European Proceedings of
Computers and Technology
EpCT

www.europeanproceedings.com

e-ISSN: 2672-8834

DOI: 10.15405/epct.23021.31

HMMOCS 2022
International Workshop "Hybrid methods of modeling and optimization in complex systems"

GRADIENT NEURAL DYNAMICS BASED ON MODIFIED ERROR FUNCTION

Predrag S. Stanimirović (a)*, Dimitrios Gerontitis (b), Nataša Tešić (c),
Vladimir L. Kazakovtsev (d), Vladislav Stasiuk (e), Xinwei Cao (f)
*Corresponding author

(a) University of Niš, Faculty of Sciences and Mathematics, Niš, 18000, Serbia,
Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University; 79, Prosp. Svobodny, Krasnoyarsk, 660041, Russia, pecko@pmf.ni.ac.rs
(b) Department of Information and Electronic Engineering, International Hellenic University, Thessaloniki, Greece, dimitrios_gerontitis@yahoo.gr
(c) Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Novi Sad, 21000, Serbia, dmi.3d.19@student.pmf.uns.ac.rs
(d) Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, 79, Prosp. Svobodny, Krasnoyarsk, 660041, Russia, vokz@bk.ru
(e) Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, 79, Prosp. Svobodny, Krasnoyarsk, 660041, Russia, vstasyuk@sfu-kras.ru
(f) School of Business, Jiangnan University, Lihu Blvd, Wuxi, 214122, China

Abstract

The present study is devoted to methods for the numerical solution to the system of equations $AXB=D$. In the case certain conditions are met, the classical gradient neural network (GNN) dynamics obtains fast convergence. However, if those conditions are not satisfied, solution to the equation does not exist and therefore the error function $E(t):=AV(t)B-D$ cannot be equal to zero, which increases the CPU time required for the calculation. In this paper, the solution to the matrix equation $AXB = D$ is studied using the novel Gradient Neural Network (GGNN) model, termed as GGNN(A,B,D). The GGNN model is developed using a gradient of the error matrix used in the development of the GNN model. The proposed method uses a novel objective function that is guaranteed to converge to zero, thus reducing the execution time of the Simulink implementation. The GGNN-based dynamical systems for computing generalized inverses are also discussed. The conducted computational experiments have shown the applicability and advantage of the developed method.

Keywords: Gradient neural network, generalized inverses, moore-penrose inverse, linear matrix equations

https://doi.org/10.15405/epct.23021.31
Corresponding Author: Predrag S. Stanimirović
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2672-8834

1. Introduction

In this study work, we deal with the real-time solutions of the general linear matrix equation (GLME) $AXB = D$ utilizing the gradient-established neural network (GNN) dynamical evolution, termed as GNN(A,B,D). Previously, GNN models were described and investigated in the works of Wang (1992, 1993), Zhang et al. (2009), Wang (1997), Wei (2000), Wang and Li (1994), Ding and Chen (2005), Zhang and Chen (2008). Convergence investigation indicates that the output of GNN(A,B,D) is specified by the choice of the initial state and belongs to the set of theoretical solutions to $AXB=D$. Also, this work contains diverse applications of the GNN(A,B,D) design described in Stanimirović et al. (2017, 2019, 2022) and improvements of proposed models for solving linear systems $Ax=b$ described in Urquhart (1968). Most applications examined the impact of activation functions on the convergence rate of GNN(A,B,D) theoretically and by means of simulation experiments. In the last section, we will test the novel gradient-based GNN formula (GGNN), which includes a different error matrix than the GNN model.

The implementation is defined on the set of real matrices and is based on making simulations of considered GNN-based models for solving matrix equations. The numerical experiments are tested in MATLAB Simulink.

2. Problem Statement

Recurrent neural networks (RNNs) form an essential class of methods for solving the matrix equations. RNNs are splitted into two categories: Gradient Neural Networks (GNN) and Zhang (or Zeroing) Neural Networks (ZNN). The GNN flow is explicit and efficient in solving time-invariant problems, which assumes constant coefficients matrices in underlying matrix equations. ZNN models are mostly implicit and efficient in solving time-varying problems (entries involved in coefficient matrices of the equations are functions of time $t \in \mathrm{R}, t > 0$).

General GNN neural dynamics are used to solve $AXB = D$. The dynamical evolution is developed based on the residual $E(t) := AV(t)B - D$, such that $V(t)$ is an unknown state-variable matrix which converges to the required matrix X of the GLME $AXB = D$. The goal function $\varepsilon(t) = \left||D - AV(t)B|\right|_F^2/2$, is the function of the Frobenius norm. The gradient matrix of the objective $\varepsilon(t)$ is computed as

$$\frac{\partial \varepsilon(\mathrm{V(t)})}{\partial \mathrm{V}} = \frac{1}{2}\frac{\partial \left||D-AV(t)B|\right|_{\mathrm{F}}^2}{\partial \mathrm{V}} = -\mathrm{A}^{\mathrm{T}}(D - AV(t)B)B^{\mathrm{T}}.$$

By the GNN evolution, one derives the dynamics

$$\dot{V}(t) = \frac{dV(t)}{dt} = -\gamma\left(\frac{\partial \varepsilon(V(t))}{\partial V}\right) = -\gamma A^{\mathrm{T}}(AV(t)B - D)B^{\mathrm{T}}, \quad (1)$$

in which $\dot{V}(t)$ is the time derivative and $\gamma > 0$ is a positive gain parameter necessary for accelerating convergence. A faster convergence is achieved by increasing the value γ. We denote this model as GNN(A,B,D). As already mentioned, the considered matrix-valued residual which cancels out over time is $E(\mathrm{t}) = D - AV(t)B$, such that $V(t)$ is the activation state variables matrix. The nonlinear GNN(A,B,D) design is defined by

https://doi.org/10.15405/epct.23021.31
Corresponding Author: Predrag S. Stanimirović
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2672-8834

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

The function array $\mathcal{F}(\mathrm{C})$ includes any odd and monotonically-increasing activation function, which is applicable to each individual entry of its own matrix arguments.

The error function $E_G(t)$ is introduced using analogy with gradient-descent iterations for unconstrained nonlinear optimization. The residual $E(t) = AV(t)B - D$ is forced to the null matrix Stanimirović et al. (2018). The gradient of

$$\varepsilon_V = \frac{||E(t)||_F^2}{2} = \frac{||AV(t)B - D||_F^2}{2}$$

is equal to

$$\frac{\partial \varepsilon_V}{\partial V} = \nabla \varepsilon_V = A^{\mathrm{T}}(AV(t)B - D)B^{\mathrm{T}}.$$

The GNN dynamic evolution minimizes $||AV(t)B - D||_{\mathrm{F}}^2$ and it is established on the direct correlation (1) among $\dot{V}(t)$ and $\nabla \varepsilon_V$ (Wang, 1993; Zhang et al., 2009; Wang, 1997).

3. Research Questions

The subsequent motivation questions were posed during the study:

- How to increase the speed of obtaining numerical solution of $AXB = D$?
- How to define the GNN design for solving $AV(t)B = D$ established on the residual matrix $E_G(t) \coloneqq \nabla \varepsilon_V(t) = A^{\mathrm{T}}(AV(t)B - D)B^{\mathrm{T}} = A^{\mathrm{T}}E(t)B^{\mathrm{T}}$?
- What is the convergence speed of the new dynamics which is developed on the basis of $E_G(t)$?
- What is the numerical behaviour of the new model?

4. Purpose of the Study

The intention of this research is to find new GNN-type dynamical system based on a novel error functions.

Standard GNN design solves the GLME $AXB - D = 0$ under the condition $AA^{\dagger}DB^{\dagger}B = D$ (Stanimirović & Petković, 2018). Our aim is to avoid this constraint and originate dynamical evolutions based on the error function that tends to zero without restrictions.

Our motivation in defining new error function arises from gradient-descent methods for minimizing nonlinear multivariate functions. Our leading idea is the fact that the GLME $\nabla \varepsilon_V = A^{\mathrm{T}}(AV(t)B - D)B^{\mathrm{T}} = 0$ is convergent without restrictions. Results about solvability of GLME and general solutions are described in Wang et al. (2018).

5. Research Methods

https://doi.org/10.15405/epct.23021.31
Corresponding Author: Predrag S. Stanimirović
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2672-8834

To improve the standard GNN design, we introduced a new GGNN dynamical flow. More precisely, instead of using the classical error matrix $E(t) = D - AV(t)B$, we took for error matrix the right hand side of GNN model (1), i.e., the gradient of ε of the GNN formula. That leads us to a new evolution

$$E_G(t) = \frac{1}{2}\frac{\partial \varepsilon(V(t))}{\partial V} = \frac{1}{2}\frac{\partial ||D-AV(t)B||_F^2}{\partial V} = -A^{\mathrm{T}}(D - AV(t)B)B^{\mathrm{T}}.$$

We denote new error matrix with E_G, because the error function would take the value of the gradient and seek minimization over the gradient.

Next step is to define new model with this error matrix, called *Gradient* GNN, or shortly GGNN. Let us define goal function $\varepsilon_G = ||E_G||_F^2$, whose gradient is equal to

$$\frac{\partial \varepsilon_G(V(t))}{\partial V} = \frac{\partial \|A^{\mathrm{T}}(D-AV(t)B)B^{\mathrm{T}}\|_F^2}{\partial V} = -2A^{\mathrm{T}}A(A^{\mathrm{T}}(D - AV(t)B)B^{\mathrm{T}})BB^{\mathrm{T}}.$$

Using the GNN-type evolution design, the dynamical system for GGNN formula is expanded as

$$\dot{V}_G(t) = \frac{\mathrm{d}V_G(t)}{\mathrm{d}t} = \gamma A^{\mathrm{T}}A(A^{\mathrm{T}}(D - AV(t)B)B^{\mathrm{T}})BB^{\mathrm{T}},$$

where $\gamma > 0$ scales the convergence. For a faster convergence, it is better to use greater values of γ, as in the GNN model. Hence, the corresponding nonlinear GGNN model is given by the following dynamics:

$$\dot{V}(t) = \gamma A^{\mathrm{T}}A\mathcal{F}(A^{T}(D - AV(t)B)B^{\mathrm{T}})BB^{\mathrm{T}}, \tag{2}$$

where $\mathcal{F}(\ \ )$ is an odd and monotonically increasing function array based on arbitrary monotonically increasing odd activation function $f(\cdot)$.

Figure 1 represents the Simulink implementation of GGNN(A,B,D) dynamics (2).

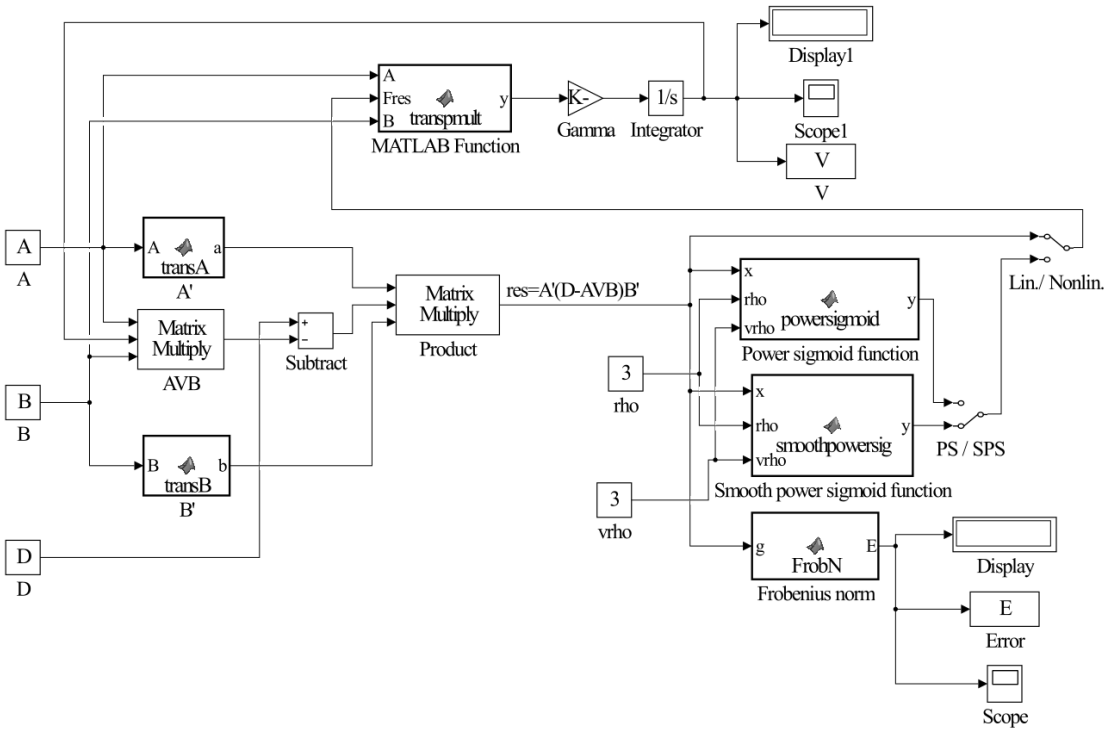


Figure 1. Simulink implementation of GGNN dynamics

https://doi.org/10.15405/epct.23021.31
Corresponding Author: Predrag S. Stanimirović
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2672-8834

6. Findings

In this section we perform numerical examples to examine the efficiency of the proposed GGNN model shown in Figure 1.

The subsequent activation functions $f(\cdot)$ are used in numerical experiments:

- Linear activation function $f_{lin}(x) = x$
- The Power-sigmoid activation function

$$f_{ps}(x,\rho,\varrho) = \begin{cases} x^{\rho}, & |x| \geq 1 \\ \dfrac{1+e^{-\varrho}}{1-e^{-\varrho}} \cdot \dfrac{1+e^{-\varrho x}}{1-e^{-\varrho x}}, & |x| < 1 \end{cases}$$

- The Smooth power-sigmoid activation function

$$f_{sps}(x,\rho,\varrho) = \frac{1}{2}x^{\rho} + \frac{1+e^{-\varrho}}{1-e^{-\varrho}} \cdot \frac{1+e^{-\varrho x}}{1-e^{-\varrho x}}$$

In power-sigmoid activation function and smooth power-sigmoid activation function $\varrho > 2, \rho \geq 3$ is odd integer. We will assume $\varrho = \rho = 3$ for all examples.

The Matlab command "*A* = rand(*m*,*k*)*rand(*k*,*n*)" is used to generate a random *m*×*n* matrix *A* of rank *r*.

Table 1 shows experimental results for square regular and non-regular random matrices of dimensions *n*×*n*. Table 2 shows experimental results obtained on regular and singular matrices of dimensions *m*×*n*. Here, NRT means that no result was obtained in a reasonable time. Experiments were conducted on computer with processor Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz, 8 GB of RAM and Windows 10 OS. MatLab Version: R2021a.

Table 1. Experiment results for squared matrices

n	Rank A	Rank B	Time	GNN	GGNN	γ	GNN CPU Time	GGNN CPU Time
50	2	2	10^{-6}	19.38	19.38	100	20.92	14.45
50	2	50	10^{-6}	17.29	17.13	100	9.05	15.17
50	50	50	10^{-6}	14.75	14.23	100	8.76	15.30
50	10	10	10^{-6}	15.77	15.59	100	7.58	16.95
50	10	50	10^{-6}	15.44	15.11	100	7.08	14.79
50	50	50	10	2.30	4.022	100	29.61	28.10
50	2	2	10	NRT	19.30	100	NRT	20.38
10	10	10	10^{-2}	2.77	2.59	1	8.82	1.43
10	2	10	10^{-2}	3.36	3.34	1	16.22	1.43
10	2	2	10^{-2}	3.60	3.54	1	9.01	1.38
10	5	10	10^{-2}	3.09	2.96	1	9.78	1.34
10	5	5	10^{-2}	3.22	3.13	1	9.92	1.47

Table 2. Experiment results for non-squared matrices

m	n	Rank A	Rank B	Time	GNN	GGNN	γ	GNN CPU Time	GGNN CPU Time

https://doi.org/10.15405/epct.23021.31
Corresponding Author: Predrag S. Stanimirović
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2672-8834

25	20	20	20	10^{-6}	26.45	12,85	100	9.25	3.42
25	20	2	2	10^{-6}	105.80	59.11	100	1.17	1.55
25	20	20	2	10^{-6}	53.10	48.76	100	1.37	2.08
25	20	10	2	10^{-6}	72.35	43.07	100	1.51	1.00
25	20	10	10	10^{-6}	33.65	28.49	100	1.58	2.16
25	20	20	10	10^{-6}	27.58	17.22	100	1.14	2.07
10	8	8	8	10^{-3}	6.19	4.26	1	0.55	1.12
10	8	2	2	10^{-3}	11.87	10.22	1	0.52	0.91
10	8	8	2	10^{-3}	6.63	6.18	1	0.70	0.90
10	8	5	2	10^{-3}	7.45	7.39	1	1.44	0.91
10	8	5	5	10^{-3}	7.37	5.88	1	0.59	0.86
10	8	8	5	10^{-3}	6.33	5.38	1	0.60	0.89

Figure 2 illustrate trajectories of residual errors $||D - AV(t)B||$ for different activation functions. The graphs included in this figure show faster convergence of nonlinear GGNN models with respect to the linear GGNN.

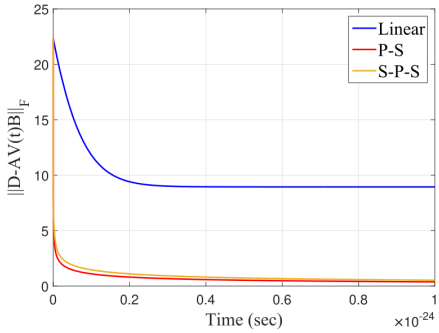
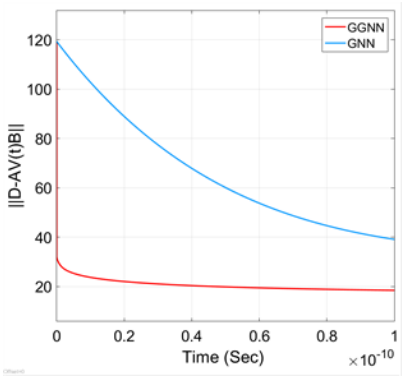
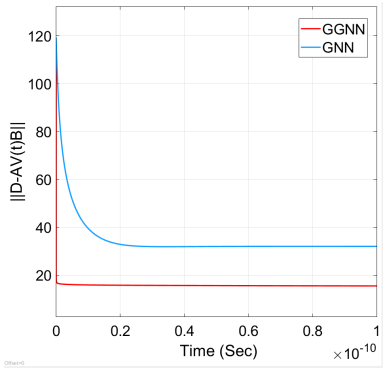


Figure 2. Trajectory of the error norm for different activation functions of GGNN

Figure 3 demonstrates a comparison of convergence rates of GNN and GGNN.

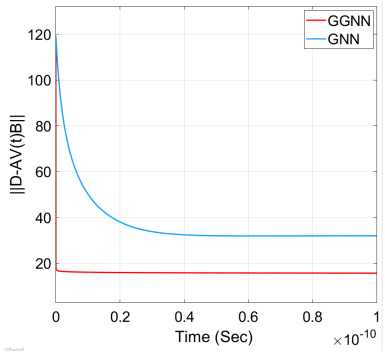


(a) Linear Activation Function



(b) Power-Sigmoid Activation Function

https://doi.org/10.15405/epct.23021.31
Corresponding Author: Predrag S. Stanimirović
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2672-8834



(c) Smooth Power-Sigmoid Activation Function

Figure 3. Frobenius norm of the error matrix $D - AV(t)B$ of GGNN against GNN.

Figure 3 clearly shows a faster convergence of the GGNN model again the GNN dynamics.

7. Conclusion

In this paper, we proposed a new method for solving the equation $AXB = D$ using the replacement of the error function and introducing a new recurrent model of GGNN. The experimental results showed that the proposed model GGNN faster converges than the GNN model without losing quality for various dimensions and ranks. Further, a non-linear activation function speeds up the convergence compared to the linear activation function for all studied cases.

Other important achievement is the fact that proposed GGNN solved all tested equations even when GNN was not able to finish computations in reasonable time.

Acknowledgments

Predrag Stanimirović is supported by the Science Fund of the Republic of Serbia, (No. 7750185, Quantitative Automata Models: Fundamental Problems and Applications - QUAM).

This work was supported by the Ministry of Science and Higher Education of the Russian Federation (Grant No. 075-15-2022-1121).

References

Ding, F., & Chen, T. (2005). Gradient based iterative algorithms for solving a class of matrix equations. *IEEE Transactions on Automatic Control*, *50*(8), 1216-1221. https://doi.org/10.1109/TAC.2005.852558

Stanimirović, P. S., Ćirić, M., Stojanović, I., & Gerontitis, D. (2017). Conditions for Existence, Representations, and Computation of Matrix Generalized Inverses. *Complexity,* 1-27. https://doi.org/10.1155/2017/6429725

Stanimirović, P. S., & Petković, M. D. (2018). Gradient neural dynamics for solving matrix equations and their applications. *Neurocomputing*, *306*, 200-212. https://doi.org/10.1016/j.neucom.2018.03.058

Stanimirović, P. S., Petković, M. D., & Gerontitis, D. (2018). Gradient neural network with nonlinear activation for computing inner inverses and the Drazin inverse. *Neural Processing Letters*, *48*(1), 109-133. https://doi.org/10.1007/s11063-017-9705-4

Stanimirović, P. S., Petković, M. D., & Mosić, D. (2022). Exact solutions and convergence of gradient based dynamical systems for computing outer inverses. *Applied Mathematics and Computation*, *412*, 126588. https://doi.org/10.1016/j.amc.2021.126588

Stanimirović, P. S., Wei, Y., Kolundžija, D., Sendra, J. R., & Sendra, J. (2019). An application of computer algebra and dynamical systems. *Proceedings of International Conference on Algebraic Informatics,* 225-236. https://doi.org/10.1007/978-3-030-21363-3_19

Urquhart, N. S. (1968). Computation of generalized inverse matrices which satisfy specified conditions. *SIAM Review*, *10*(2), 216-218. https://doi.org/10.1137/1010035

Wang, G., Wei, Y., & Qiao, S. (2018). *Generalized Inverses: Theory and Computations, Developments in Mathematics 53.* Springer. Science Press. https://doi.org/10.1007/978-981-13-0146-9

Wang, J. (1992). Electronic realisation of recurrent neural network for solving simultaneous linear equations. *Electronics Letters*, *28*, 493-495. https://doi.org/10.1049/el:19920311

Wang, J. (1993). A recurrent neural network for real-time matrix inversion. *Applied Mathematics and Computation*, *55*(1), 89-100. https://doi.org/10.1016/0096-3003(93)90007-2

Wang, J. (1997). Recurrent neural networks for computing pseudoinverses of rank-deficient matrices. *SIAM Journal on Scientific Computing*, *18*(5), 1479-1493. https://doi.org/10.1137/s1064827594267161

Wang, J., & Li, H. (1994). Solving simultaneous linear equations using recurrent neural networks. *Information Sciences*, *76*(3-4), 255-277. https://doi.org/10.1016/0020-0255(94)90012-4

Wei, Y. (2000). Recurrent neural networks for computing weighted Moore–Penrose inverse. *Applied Mathematics and Computation*, *116*(3), 279-287. https://doi.org/10.1016/s0096-3003(99)00147-2

Zhang, Y., & Chen, K. (2008). Comparison on Zhang neural network and gradient neural network for time-varying linear matrix equation AXB=C solving. *Proceedings of 2008 IEEE International Conference on Industrial Technology*, 1-6. https://doi.org/10.1109/ICIT.2008.4608579

Zhang, Y., Chen, K., & Tan, H. Z. (2009). Performance analysis of gradient neural network exploited for online time-varying matrix inversion. *IEEE Transactions on Automatic Control*, *54*(8), 1940-1945. https://doi.org/10.1109/TAC.2009.2023779