Performance Analysis of Gradient Neural Network Exploited for Online Time-Varying Matrix Inversion

Yunong Zhang, Member, IEEE, Ke Chen, and Hong-Zhou Tan

Abstract—This technical note presents theoretical analysis and simulation results on the performance of a classic gradient neural network (GNN), which was designed originally for constant matrix inversion but is now exploited for time-varying matrix inversion. Compared to the constant matrix-inversion case, the gradient neural network inverting a time-varying matrix could only approximately approach its time-varying theoretical inverse, instead of converging exactly. In other words, the steady-state error between the GNN solution and the theoretical/exact inverse does not vanish to zero. In this technical note, the upper bound of such an error is estimated firstly. The global exponential convergence rate is then analyzed for such a Hopfield-type neural network when approaching the bound error. Computer-simulation results finally substantiate the performance analysis of this gradient neural network exploited to invert online time-varying matrices.

Index Terms—Global exponential convergence rate, gradient neural networks, performance analysis, residual error bound, time-varying matrix inversion.

I. INTRODUCTION

Online matrix-inversion problems are ubiquitous in science and engineering, e.g., in optimization [1], [2], signal-processing [3]–[5], statistics [1], [4], [5], electromagnetic systems [6], and robot control [7], [8]. There are two general types of solution to the problem. One is the numerical algorithms performed on digital computers (i.e., on our computers of today). Usually, the minimal arithmetic operations for such a numerical algorithm are proportional to the cube of matrix dimension \( n \), i.e., \( O(n^3) \) operations with \( n \) denoting here the dimension of square matrix \( A \) to be inverted. Because of serial-processing nature, such numerical algorithms may not be efficient enough for large-scale online or real-time applications. For example, as shown in [5], it takes on average one hour to invert a 60000-dimensional matrix once, though the algorithm used therein is of \( O(n^2) \) operations. Being the second general type of solution, many parallel-processing computational methods have been developed, analyzed, and implemented on specific architectures [2], [3], [7]–[15]. The dynamic-system approach is one of the important parallel-processing methods for solving online matrix-inverses. Because of the in-depth research in recurrent neural networks, various dynamic and analog solvers have recently been developed and investigated [3], [8], [9], [11]–[14]. The neural-dynamic approach is now thus regarded as a powerful alternative to online computation of matrix-related problems owing to its parallel distributed nature and convenience of hardware/circuit implementation [2], [9], [15].

The neural-network approach for matrix inversion is designed conventionally based on the gradient-descent method in optimization [1], [2] and intrinsically for constant matrices. Specifically speaking, to solve for the inverse of constant nonsingular matrix \( A \in \mathbb{R}^{n \times n} \), we can design a classic gradient neural network (or termed, gradient-based neural network) as follows.

- Firstly, it is well known that the following defining equation of matrix inverse \( A^{-1} \in \mathbb{R}^{n \times n} \) could be given and exploited:
  \[
  AX - I = 0 \quad \text{or} \quad XA - I = 0
  \] (1)
  where \( I \in \mathbb{R}^{n \times n} \) denotes the identity matrix, and \( X \in \mathbb{R}^{n \times n} \) denotes the unknown matrix to be solved which corresponds to the matrix inverse \( A^{-1} \).

- Secondly, to solve for \( X \) via a dynamic-system approach, we can define a scalar-valued norm-based error function, \( \mathcal{E}(t) = \| AX(t) - I \|_F^2 / 2 \), where \( \| A \|_F = \sqrt{\text{tr}(A^T A)} \) denotes the Frobenius norm of matrix \( A \). In addition, \( \text{tr}(\cdot) \) denotes the trace operator of a matrix, which equals the sum of the main diagonal elements of such a matrix. Note that a minimum point of residual-error function \( \mathcal{E}(t) = \| AX(t) - I \|_F^2 / 2 \) is achieved with \( \mathcal{E}(t) = 0 \), if and only if \( X(t) \) is the exact solution of (1) [in other words, \( X(t) = X^* := A^{-1} \)].

- Thirdly, a computational scheme could be designed to evolve along a descent direction of this error function \( \mathcal{E}(t) \), until the minimum point \( X^* \) is reached. Note that a typical descent direction is the negative gradient of \( \mathcal{E}(t) \), i.e., \( - \frac{\partial \mathcal{E}}{\partial X} \in \mathbb{R}^{n \times n} \).

- Fourthly, it follows from the above three steps that we could have the lemma below.

**Lemma 1:** Consider nonsingular matrix \( A \in \mathbb{R}^{n \times n} \). Define residual-error function \( \mathcal{E} = \| AX(t) - I \|_F^2 / 2 \in \mathbb{R} \). The derivative of \( \mathcal{E} \) with respect to \( X \in \mathbb{R}^{n \times n} \) could simply be derived as \( \frac{\partial \mathcal{E}}{\partial X} = A^T (AX(t) - I) \in \mathbb{R}^{n \times n} \).

**Proof:** See Appendix.

- Finally, it follows from the above and according to design formula \( \dot{X}(t) = -\gamma \frac{\partial \mathcal{E}}{\partial X} \), we could have the dynamic equation of conventional gradient neural network as below for matrix inversion (which is a Hopfield-type neural network as well) [3], [8], [9], [11], [12]:
  \[
  \dot{X}(t) = -\gamma A^T (AX(t) - I), \quad \text{with} \quad X(0) = X_0 \in \mathbb{R}^{n \times n} \] (2)
  for \( t \in [0, +\infty) \), where design parameter \( \gamma > 0 \), being an inductance parameter or the reciprocal of a capacitance parameter, could be set as large as the hardware permits, or selected appropriately for simulative and/or experimental purposes. The network structure of GNN (2) is shown in Fig. 1, where \( x_{ij} \) denotes the \( i \) th neuron (or termed, the \( i \)th-element state) of the neural network, \( \forall \ i, j \in \{1, 2, \ldots, n\} \). For more details, please also refer to [3], [8], [9], [11], [12].

Now, the negative-gradient design method for recurrent neural networks has been shown very clearly as in the above, and its resultant GNN model for matrix inversion has also been derived as in (2). It could be proved [8], [16] that, for constant nonsingular matrix \( A \), the state matrix \( X(t) \) of gradient neural network (2) could globally exponentially converge to its inverse, \( A^{-1} \). However, as shown by Zhang et al. in [13], [14], [17], when applied to time-varying matrix inversion [i.e., \( A \in \mathbb{R}^{n \times n} \) is a time-varying matrix, \( A(t) \)], the gradient neural network (2) may generate a considerably large solution-error (not only in magnitudes, but also lagging behind in time phase). For example,