}L(\theta_{k-1})\| + \epsilon\right)\|\theta_k - \theta_{k-1}\|.\end{aligned} \tag{5.9}$$

Since $\theta_k, \theta_{k-1} \in B(\theta^*, \delta)$, then by (5.6), it leads to

$$\|\theta_{k+1} - \theta_k\| < h\|\theta_k - \theta_{k-1}\| < \|\theta_k - \theta_{k-1}\| \tag{5.10}$$

so the convergence is guaranteed. Therefore, we obtain that $\|\theta_j - \theta_{j-1}\| \le h^j\|\theta_1 - \theta_0\|$ for $1 \le j \le k+1$, and

$$\begin{aligned}\|\theta_{k+1} - \theta^*\| &\le \|\theta_0 - \theta^*\| + \sum_{j=0}^{k}\|\theta_{k-j+1} - \theta_{k-j}\| \\ &\le \|\theta_0 - \theta^*\| + \sum_{j=0}^{k} h^j\|\theta_0 - \theta^*\| \\ &< \frac{1}{1-h}\|\theta_1 - \theta_0\| + \|\theta_0 - \theta^*\| \\ &< \frac{1}{1-h}\frac{1-h}{2}\delta + \frac{1}{2}\delta = \delta,\end{aligned}$$

which completes the induction. Thus we conclude that the sequence $\{\theta_k\}_{k=0}^{\infty} \subset B(\theta^*, \delta) \subset \Omega_1$ as long as the initial iterate $\theta_0 \in B(\theta^*, \tau) = \Omega_2$.
Secondly, let us define $\hat{\theta} = \lim_{k\to\infty}\theta_k$ so that $\hat{\theta} \in \Omega_1$. By the smoothness of $\nabla_{\theta}L(\theta)$ and the convergence property (5.10), there exists a constant $\mu > 0$ such that

$$\begin{aligned}\|\nabla_{\theta}L(\theta_{k-1})\| &= \|\nabla_{\theta}L(\theta_{k-1}) - \nabla_{\theta}L(\hat{\theta})\| \\ &\le \mu\|\theta_{k-1} - \hat{\theta}\| \\ &\le \mu\left(\|\theta_{k-1} - \theta_k\| + \|\theta_k - \theta_{k+1}\| + \cdots\right) \\ &\le \frac{\mu}{1-h}\|\theta_k - \theta_{k-1}\|.\end{aligned} \tag{5.11}$$

Now let us combine (5.9) and (5.11) to find that

$$\|\theta_{k+1}-\theta_k\| \le \beta\|\theta_k-\theta_{k-1}\|^2+\epsilon\|\theta_k-\theta_{k-1}\| \tag{5.12}$$

$$= (\beta\|\theta_k-\theta_{k-1}\|+\epsilon)\,\|\theta_k-\theta_{k-1}\| \tag{5.13}$$

for some constant $\beta > 0$.

In the case when k is not large enough, i.e., for $k \le k_0$ with some k_0, such that $\beta\|\theta_k - \theta_{k-1}\| \ge \epsilon$, we have

$$\begin{aligned}\|\theta_{k+1}-\theta_k\| &= (\beta\|\theta_k-\theta_{k-1}\|+\epsilon)\,\|\theta_k-\theta_{k-1}\|\\ &\le (\beta\|\theta_k-\theta_{k-1}\|+\beta\|\theta_k-\theta_{k-1}\|)\,\|\theta_k-\theta_{k-1}\|\\ &= 2\beta\|\theta_k-\theta_{k-1}\|^2\\ &\le (2\beta)^{1+2+4+\cdots+2^{k-1}}\|\theta_1-\theta_0\|^{2^k}\\ &= (2\beta)^{\frac{2^k-1}{2}}\|\theta_1-\theta_0\|^{2^k}\end{aligned}$$

On the other hand, it can be easily seen that

$$\begin{aligned}\|\theta_{k+1}-\hat{\theta}\| &\le \|\theta_{k+1}-\theta_{k+2}\|+\|\theta_{k+2}-\theta_{k+3}\|+\|\theta_{k+3}-\theta_{k+4}\|+\cdots\\ &\le \left(h+h^2+h^3+\cdots\right)\|\theta_{k+1}-\theta_k\|\\ &\le \frac{h}{1-h}\|\theta_{k+1}-\theta_k\|.\end{aligned}$$

Therefore, when $k \le k_0$ with some k_0, we have quadratic convergence as follows

$$\|\theta_{k+1}-\hat{\theta}\| \le \frac{h}{1-h}\|\theta_{k+1}-\theta_k\| \tag{5.14}$$

$$= \frac{h}{1-h}(2\beta)^{\frac{2^k-1}{2}}\|\theta_1-\theta_0\|^{2^k}. \tag{5.15}$$

As k grows sufficiently large with $k \ge \bar{k}$ such that $\beta\|\theta_k - \theta_{k-1}\| \le \epsilon$, we can use the estimate (5.12) to obtain

$$(\beta\|\theta_k-\theta_{k-1}\|+\epsilon) \le 2\epsilon < 1.$$

Therefore

$$\|\theta_{k+1}-\theta_k\| \le 2\epsilon\|\theta_k-\theta_{k-1}\| \tag{5.16}$$

$$\le (2\epsilon)^k\|\theta^1-\theta^0\| \tag{5.17}$$

Hence

$$\|\theta_{k+1}-\hat{\theta}\| \le \frac{h}{1-h} \le (2\epsilon)^k\|\theta^1-\theta^0\|. \tag{5.18}$$

The estimates (5.14) and (5.18) show that $\{\|\theta_{k+1}-\hat{\theta}\|\}_{k=1}^{\infty}$ is a decreasing sequence that exhibits quadratic convergence when the number of iterations is small, and at least has a linear convergence sequence with coefficient 2ϵ when the number of iterations is large. Since 2ϵ can be very small due to the construction of this iteration, the parameter sequence $\{\theta_k\}_{k=1}^{\infty}$ will converge rapidly in numerical experiments. □

5.2 Random Gauss–Newton Method

By employing Monte Carlo approximation with random sampling points to approximate the integration of $L(\theta)$, the Gauss–Newton method can be transformed into a random algorithm. Specifically, we define $\xi : \Omega \to \Gamma$ as a random variable, where Γ is a set containing all the combinations of $\tilde{N} + \tilde{n}$ numbers that denote the sampling indices on both the domain and boundaries selected from $1, 2, \cdots, N, N+1, \cdots, N+n,$. The cardinality of Γ is given by $|\Gamma| = \binom{N+n}{m+r}$. Here, we assume that the total number of sampling points on the domain and boundary are N and n, respectively. In each iteration, we draw $\tilde{N}$ and $\tilde{n}$ random samples from the domain and boundary, respectively.

Therefore, the random Gauss–Newton method can be expressed as follows:

$$\theta_{k+1}(\xi_k) = \theta_k(\xi_k) - \boldsymbol{J}(\theta_k; \xi_k)^{\dagger} \nabla_\theta L(\theta_k; \xi_k). \tag{5.19}$$

Here, $\theta_{k+1}(\xi_k)$ represents the updated parameter vector at the $(k+1)$-th iteration, which depends on the random variable ξ_k.

Additionally, it's important to note that at each step k, the sample points on both the domain and boundary are given by:

$$x_{s_1}, x_{s_2}, \cdots, x_{s_{\tilde{N}}}. x^b_{s^b_1}, x^b_{s^b_2}, \cdots, x^b_{s^b_{\tilde{n}}}.$$

Lemma 4 *Let $\boldsymbol{J}(\theta)$ be the approximated Hessian defined in (3.4) and θ^* be the stationary point. Then there exists a small open set $\Omega_* \ni \theta^*$ and constants $\epsilon, \zeta, \alpha > 0$ such that for every $\theta_1, \theta_2 \in \Omega_*$, the following inequalities holds:*

$$\mathbb{E}_{\xi_2,\xi_1}\left(\|\boldsymbol{J}(\theta_2,\xi_2)\boldsymbol{J}(\theta_2,\xi_2)^{\dagger} - \boldsymbol{J}(\theta_1,\xi_1)\boldsymbol{J}(\theta_1,\xi_1)^{\dagger}\|\right) \le \zeta \mathbb{E}_{\xi}\left(\|\theta_2-\theta_1\|\right), \tag{5.20}$$

$$\begin{aligned} &\mathbb{E}_{\xi_2,\xi_1}\left(\|\nabla_\theta L(\theta_2,\xi_2) - \nabla_\theta L(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_1)(\theta_2-\theta_1)\|\right) \\ &\quad \le \epsilon \mathbb{E}_{\xi}\left(\|\theta_2-\theta_1\|\right) + \alpha \mathbb{E}_{\xi}\left(\|\theta_2-\theta_1\|\right)^2. \end{aligned} \tag{5.21}$$

Proof From Taylor's expansion, we have

$$\begin{aligned} &\nabla_\theta L(\theta_2,\xi_2) - \nabla_\theta L(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_1)(\theta_2-\theta_1) \\ &\quad = \nabla_\theta L(\theta_2,\xi_2) - \nabla_\theta L(\theta_1,\xi_2) + \nabla_\theta L(\theta_1,\xi_2) - \nabla_\theta L(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_1)(\theta_2-\theta_1) \\ &\quad = \mathbf{H}(\theta_1,\xi_2)(\theta_2-\theta_1) + \mathcal{O}(\|\theta_2-\theta_1\|^2) + \nabla_\theta L(\theta_1,\xi_2) - \nabla_\theta L(\theta_1,\xi_1) \\ &\qquad - (\boldsymbol{J}(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_2) + \boldsymbol{J}(\theta_1,\xi_2))(\theta_2-\theta_1) \\ &\quad = \boldsymbol{H}(\theta_1,\xi_2)(\theta_2-\theta_1) - \boldsymbol{J}(\theta_1,\xi_2)(\theta_2-\theta_1) + \mathcal{O}(\|\theta_2-\theta_1\|^2) \\ &\qquad + \nabla_\theta L(\theta_1,\xi_2) - \nabla_\theta L(\theta_1,\xi_1) - (\boldsymbol{J}(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_2))(\theta_2-\theta_1) \\ &\quad = \boldsymbol{Q}(\theta_1,\xi_2)(\theta_2-\theta_1) + \mathcal{O}(\|\theta_2-\theta_1\|^2) + \nabla_\theta L(\theta_1,\xi_2) - \nabla_\theta L(\theta_1,\xi_1) \\ &\qquad - (\boldsymbol{J}(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_2))(\theta_2-\theta_1). \end{aligned}$$

Now taking norms and expectation from both sides and noting that

$$\mathbb{E}_{\xi_2,\xi_1}\left(\|\nabla_\theta L(\theta_1,\xi_2) - \nabla_\theta L(\theta_1,\xi_1)\|\right) = 0$$

and

$$\mathbb{E}_{\xi_2,\xi_1}\left(\|(\boldsymbol{J}(\theta_1,\xi_1) - \boldsymbol{J}(\theta_1,\xi_2))(\theta_2-\theta_1)\|\right) = 0$$

we have

$$\begin{aligned}&\mathbb{E}_{\xi_2,\xi_1}\left(\|\nabla_\theta L(\theta_2,\xi_2)-\nabla_\theta L(\theta_1,\xi_1)-\boldsymbol{J}(\theta_1,\xi_1)(\theta_2-\theta_1)\|\right)\\ &\le \mathbb{E}_{\xi_2,\xi_1}\left(\|\boldsymbol{Q}(\theta_1,\xi_2)(\theta_2-\theta_1)\|\right)+\mathbb{E}_\xi\left(\mathcal{O}(\|\theta_2-\theta_1\|^2)\right).\end{aligned}$$

Therefore, due to the smoothness of the Hessian with respect to θ and Lemma 1, there exists a small neighborhood Ω_* of θ^* and a constant $\alpha>0$ such that for every $\theta_1,\theta_2\in\Omega_*$, we have $\mathbb{E}_{\xi_2}(\|\boldsymbol{Q}(\theta_1,\xi_2)\|)\le\epsilon$, and the following inequality holds:

$$\mathbb{E}_{\xi_2,\xi_1}\left(\|\nabla_\theta L(\theta_2,\xi_2)-\nabla_\theta L(\theta_1,\xi_1)-\boldsymbol{J}(\theta_1,\xi_1)(\theta_2-\theta_1)\|\right) \tag{5.22}$$

$$\le \epsilon\mathbb{E}_\xi(\|\theta_2-\theta_1\|)+\alpha\mathbb{E}_\xi(\|\theta_2-\theta_1\|^2) \tag{5.23}$$

for every $\theta_1,\theta_2\in\Omega_*$. On the other hand, Weyl's Theorem guarantees the singular value to be continuous with respect to the matrix entries. Therefore, as the open set Ω_* being sufficiently small, it holds

$$\begin{aligned}&\sup_{\theta\in\Omega_*}\mathbb{E}_\xi(\|\boldsymbol{J}(\theta,\xi)\|)=\sup_{\theta\in\Omega_*}\sigma_1\mathbb{E}_\xi(\boldsymbol{J}(\theta,\xi))\le C\mathbb{E}_\xi(\|\boldsymbol{J}(\theta^*,\xi)\|)\\ &\sup_{\theta\in\Omega_*}\mathbb{E}_\xi(\|\boldsymbol{J}(\theta,\xi)^\dagger\|)=\frac{1}{\sup_{\theta\in\Omega_*}\sigma_r\mathbb{E}_\xi(\boldsymbol{J}(\theta,\xi))}\le C\mathbb{E}_\xi(\|\boldsymbol{J}(\theta^*,\xi)^\dagger\|)\end{aligned}$$

for all $\theta\in\Omega_*$. By the smoothness of $\mathbb{E}_\xi(\boldsymbol{J}(\theta,\xi))$ with respect to θ and the estimation in Lemma 8, we have

$$\begin{aligned}&\mathbb{E}_{\xi_2,\xi_1}(\|\boldsymbol{J}(\theta_2,\xi_2)\boldsymbol{J}(\theta_2,\xi_2)^\dagger-\boldsymbol{J}(\theta_1,\xi_1)\boldsymbol{J}(\theta_1,\xi_1)^\dagger\|)\\ &\quad\le\mathbb{E}_\xi(\|\boldsymbol{J}(\theta_2,\xi_2)^\dagger\|)\mathbb{E}_{\xi_2,\xi_1}(\|\boldsymbol{J}(\theta_2,\xi_2)-\boldsymbol{J}(\theta_1,\xi_1)\|)\\ &\qquad+\mathbb{E}_{\xi_1}(\|\boldsymbol{J}(\theta_1,\xi_1)\|)\mathbb{E}_{\xi_2,\xi_1}(\|\boldsymbol{J}(\theta_2,\xi_2)^\dagger-\boldsymbol{J}(\theta_1,\xi_1)^\dagger\|)\\ &\quad\le\zeta\mathbb{E}_\xi(\|\theta_2-\theta_1\|).\end{aligned}$$

□

Theorem 2 *(Convergence Theorem) Let $L(\theta)$ be a sufficiently smooth target function of θ and $\boldsymbol{J}(\theta)$ be the approximated Hessian of $L(\theta)$ defined in (3.4). Then for every open neighbourhood Ω_1 of θ^*, there exists another neighbourhood $\Omega_2\ni\theta^*$ such that, from every initial guess $\theta_0\in\Omega_2$, the expectation of the sequence $\{\theta_k\}_{k=1}^\infty$ generated by the iteration (5.19), we have*

$$\mathbb{E}(\|\theta_{k+1}-\hat{\theta}\|)\le C\epsilon^k, \tag{5.24}$$

where $\hat{\theta}$ is a semi-regular zero and ϵ is the given constant in Lemma 1.

Proof Firstly, Let Ω_* be the small open neighbourhood in Lemma 3 such that $\mathbb{E}_\xi(\|\boldsymbol{Q}(\theta,\xi)\|)\le\epsilon<\frac{1}{2}$. For any open neighbourhood Ω_1 of θ^*, there exists a constant $0<\delta<2$ such that $B(\theta^*,\delta)\subset\Omega_1\cap\Omega_*$ and for any $\theta_1,\theta_2\in B(\theta^*,\delta)$, it holds

$$\mathbb{E}_{\xi_2}(\|\boldsymbol{J}(\theta_2,\xi_2)^\dagger\|)\left(\alpha\mathbb{E}_\xi(\|\theta_2-\theta_1\|)+\zeta\mathbb{E}_{\xi_1}(\|\nabla_\theta L(\theta_1,\xi_1)\|)\right)\le h<1. \tag{5.25}$$

On the other hand, there also exists $0<\tau<\frac{\delta}{2}$ such that

$$\mathbb{E}_\xi(\|\boldsymbol{J}(\theta,\xi)^\dagger\|)\mathbb{E}_\xi(\|\nabla_\theta L(\theta,\xi)\|)\le\frac{1-h}{2}\delta<\frac{\delta}{2}<1 \tag{5.26}$$

holds for any $\theta \in B(\theta^*, \tau)$. Let $\Omega_2 = B(\theta^*, \tau)$, then for $\forall\theta_0 \in \Omega_2$, the iteration implies

$$\begin{aligned}\mathbb{E}_\xi(\|\theta_1 - \theta^*\|) &\le \mathbb{E}_\xi(\|\theta_1 - \theta_0\|) + \mathbb{E}_\xi(\|\theta_0 - \theta^*\|)\\ &\le \mathbb{E}_{\xi_0}(\|\boldsymbol{J}(\theta_0, \xi_0)^\dagger\|)\mathbb{E}_{\xi_0}(\|\nabla_\theta L(\theta_0, \xi_0)\|) + \tau < \delta. \end{aligned} \tag{5.27}$$

Next we assume that $\theta_k \in B(\theta^*, \delta)$ for some integer $k \ge 1$. From Lemma 3 we have the following estimation

$$\begin{aligned}&\mathbb{E}_\xi(\|\theta_{k+1} - \theta_k\|) = \mathbb{E}_\xi(\|\boldsymbol{J}(\theta_k, \xi_k)^\dagger\nabla_\theta L(\theta_k, \xi_k)\|)\\ &\quad = \mathbb{E}_{\xi_0,\xi_1,\cdots,\xi_{k-1}}\mathbb{E}_{\xi_k}\\ &\qquad \times\left(\|\boldsymbol{J}(\theta_k, \xi_k)^\dagger\left(\nabla_\theta L(\theta_k, \xi_k) - \boldsymbol{J}(\theta_{k-1}, \xi_{k-1})\left(\theta_k - \theta_{k-1} + \boldsymbol{J}(\theta_{k-1}, \xi_{k-1})^\dagger\nabla_\theta L(\theta_{k-1}, \xi_{k-1})\right)\right)\|\right)\\ &\quad \le \mathbb{E}_{\xi_0,\xi_1,\cdots,\xi_{k-1}}\mathbb{E}_{\xi_k}\\ &\qquad \times\left(\|\boldsymbol{J}(\theta_k, \xi_k)^\dagger\left(\nabla_\theta L(\theta_k, \xi_k) - \nabla_\theta L(\theta_{k-1}, \xi_{k-1}) - \boldsymbol{J}(\theta_{k-1}, \xi_{k-1})(\theta_k - \theta_{k-1})\right)\|\right)\\ &\qquad + \mathbb{E}_{\xi_0,\xi_1,\cdots,\xi_{k-1}}\mathbb{E}_{\xi_k}\left(\|\boldsymbol{J}(\theta_k, \xi_k)^\dagger\left(\boldsymbol{J}(\theta_k, \xi_k)\boldsymbol{J}(\theta_k, \xi_k)^\dagger - \boldsymbol{J}(\theta_{k-1}, \xi_{k-1})\boldsymbol{J}(\theta_{k-1}, \xi_{k-1})^\dagger\right)\nabla_\theta L(\theta_{k-1}, \xi_{k-1})\|\right)\\ &\quad \le \mathbb{E}_{\xi_k}(\|\boldsymbol{J}(\theta_k, \xi_k)^\dagger\|)\left(\alpha\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|) + \zeta\mathbb{E}_{\xi_{k-1}}(\|\nabla_\theta L(\theta_{k-1}, \xi_{k-1})\|) + \epsilon\right)\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|).\end{aligned} \tag{5.28}$$

Since $\theta_k \in B(\theta^*, \delta)$, then it leads to

$$\mathbb{E}_\xi(\|\theta_{k+1} - \theta_k\|) < h\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|) < \mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|) \tag{5.29}$$

so the convergence is guaranteed. Therefore, we obtain that $\mathbb{E}_\xi(\|\theta_j - \theta_{j-1}\|) \le h^j\mathbb{E}_\xi(\|\theta_1 - \theta_0\|)$ for $1 \le j \le k + 1$, and

$$\begin{aligned}\mathbb{E}_\xi(\|\theta_{k+1} - \theta^*\|) &\le \mathbb{E}_\xi(\|\theta_0 - \theta^*\|) + \sum_{j=0}^{k}\mathbb{E}_\xi(\|\theta_{k-j+1} - \theta_{k-j}\|)\\ &\le \mathbb{E}_\xi(\|\theta_0 - \theta^*\|) + \sum_{j=0}^{k}h^j\mathbb{E}_\xi(\|\theta_0 - \theta^*\|)\\ &< \frac{1}{1-h}\mathbb{E}_\xi(\|\theta_1 - \theta_0\|) + \mathbb{E}_\xi(\|\theta_0 - \theta^*\|)\\ &< \frac{1}{1-h}\frac{h-1}{2}\delta + \frac{1}{2}\delta = \delta,\end{aligned}$$

which completes the induction. Thus we conclude that the sequence $\{\mathbb{E}_\xi(\theta_k)\}_{k=0}^\infty \subset B(\theta^*, \delta) \subset \Omega_1$ as long as the initial iterate $\mathbb{E}_\xi(\theta_0) \in B(\theta^*, \tau) = \Omega_2$.

Secondly, let us define $\mathbb{E}_\xi(\hat{\theta}) = \lim_{k\to\infty}\mathbb{E}_\xi(\theta_k)$ so that $\mathbb{E}_\xi(\hat{\theta}) \in \Omega_1$. By the smoothness of $\mathbb{E}_\xi(\nabla_\theta L(\theta, \xi))$ and the convergence property (5.29), there exists a constant $\mu > 0$ such that

$$\begin{aligned}\mathbb{E}_{\xi_{k-1}}(\|\nabla_\theta L(\theta_{k-1}, \xi_{k-1})\|) &= \mathbb{E}_{\xi_{k-1}}(\|\nabla_\theta L(\theta_{k-1}, \xi_{k-1}) - \nabla_\theta L(\hat{\theta}, \xi_{k-1})\|)\\ &\le \mu\mathbb{E}_\xi(\|\theta_{k-1} - \hat{\theta}\|)\\ &\le \mu\mathbb{E}_\xi\left((\|\theta_{k-1} - \theta_k\|) + \mathbb{E}_\xi(\|\theta_k - \theta_{k+1}\|) + \cdots\right)\\ &\le \frac{\mu}{1-h}\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|).\end{aligned} \tag{5.30}$$

Now let us combine (5.28) and (5.30) to find that

$$\mathbb{E}_\xi(\|\theta_{k+1} - \theta_k\|) \le \beta\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|)^2 + \epsilon\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|) \tag{5.31}$$

$$= \left(\beta\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|) + \epsilon\right)\mathbb{E}_\xi(\|\theta_k - \theta_{k-1}\|) \tag{5.32}$$

for some constant $\beta > 0$.

In the case when k is not large enough, i.e., for $k \le k_0$ with some k_0, such that $\beta\|\theta_k - \theta_{k-1}\| \ge \epsilon$, we have

$$\begin{aligned}\mathbb{E}_\xi(\|\theta_{k+1}-\theta_k\|) &= \big(\beta\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)+\epsilon\big)\,\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)\\ &= \big(\beta\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)+\beta\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)\big)\,\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)\\ &= 2\beta\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)^2\\ &\le (2\beta)^{1+2+4+\cdots+2^{k-1}}\mathbb{E}_\xi(\|\theta_1-\theta_0\|)^{2^k}\\ &= (2\beta)^{\frac{2^k-1}{2}}\mathbb{E}_\xi(\|\theta_1-\theta_0\|)^{2^k}\end{aligned}$$

On the other hand, it can be easily seen that

$$\begin{aligned}\mathbb{E}_\xi(\|\theta_{k+1}-\hat{\theta}\|) &= \mathbb{E}_\xi(\|\theta_{k+1}-\theta^{k+2}\|)+\mathbb{E}_\xi(\|\theta^{k+2}-\theta^{k+3}\|)+\mathbb{E}_\xi(\|\theta^{k+3}-\theta^{k+4}\|)+\cdots\\ &\le \big(h+h^2+h^3+\cdots\big)\,\mathbb{E}_\xi(\|\theta_{k+1}-\theta_k\|)\\ &\le \frac{h}{1-h}\mathbb{E}_\xi(\|\theta_{k+1}-\theta_k\|).\end{aligned}$$

Therefore, when $k \le k_0$ with some k_0, we have quadratic convergence as follows

$$\mathbb{E}_\xi(\|\theta_{k+1}-\hat{\theta}\|) \le \frac{h}{1-h}\mathbb{E}_\xi(\|\theta_{k+1}-\theta_k\|) \tag{5.33}$$

$$= \frac{h}{1-h}(2\beta)^{\frac{2^k-1}{2}}\mathbb{E}_\xi(\|\theta_1-\theta_0\|)^{2^k}. \tag{5.34}$$

As k grows sufficiently large with $k \ge \bar{k}$ such that $\beta\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|) \le \epsilon$, we can use the estimate (5.31) to obtain

$$\big(\beta\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|)+\epsilon\big) \le 2\epsilon < 1.$$

Therefore

$$\mathbb{E}_\xi(\|\theta_{k+1}-\theta_k\|) \le 2\epsilon\mathbb{E}_\xi(\|\theta_k-\theta_{k-1}\|) \tag{5.35}$$

$$\le (2\epsilon)^k\mathbb{E}_\xi(\|\theta^1-\theta^0\|) \tag{5.36}$$

Hence

$$\mathbb{E}_\xi(\|\theta_{k+1}-\hat{\theta}\|) \le \frac{h}{1-h} \le (2\epsilon)^k\mathbb{E}_\xi(\|\theta^1-\theta^0\|). \tag{5.37}$$

The estimates (5.33) and (5.37) show that $\{\mathbb{E}_\xi(\|\theta_{k+1}-\hat{\theta}\|)\}_{k=1}^\infty$ is a decreasing sequence that exhibits quadratic convergence when the number of iterations is small, and at least has a linear convergence sequence with coefficient 2ϵ when the number of iterations is large. Since 2ϵ can be very small due to the construction of this iteration, the parameter sequence $\{\mathbb{E}_\xi(\theta_k)\}_{k=1}^\infty$ will converge rapidly in numerical experiments. □

6 Numerical Experiments

In this section, we numerically investigate the efficiency and robustness of the Gauss–Newton method by solving the model problem (2.7) in different dimensions. All experiments in the following were run on a single NVIDIA TESLA P100 GPU with double precision. All the code in this section can be found on GitHub: https://github.com/Jinxl-pp/GaussNewtonDRM.

Table 1 Errors in both L^2 and H^1 norms for various training algorithms with different neural network architectures

Hidden Layer Width		16	32	64	128	256
SGD	L^2-error	7.04e−3	4.49e−3	3.53e−3	3.95e−3	3.98e−3
	H^1-error	6.98e−2	5.45e−2	4.26e−2	4.57e−2	4.73e−2
ADAM	L^2-error	1.89e−3	1.22e−3	2.24e−4	2.03e−4	1.37e−4
	H^1-error	2.76e−2	1.97e−2	6.07e−3	4.76e−3	4.01e−3
L-BFGS	L^2-error	2.81e−4	1.24e−4	4.19e−5	2.18e−5	2.36e−5
	H^1-error	6.96e−3	3.55e−3	1.53e−3	8.19e−4	9.06e−4
Gauss–Newton	L^2-error	**7.86e−5**	**3.45e−5**	**5.83e−6**	**2.71e−6**	**3.78e−7**
	H^1-error	**2.43e−3**	**1.30e−3**	**3.71e−4**	**2.05e−4**	**4.50e−5**

Example 1 A second-order elliptic equation in one-dimensional space is given as follows:

$$\begin{aligned} -u''(x) + u(x) &= f(x),\ x \in (-1, 1), \\ u'(-1) = u'(1) &= 0. \end{aligned}$$

The corresponding variational problem becomes

$$\min_{v \in H^1} \mathcal{J}(v) = \int_{-1}^{1} \left(\frac{1}{2}(v')^2 + \frac{1}{2}v^2 - fv \right) \mathrm{d}x.$$

By choosing $f(x) = (\pi^2+1)\cos(\pi x)$, we have $u(x) = \cos(\pi x)$. We employ shallow neural networks represented as u_n with a ReLU^3 activation function. Then, for the discretized energy denoted as $\mathcal{J}(u_n)$, we utilize a piecewise Gauss-Legendre quadrature rule with 12,000 sample points. This rule is based on the 2-point Gauss-Legendre rule and is distributed across 6,000 uniform sub-intervals within the range of $[-1, 1]$. These sample points serve as the training data. As for the testing dataset, we extract 16,000 sample points from the same quadrature rule across 8,000 uniform sub-intervals. These testing points are used for calculating the numerical errors.

Now we proceed to compare the Gauss–Newton method with other training techniques, such as SGD, ADAM, and L-BFGS, across various neural network architectures. The adaptive learning rate is implemented through a back-tracking strategy. This adjustment is crucial as the Gauss–Newton method exhibits local convergence, necessitating global optimization techniques for the warming-up stage and achieving global convergence.

During training with Newton-type algorithms (L-BFGS, Gauss–Newton), we conduct 1000 epochs, while for gradient-based algorithms (SGD and ADAM), we extend the training to 20,000 epochs. Figure 1 illustrates the detailed dynamics of the training loss in both L^2 and H^1. The shaded areas represent the range between the first and third quartiles of the 10 experiments, while the lines depict the median values. Since $\mathcal{J}(v) - \mathcal{J}(u) = \frac{1}{2}\|v-u\|_{H^1}^2$, greater accuracy in the H^1 norm signifies better convergence of the loss function.

Table 1 presents the best testing error from 10 experiments with independently initialized network parameters, while the architecture remains fixed in each column. It highlights that the accuracy of the Gauss–Newton method is at least two orders of magnitude higher than that of gradient-based training algorithms and one order higher than L-BFGS.

Additionally, we recorded the total training time required to achieve different tolerances in L^2-error for different methods in Table 2, utilizing a hidden layer with 64 nodes. The results

Table 2 Training time for different training algorithms with a hidden layer width of 64

Stopping Criterion	SGD	ADAM	L-BFGS	Gauss–Newton
L^2-error $< 1e-2$	**2.15s**	3.15s	3.70s	17.25s
L^2-error $< 1e-3$	–	39.76s	**7.54s**	18.20s
L^2-error $< 1e-4$	–	–	83.14s	**58.07s**
L^2-error $< 1e-5$	–	–	–	**132.72s**
Average Iteration Time	**0.0029s**	0.0034s	0.72s	0.17s

The average iteration time is computed by taking the average of 1000 iterations for each training algorithm

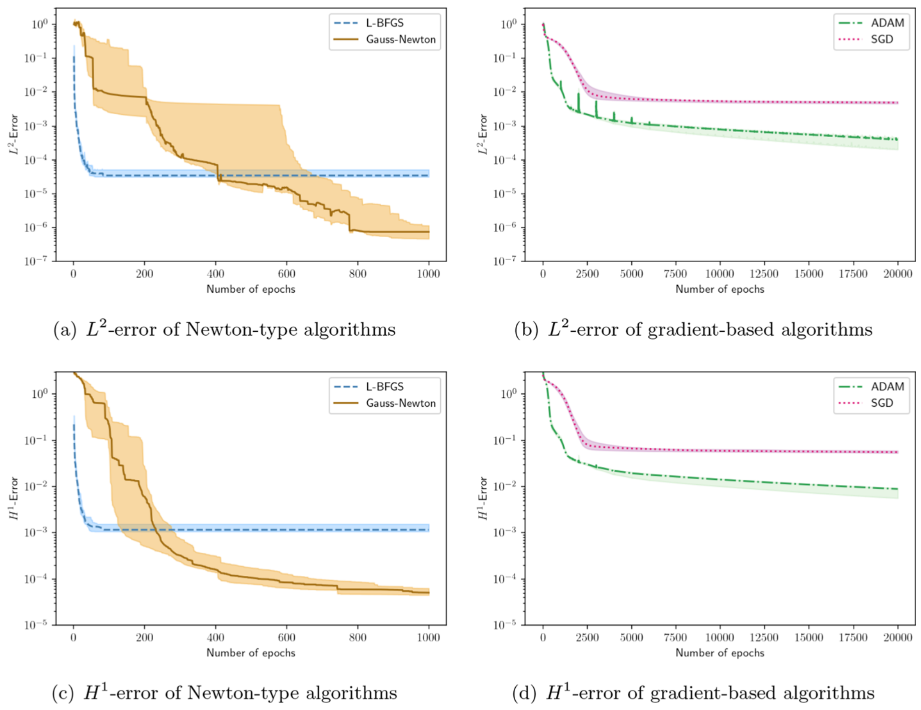


Fig. 1 Testing errors versus iterations. Left: Newton-type training algorithms. The Gauss–Newton method is employed with a back-tracking strategy to determine the learning rate, while L-BFGS is applied with the strong Wolfe condition. Right: Gradient-based training algorithms are applied with an initial learning rate of 1×10^{-3}, which is then halved every 1000 epochs until it reaches 1×10^{-5}

demonstrate that the Gauss–Newton method is more efficient, particularly when higher-order accuracy is essential.

Example 2 We consider the following elliptic equation in two-dimensional space:

$$-\Delta u(x, y) + u(x, y) = f(x, y), \ (x, y) \in \Omega = (0, 1)^2,$$
$$\frac{\partial u(x, y)}{\partial n} = 0, \ (x, y) \in \partial(0, 1)^2.$$

Table 3 Errors in both L^2 and H^1 norms for various training algorithms with a deep architecture 2-20-20-1

Numerical error	SGD	ADAM	L-BFGS	Gauss–Newton
L^2-error	1.13e−2	1.73e−3	4.77e−4	**2.88e−5**
H^1-error	2.02e−1	5.25e−2	1.77e−2	**1.57e−3**

The corresponding variational problem for the 2D equation is given as

$$\min_{v\in H^1} \mathcal{J}(v) = \int_\Omega \left(\frac{1}{2}\nabla v\cdot\nabla v + \frac{1}{2}v^2 - fv\right)\mathrm{d}x\mathrm{d}y$$

By choosing an appropriate source term, we have an exact solution $u(x, y) = \cos(2\pi x)\cos(2\pi y)$. In this example, we employ deep neural networks represented as u_n with ReLU^3 activation function and an architecture of 2-20-20-1, which contains two hidden layers with 20 neurons in each. Therefore, the total number of parameters in this architecture is 501. Similarly, we discretize the energy $\mathcal{J}(u_n)$ using a piecewise Gauss-Legendre rule. We take 160,000 training samples based on the tensored 2-point Gauss-Legendre rule across 200×200 uniform sub-rectangles in $[0, 1]^2$. For the testing dataset used for calculating the numerical errors, we extract 360,000 samples using the same quadrature rule across 300×300 uniform sub-rectangles. The comparison of the four different training algorithms is listed in Table 3, which shows the best approximation results of 10 independent experiments with different neural network initializations. For the training with Newton-type algorithms (L-BFGS, Gauss–Newton) we conduct 2500 epochs, and for the gradient-based algorithms (SGD, ADAM) we extend the training to 20,000 epochs. Further, the training details of the 10 experiments are plotted in Fig. 2, which illustrates the training dynamics in both L^2 and H^1 norms. The shaded area represents the range between the first and third quartiles of these experiments, and lines depict the median values. In both Table 3 and Fig. 2 it shows that the accuracy of the Gauss–Newton method is at least two orders of magnitude higher than the gradient-based algorithms and one order higher than L-BFGS. In addition, it also shows the robustness of the Gauss–Newton algorithm under different dimensions while other training algorithms appear simultaneously more oscillations in the training process when compared with their one-dimensional training dynamics in Fig. 1.

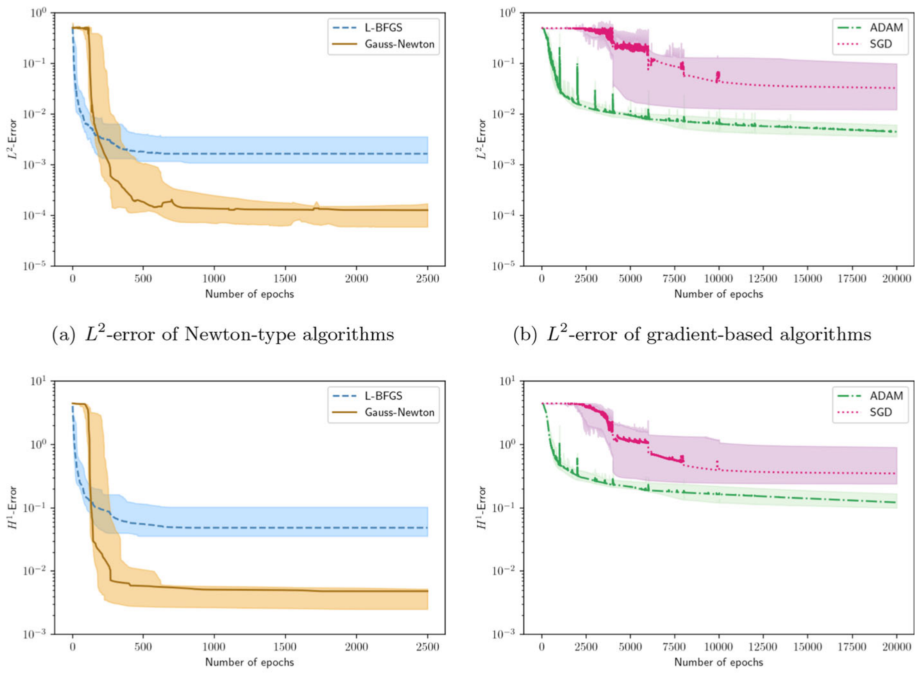
Example 3 (High dimensional example) Let us denote $x := (x_1, x_2, \cdots, x_5)$. A second-order elliptic equation in five-dimensional space is considered as follows:

$$-\Delta u(x) + \pi^2 u(x) = 2\pi^2\sum_{k=1}^{5}\cos(\pi x_k),\ x\in(0,1)^5,$$
$$\frac{\partial u(x)}{\partial n} = 0,\ x\in\partial(0,1)^5.$$

Then the corresponding variational problem becomes

$$\min_{v\in H^1}\mathcal{J}(v) = \int_\Omega\left(\frac{1}{2}\nabla v\cdot\nabla v + \frac{\pi^2}{2}v^2 - 2\pi^2 v\sum_{k=1}^{5}\cos(\pi x_k)\right)\mathrm{d}x.$$

The exact solution of the equation is given by $u(x) = \sum_{k=1}^{5}\cos(\pi x_k)$. In this example we employ a shallow neural network u_n, with 64 neurons in the hidden layer and with ReLU^4 as its activation function, to solve the equation. The total number of parameters



(a) L^2-error of Newton-type algorithms (b) L^2-error of gradient-based algorithms

(c) H^1-error of Newton-type algorithms (d) H^1-error of gradient-based algorithms

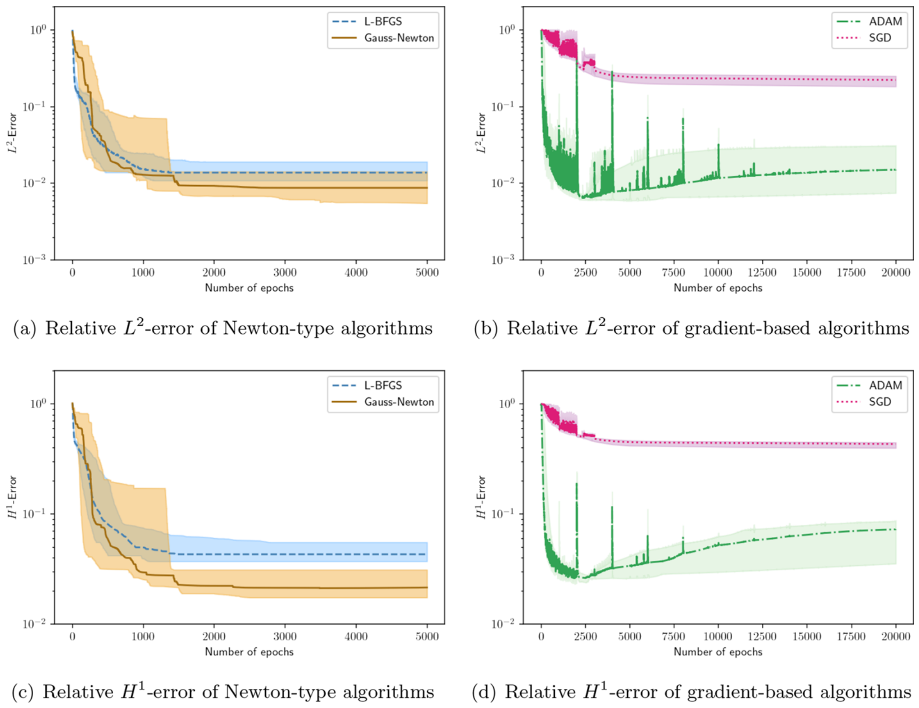
Fig. 2 Testing errors versus iterations. Left Newton-type training algorithms. The Gauss–Newton method is employed with a back-tracking strategy to determine the learning rate, while L-BFGS is applied with the strong Wolfe condition. Right Gradient-based algorithms. ADAM is applied with an initial learning rate of 1×10^{-3}, which is then halved every 2000 epochs until it reaches 1×10^{-5}. SGD is applied with an initial learning rate of 1×10^{-2} which is halved every 2000 epochs before going below 1×10^{-4}

Table 4 Errors in both L^2 and H^1 norms for various training algorithms with a shallow neural network of 64 neurons

Numerical error	SGD	ADAM	L-BFGS	Gauss–Newton
Relative L^2-error	1.74e−1	5.41e−3	5.73e−3	**5.09e−3**
Relative H^1-error	3.42e−1	2.08e−2	2.16e−2	**1.44e−2**

in this architecture is 448. In order to obtain a relatively good accuracy when computing integrals in five-dimensional spaces, we generate 16,000 training samples and 20,000 testing samples in $[0, 1]^5$ using the quasi-Monte-Carlo method associated with the Halton sequence. The training samples will only be used for optimization and the testing samples will be used for computing the relative numerical errors in both L^2 and H^1 norms. Next, we give a comparison between the Gauss–Newton method and other popular training methods. The best approximation results of 10 independent experiments are listed in Table 4. We conduct 5000 epochs when training the Newton-type algorithms (L-BFGS, Gauss–Newton) and for the gradient-based method (SGD, ADAM) the training epochs are extended to 20,000.

Moreover, the training details of these 10 experiments are shown in Fig. 3 below, which gives the training dynamics in the sense of both L^2 and H^1 norm. Again, the line represents



(a) Relative L^2-error of Newton-type algorithms (b) Relative L^2-error of gradient-based algorithms

(c) Relative H^1-error of Newton-type algorithms (d) Relative H^1-error of gradient-based algorithms

Fig. 3 Testing relative errors versus iterations. Left Newton-type training algorithms. The Gauss–Newton method is employed with a back-tracking strategy to determine the learning rate, while L-BFGS is applied with the strong Wolfe condition. Right Gradient-based algorithms. Both ADAM and SGD are applied with an initial learning rate of 1×10^{-3}, which is then halved every 2000 epochs until it reaches 1×10^{-5}

the median values of the numerical relative errors and the shaded area that covers the line represents the range between the first and the third quartiles. Due to the limit of the computational resource and the insufficient accuracy of using the quasi-Monte-Carlo method instead of another high-resolution quadrature rule, all four training algorithms showed less approximation accuracy than their results in lower-dimensional cases like Examples 1 and 2. However, we can still observe a better performance of the Gauss–Newton method than the other three training algorithms. The Gauss–Newton method demonstrates faster convergence and greater stability when compared to the Adam method, which exhibits numerous oscillations during training.

7 Conclusions

In this paper, the Gauss–Newton method has been introduced as a powerful approach for solving partial differential equations (PDEs) using neural network discretization in the variational energy formulation. This method offers significant advantages in terms of convergence and computational efficiency. The comprehensive analysis conducted in this paper has demonstrated the superlinear convergence properties of the Gauss–Newton method, positioning it as another choice for numerical solutions of PDEs. The method converges to semi-regular zeros of the vanishing gradient, indicating its effectiveness in reaching optimal solutions.

Furthermore, we provide the conditions under which the Gauss–Newton method is identical for both variational and L2 minimization problems. Additionally, a variant of the method known as the random Gauss–Newton method has been analyzed, highlighting its potential for large-scale systems. The numerical examples presented in this study reinforce the efficiency and accuracy of the proposed Gauss–Newton method.

In summary, the Gauss–Newton method introduces a novel framework for solving PDEs through neural network discretization. Its convergence properties and computational efficiency make it a valuable tool in the field of computational mathematics. Future research directions may involve incorporating fast linear solvers into the Gauss–Newton method to further enhance its efficiency. This advancement would contribute to the broader utilization of the method and accelerate its application in various domains.

Appendix

In this section, we will introduce some preliminaries for the convergence analysis of Gauss–Newton's method. It is worth noting that the following results do not specify the form of activation functions.

To begin, let us consider the minimization problem (2.7). We seek to find a minimizer v of the functional $\mathcal{J}(w) = a(w, w) - (f, w)$ over a space V of admissible functions. Here, $a(w, z)$ is a symmetric bilinear form defined as $a(w, z) = \int_\Omega \left(\frac{a}{2}\nabla w \cdot \nabla z + \frac{c}{2} wz\right) dx$.

Lemma 5 *Suppose v is the minimizer of (2.7) or the solution of (2.5), and define $\|u\|_a^2 = a(u, u)$. Then, we have*

$$\mathcal{J}(u) - \mathcal{J}(v) = \|u - v\|_a^2, \ \forall u \in V. \tag{7.1}$$

Proof For any $u \in V$, we have the optimality condition for the minimizer v:

$$0 = \delta\mathcal{J}(v)[u] = 2a(v, u) - (f, u), \quad \forall u \in V. \tag{7.2}$$

By a simple computation, we can show that

$$\begin{aligned}\mathcal{J}(u) - \mathcal{J}(v) &= a(u, u) - (f, u) - a(v, v) + (f, v)\\ &= a(u, u) - (f, u) + a(v, v)\\ &= a(u, u) - 2a(v, u) + a(v, v)\\ &= \|u - v\|_a^2.\end{aligned}$$

□

The energy minimization problem (2.7) requires an admissible set with at least H^1 regularity, but different training algorithms may require more regularity. Therefore, the choice of the activation function in DNN models is crucial in ensuring that the admissible set has the required regularity. The ReLU^k activation function with $k \geq 2$ is a suitable choice as it guarantees H^k regularity, which is necessary for the gradient method to work. The ReLU^k-DNN with one hidden layer, denoted by $V_N^k = \{\sum_{i=1}^N a_i \mathrm{ReLU}^k\left(w_i x + b_i\right), a_i, b_i \in \mathbb{R}^1, w_i \in \mathbb{R}^{1\times d}\}$, then we have $V_N^k(\Omega) \subset H^k(\Omega)$, $k \geq 1$.

By considering a general J hidden layer ReLU^k-DNN, we denote the admissible set on Ω for the energy functional as DNN_J. To ensure that the DNN model we use is at least a C^0 function, we assume that $DNN_J \subset V = H^1$. We have an approximation result for DNN_J:

Lemma 6 *For any given $\epsilon > 0$, there exist $J \in \mathbb{N}^+$ and $m \in \mathbb{N}^+$ such that $u(\cdot, \theta) \in DNN_J$, $\theta \in \mathbb{R}^m$, and $\theta^* = \arg\min_\theta \mathcal{J}(u(x,\theta))$ satisfies*

$$\|u(\cdot, \theta^*) - v\|_{L^2(\Omega)} \lesssim \epsilon, \ \left|u(\cdot, \theta^*) - v\right|_{H^1(\Omega)} \lesssim \epsilon. \tag{7.3}$$

We will use the notation "$A \lesssim B$" to denote $A \leq CB$ for some constant C independent of crucial parameters such as θ.

Proof By the Weierstrass theorem in Sobolev space W_p^1 [9] (for $1 \leq p \leq \infty$) and the fact that ReLU^k-DNN functions are piecewise polynomials that belong to H^1, there exist $J \in \mathbb{N}^+$, $m \in \mathbb{N}^+$ and $\theta \in \mathbb{R}^m$ such that $u(\cdot, \theta) \in DNN_J$ and

$$\|u(\cdot, \theta) - v\|_a \lesssim \|u(\cdot, \theta) - v\|_{H^1(\Omega)} < \epsilon. \tag{7.4}$$

In particular, when considering ReLU^1-DNN functions, one can refer to [16] for estimations of a sufficiently large depth J and a sufficiently large number of neurons required to represent all the piecewise linear functions in $\mathbb{R}^d$.

Next, as $DNN_J \subset V$, we have

$$\mathcal{J}(u(\cdot, \theta)) \geq \mathcal{J}(u(\cdot, \theta^)) \geq \mathcal{J}(v), \tag{7.5}$$

which, together with Lemma 5, implies that

$$\begin{aligned}\|u(\cdot, \theta) - v\|_{L^2(\Omega)} &\lesssim \|u(\cdot, \theta) - v\|_a \leq \|u(\cdot, \theta) - v\|_a < \epsilon \\ \left|u(\cdot, \theta) - v\right|_{H^1(\Omega)} &\lesssim \|u(\cdot, \theta^*) - v\|_a \leq \|u(\cdot, \theta) - v\|_a < \epsilon.\end{aligned}$$

□

Based on the lemmas above, we can estimate the residual function as follows:

Lemma 7 *For $\forall \epsilon > 0$, there exist $\delta > 0$, $J \in \mathbb{N}^+$ and $m \in \mathbb{N}^+$, such that $\theta \in \mathbb{R}^m$, $u(\cdot, \theta) \in DNN_J \subset H^1(\Omega)$, and $\|\theta - \theta^*\| < \delta$, it holds*

$$\|u(\cdot, \theta) - v\|_{L^2(\Omega)} < \epsilon \tag{7.6}$$

and

$$\int_\Omega (\nabla u(x,\theta) \cdot \nabla w(x) + u(x,\theta) w(x) - f(x) w(x))\, dx < C\epsilon, \quad \forall w \in H^1(\Omega) \tag{7.7}$$

where $C > 0$ is independent of θ.

Proof By the continuity of $u(\cdot, \theta)$ with respect to θ, it is easy to see from Lemma 6 that

$$\|u(\cdot, \theta) - v\|_{L^2(\Omega)} \leq \|u(\cdot, \theta) - u(\cdot, \theta^*)\|_{L^2(\Omega)} + \|u(\cdot, \theta^*) - v\|_{L^2(\Omega)} < \epsilon. \tag{7.8}$$

Next, we have for $\forall w \in H^1(\Omega)$,

$$\begin{aligned}(r(u(\cdot, \theta)), w)_{L^2(\Omega)} &= \int_\Omega (a\nabla u(x,\theta) \cdot \nabla w(x) + c u(x,\theta) w(x) - f(x) w(x))\, dx \\ &= \int_\Omega (a\nabla (u(x,\theta) - v(x)) \cdot \nabla w(x) + c(u(x,\theta) - v(x)) w(x))\, dx \\ &\lesssim \left|u(\cdot, \theta) - v\right|_{H^1(\Omega)} \left|w\right|_{H^1(\Omega)} + \|u(\cdot, \theta) - v\|_{L^2(\Omega)} \|w\|_{L^2(\Omega)}.\end{aligned}$$

Since $w \in H^1(\Omega)$, we can conclude form Lemma 6 that the estimation (7.7) is valid. □

Next, we have Wedin's Theorem about a perturbed matrix.

Lemma 8 *(Wedin, see [33] or [7]) Let $A \in \mathbb{R}^{m\times n}$ and $B = A+E$ with $rank(B) = rank(A)$. Then in any unitary invariant norm $\|\cdot\|$,*

$$\|B^\dagger - A^\dagger\| \le \mu \|A^\dagger\|_2 \|B^\dagger\|_2 \|E\| \tag{7.9}$$

for some moderate constant $\mu > 0$ that depends on the norm used. Moreover, if

$$\|E\|_2 < \frac{1}{\|A^\dagger\|_2}, \tag{7.10}$$

then the inverse of B can be bounded by

$$\|B^\dagger\|_2 \le \frac{\|A^\dagger\|_2}{1 - \|A^\dagger\|_2 \|E\|_2} \tag{7.11}$$

Remark 1 A unitarily invariant norm $\|\cdot\|$ satisfies $\|A\| = \|UAV\|$ for any unitary matrices U and V. Specifically, the 2-norm, the Frobenius norm, and many other norms related to the singular value are unitarily invariant norms.

The following lemmas are just from [36].

Lemma 9 *For $\mathbb{F} = \mathbb{C}$ or $\mathbb{R}$, let $\mathbf{z} \mapsto \phi(\mathbf{z})$ be a continuous injective mapping from an open set Ω in $\mathbb{F}^n$ to $\mathbb{F}^m$. At any $\mathbf{z}_0 \in \Omega$, there is an open neighborhood Δ of $\phi(\mathbf{z}_0)$ in $\mathbb{F}^m$ such that, for every $\mathbf{b} \in \Delta$, there exists a $\mathbf{z_b}$ in Ω and an open neighborhood Σ_0 of $\mathbf{z_b}$ with*

$$\|\mathbf{b} - \phi(\mathbf{z_b})\|_2 = \min_{\mathbf{z}\in\Sigma_0} \|\mathbf{b} - \phi(\mathbf{z})\|_2.$$

Further assume ϕ is differentiable in Ω. Then

$$\phi_{\mathbf{z}}(\mathbf{z_b})^\dagger (\mathbf{b} - \phi(\mathbf{z_b})) = \mathbf{0}.$$

Lemma 10 *(Stationary Point Property) Let $\theta \mapsto \nabla L(\theta)$ be a smooth mapping with a semiregular zero θ^* and $r = rank(\mathbf{H}(\theta^*))$. Then there is an open neighborhood Ω_* of θ^* such that, for any $\hat{\theta} \in \Omega_*$, the equality $(\mathbf{H})(\hat{\theta})^\dagger_{rank\text{-}r} \nabla L(\hat{\theta}) = \mathbf{0}$ holds if and only if $\hat{\theta}$ is a semiregular zero of ∇L in the same branch of θ^*.*

Lemma 11 *(Local Invariance of Semiregularity) Let θ^* be a semiregular zero of a smooth mapping ∇L. Then there is an open neighborhood Δ_* of θ^* such that every $\hat{\theta} \in \Delta_* \cap (\nabla L)^{-1}(\mathbf{0})$ is a semiregular zero of ∇L in the same branch of θ^*.*

Acknowledgements We thank Dr. Yiran Wang from Purdue University for valuable discussions and contributing efforts to the first version of numerical experiments.

Funding WH and QH are supported by NIH via 1R35GM146894.

Data Availability Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of interest There is no Conflict of interest.

References

1. Allen-Zhu, Z., Li, Y., Song Z.: A convergence theory for deep learning via over-parameterization. In: International Conference on Machine Learning, PMLR, ,pp. 242–252 (2019)
2. Arora, S., Du, S., Hu, W., Li, Z., Wang, R.: Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In: International Conference on Machine Learning, PMLR, pp. 322–332 (2019)
3. Barron, A.R.: Universal approximation bounds for superpositions of a sigmoidal function. IEEE Trans. Inf. Theory **39**(3), 930–945 (1993)
4. Bates, D.J., Sommese, A.J., Hauenstein, J.D., Wampler, C.W.: Numerically Solving Polynomial Systems with Bertini. SIAM, Philadelphia (2013)
5. Cai, S., Mao, Z., Wang, Z., Yin, M., Karniadakis, G.E.: Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta Mech. Sin., pp. 1–12 (2022)
6. 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 (2019). arXiv:1905.11675
7. Castro-González, N., Dopico, F.M., Molera, J.M.: Multiplicative perturbation theory of the Moore–Penrose inverse and the least squares problem. Linear Algebra Appl. **503**, 1–25 (2016)
8. Chen, Q., Hao, W.: Randomized Newton's method for solving differential equations based on the neural network discretization. J. Sci. Comput. **92**(2), 49 (2022)
9. De Branges, L.: The Stone–Weierstrass theorem. Proc. Am. Math. Soc. **10**(5), 822–824 (1959)
10. Dissanayake, M.W.M.G., Phan-Thien, N.: Neural-network-based approximations for solving partial differential equations. Commun. Numer. Methods Eng. **10**(3), 195–201 (1994)
11. Evans, L.C.: Partial Differential Equations, vol. 19. American Mathematical Society, Providence (2022)
12. Greville, T.N.E.: Note on the generalized inverse of a matrix product. SIAM Rev. **8**(4), 518–521 (1966)
13. Yiqi, G., Yang, H., Zhou, C.: SelectNet: self-paced learning for high-dimensional partial differential equations. J. Comput. Phys. **441**, 110444 (2021)
14. Hammersley, J.: Monte Carlo Methods. Springer, Berlin (2013)
15. Han, J., Jentzen, A., Weinan, E.: Solving high-dimensional partial differential equations using deep learning. Proc. Natl. Acad. Sci. **115**(34), 8505–8510 (2018)
16. He, J., Li, L., Jinchao, X., Zheng, C.: ReLU deep neural networks and linear finite elements. J. Comput. Math. **38**(3), 502–527 (2020)
17. Huang, Y., Hao, W., Lin, G.: HomPINNs: Homotopy physics-informed neural networks for learning multiple solutions of nonlinear elliptic differential equations. Comput. Math. Appl. **121**, 62–73 (2022)
18. Khoo, Y., Jianfeng, L., Ying, L.: Solving parametric PDE problems with artificial neural networks. Eur. J. Appl. Math. **32**(3), 421–435 (2021)
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization, arXiv preprint (2014). arXiv:1412.6980
20. Klusowski, J.M., Barron, A.R.: Approximation by combinations of ReLU and squared ReLU ridge functions with ℓ^1 and ℓ^0 controls. IEEE Trans. Inf. Theory **64**(12), 7649–7656 (2018)
21. Jianfeng, L., Yulong, L.: A priori generalization error analysis of two-layer neural networks for solving high dimensional Schrödinger eigenvalue problems. Commun. Am. Math. Soc. **2**(01), 1–21 (2022)
22. Jianfeng, L., Shen, Z., Yang, H., Zhang, S.: Deep network approximation for smooth functions. SIAM J. Math. Anal. **53**(5), 5465–5506 (2021)
23. Müller, J., Zeinhofer, M.: Achieving high accuracy with PINNs via energy natural gradient descent (2023)
24. Pang, G., Lu, L., Em Karniadakis, G.: fPINNs: Fractional physics-informed neural networks. SIAM J. Sci. Comput. **41**(4), A2603–A2626 (2019)
25. Rahaman, N, Baratin, A, Arpit, D, Draxler, F, Lin, M, Hamprecht, F, Bengio, Y, Courville, A: On the spectral bias of neural networks. In: International Conference on Machine Learning, PMLR, pp. 5301–5310 (2019)
26. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019)
27. Shen, Z., Yang, H., Zhang, S.: Optimal approximation rate of ReLU networks in terms of width and depth. Journal de Mathématiques Pures et Appliquées **157**, 101–135 (2022)
28. Siegel, J.W., Hong, Q., Jin, X., Hao, W., Jinchao, X.: Greedy training algorithms for neural networks and applications to PDEs. J. Comput. Phys. **484**, 112084 (2023)
29. Siegel, J.W., Jinchao, X.: Approximation rates for neural networks with general activation functions. Neural Netw. **128**, 313–321 (2020)
30. Siegel, J.W., Jinchao, X.: High-order approximation rates for shallow neural networks with cosine and ReLUk activation functions. Appl. Comput. Harmon. Anal. **58**, 1–26 (2022)

31. Siegel, J.W., Jinchao, X.: Sharp bounds on the approximation rates, metric entropy, and n-widths of shallow neural networks. Found. Comput. Math. 1–57 (2022)
32. Sirignano, J., Spiliopoulos, K.: DGM: A deep learning algorithm for solving partial differential equations. J. Comput. Phys. **375**, 1339–1364 (2018)
33. Wedin, P.Å.: Perturbation theory for pseudo-inverses. BIT Numer. Math. **13**, 217–232 (1973)
34. Jinchao, X.: Finite neuron method and convergence analysis. Commun. Comput. Phys. **28**(5), 1707–1745 (2020)
35. Yu, B., Weinan, E.: The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun. Math. Stat. **6**(1), 1–12 (2018)
36. Zeng, Z.: A newton's iteration converges quadratically to nonisolated solutions too, Math. Comput. (2023)
37. Zou, D., Cao, Y., Zhou, D., Quanquan, G.: Gradient descent optimizes over-parameterized deep ReLU networks. Mach. Learn. **109**(3), 467–492 (2020)