RNN for solving linear matrix equations

Predrag Stanimirović[1]

[1]Department of Computer Science, Faculty of Sciences and Mathematics, University of Niš,
pecko@pmf.ni.ac.rs

We are concerned with the solution to the general time-invariant matrix equation $AV(t)B = D$ and the time-varying matrix equation $A(t)V(t)B(t) = D(t)$ by means of gradient based neural network (GNN) model, called the GNNABD model. The resulting matrix generated by the GNNABD model is defined by the choice of the initial state and coincides with the general solution of the matrix equation $AVB = D$. Several particular appearances of this matrix equation and their applications in approximatting various inner and outer inverses are considered. Particularly, two particular cases of the general GNNABD model, globally convergent to the Moore-Penrose inverse and the Drazin inverse are defined and investigated theoretically and numerically. The influence of various nonlinear activation functions on several variants of the GNNABD model are investigated.

Required matrix equations can be solved by the generalized nonlinearly activated GNN model (GGNN model) which is applicable in both time-varying and time-invariant case and possesses the form

$$\frac{\mathrm{d}V(t)}{\mathrm{d}t} = \dot{V}(t) = \gamma A^{\mathrm{T}}\mathcal{F}(D - AV(t)B)B^{\mathrm{T}}. \tag{1}$$

The matrix-valued activation function $\mathcal{F}(E)$, $E = (e_{ij})$, is defined as $(f(e_{ij}))$, $i, j = 1, 2, \dots, n$, where $f(\cdot)$ is a scalar-valued monotonically-increasing odd function.

Theorem 1. *Assume that real matrices $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{p\times q}$ and $D \in \mathbb{R}^{m\times q}$ satisfy*

$$AA^{(1)}DB^{(1)}B = D, \tag{2}$$

for some inner inverses $A^{(1)}$ and $B^{(1)}$. If an odd and monotonically increasing function $f(\cdot)$ is used to define the array activation function $\mathcal{F}(\cdot)$, then the state matrix $V(t) \in \mathbb{R}^{n\times m}$ of the GNNABD model (1) *satisfies $AV(t)B \to D$ when $t \to +\infty$, for an arbitrary initial state matrix $V(0)$.*

Theorem 2. *Assume that the real matrices $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{p\times q}$ and $D \in \mathbb{R}^{m\times q}$ satisfy*

$$AA^{\dagger}DB^{\dagger}B = D. \tag{3}$$

Then the unknown matrix $V(t)$ of the model GNNABD is convergent when $t \to +\infty$ and has the limit value

$$\tilde{V} = A^{\dagger}DB^{\dagger} + V(0) - A^{\dagger}AV(0)BB^{\dagger} \tag{4}$$

for every initial matrix $V(0) \in \mathbb{R}^{n\times p}$.

Some appearances of the general linear matrix equation $AXB = D$ are considered.
Conditions for the existence and representations of $\{2\}$-, $\{1\}$- and $\{1,2\}$-inverses which satisfy certain conditions on ranges and/or null spaces are introduced in [4]. These representations are applicable to complex matrices and involve solutions of certain matrix equations.

Solution $\tilde{V}$ of the matrix equation

$$BV(t)CAB = B$$

defined by the GNABD model

$$\dot{V}(t) = B^{\mathrm{T}}\mathcal{F}(B - BV(t)CAB)(CAB)^{\mathrm{T}} \tag{5}$$

gives $\tilde{V} \in (CAB)\{1\}$. Then $X = B\tilde{V}C$ gives various representations of outer inverses, according to Urguhart formula.

Algorithms arising from the introduced representations are developed. Particularly, these algorithms can be used to compute the Moore-Penrose inverse, the Drazin inverse and the usual matrix inverse. The implementation of introduced algorithms is defined on the set of real matrices and it is based on the Simulink implementation of GNN models for solving the involved matrix equations. In this way, we develop computational procedures which generate various classes of inner and outer generalized inverses on the basis of resolving certain matrix equations. As a consequence, some new relationships between the problem of solving matrix equations and the problem of numerical computation of generalized inverses are established. Theoretical results are applicable to complex matrices and the developed algorithms are applicable to both the time-varying and time-invariant real matrices.

The general computational pattern for commuting generalized inverses is based on the general representation $B(CAB)^{(1)}C$, where the matrices A, B, C satisfy various conditions imposed in the proposed algorithms.

The general computational pattern for computing generalized inverses can be described in two main steps:
(1) Solve appropriate linear matrix equation $BUCAB = B$ with respect to U using

GNNABD model.
(2) Compute the matrix product BUC.

GNN models defined in [1, 2, 3, 5] can be derived as modifications of some appearances of the GNNABD model.

The GNNABD model for solving the matrix equation $AA^{\mathrm{T}}VA^{\mathrm{T}}A = A$ is given by

$$\dot{V} = \gamma AA^{\mathrm{T}}\mathcal{F}\left(A - AA^{\mathrm{T}}V(t)A^{\mathrm{T}}A\right)A^{\mathrm{T}}A, \tag{6}$$

and it is called as GNNABD-MP.

Theorem 3. *Let $\tilde{V}(t)$ be a solution of the model* (6). *Then the matrix $X(t) = A^{\mathrm{T}}\tilde{V}(t)A^{\mathrm{T}}$ converges to the Moore-Penrose inverse $A^{\dagger}$ for every initial matrix $V(0)$.*

Keywords: Recurrent neural network; pseudoinverse; outer inverse; Dynamic equation; Activation function.

References

[1] P. S. Stanimirović, I. S. Živković and Y. Wei, Recurrent neural network for computing the Drazin inverse, IEEE Trans. Neural Netw. Learn. Syst. **26** (2015), 2830–2843.

[2] P. S. Stanimirović, I. Živković and Y. Wei, Neural network approach to computing outer inverses based on the full rank representation, Linear Algebra Appl. **501** (2016), 344–362.

[3] P. S. Stanimirović, I. S. Živković and Y.Wei, Recurrent neural network for computing the Drazin inverse, IEEE Trans. Neural Netw. Learn. Syst. **26** (2015), 2830–2843.

[4] P. S. Stanimirović, M. Ćirić, I. Stojanović and D. Gerontitis, Conditions for existence, representations and computation of matrix generalized inverses, Complexity **2017**, Article ID 6429725, 27 pages, https://doi.org/10.1155/2017/6429725.

[5] I. Živković, P. S. Stanimirović and Y. Wei, Recurrent neural network for computing outer inverses, Neural Computation **28**(5) (2016), 970–998.