Convergence of Newton’s Method in solving Quadratic Matrix Equation with Artificial Nerual Network

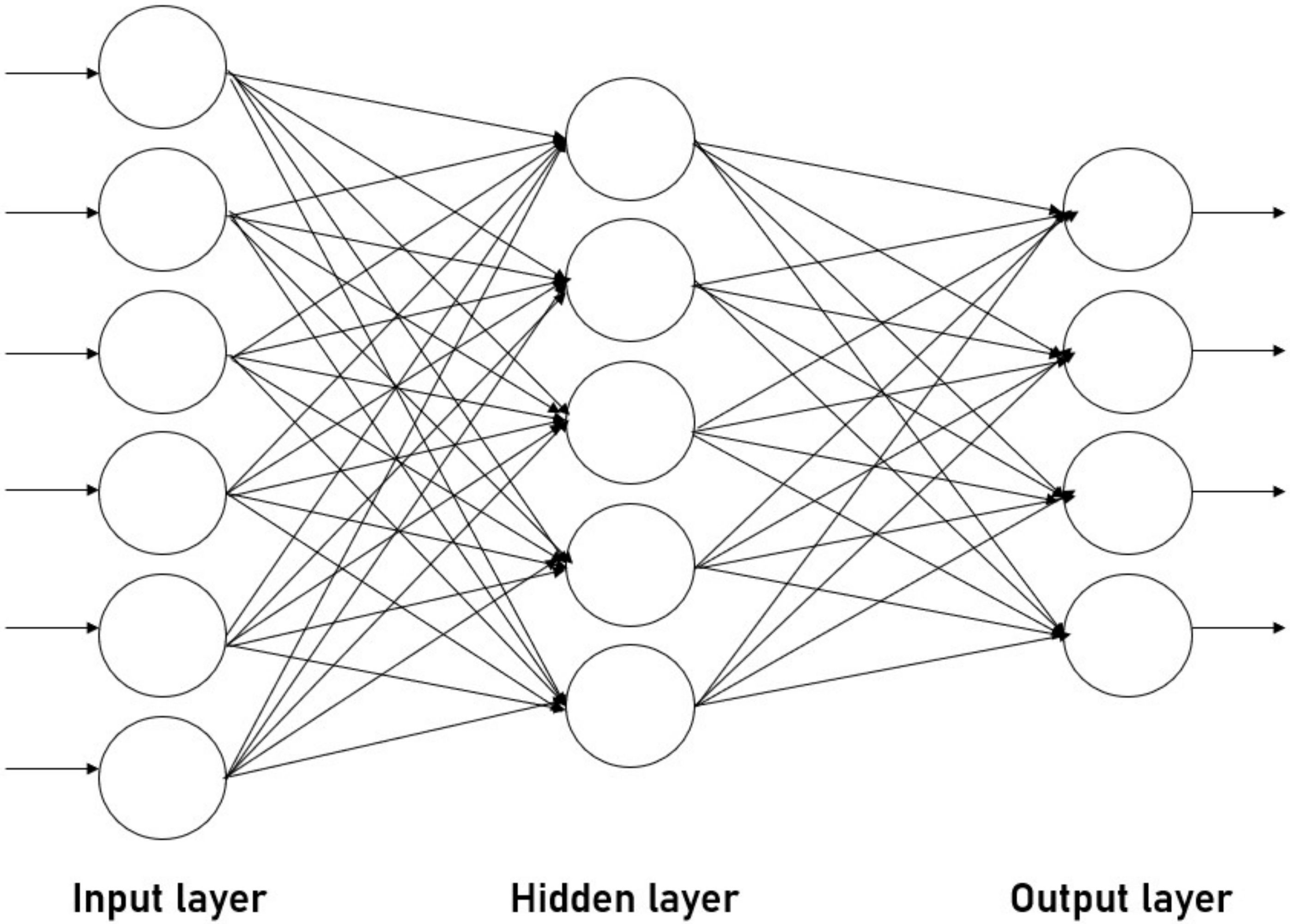
Garam Kim

Pusan National University

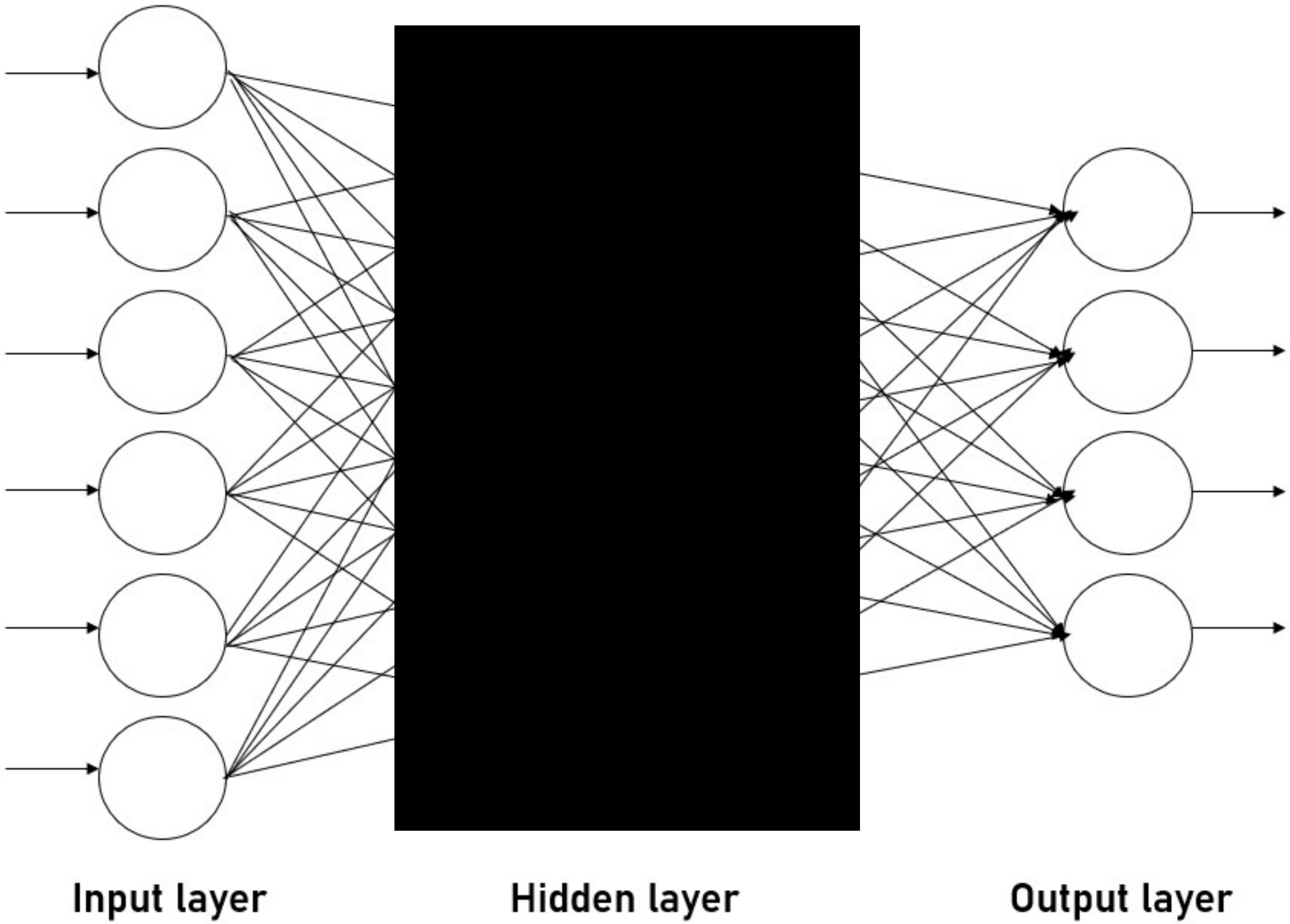
May 21, 2020

Contents

Introduction



Introduction



Introduction

The main purpose of the paper is to solve the quadratic matrix equation (QME),

$$Q(X) \equiv X^2 - EX - F = 0 \tag{1.1}$$

where $E \in R^{n\times n}$ is diagonal matrix and $\mathrm{F} \in R^{n\times n}$ is nonsingular M-matrix

Introduction

The main purpose of the paper is to solve the quadratic matrix equation (QME),

$$Q(X) \equiv X^2 - EX - F = 0 \qquad (1.1)$$

where $E \in R^{n\times n}$ is diagonal matrix and $\mathrm{F} \in R^{n\times n}$ is nonsingular M-matrix

Motivation [9]

QME (1.1) is motivated by noisy Wiener-Hopf problem for Markov Chain,

$$\frac{1}{2}\varepsilon^2 Z^2 - VZ + Q = 0 \qquad (1.2)$$

$$\frac{1}{2}\varepsilon^2 Z^2 + VZ + Q = 0 \qquad (1.3)$$

where V is a diagonal matrix, Q is Q-matrix and for given $\varepsilon > 0$ is the level of noise from a Brownian motion independent of Markov Chain.

Numerical Method for Finding Solution

Theorem (Existence and Uniqueness of the M solution [1])

If F *is nonsingular M-matrix* and E is *diagonal*, then (1.1) has exactly *one M-matrix as its solution* and the M-matrix is nonsingular.

One of the numerical methods to solve QME (1.1) is to apply Newton's method.

Remark (Newton's iteration)

$$\begin{cases} D_{X_i}(H_i) = -Q(X_i), \\ X_{i+1} = X_i + H_i \end{cases} \quad i = 0, 1, 2, \cdots$$

where D_{X_i} is the Fréchet derivative of Q at X_i in the direction H_i.

Numerical Method for Finding Solution

By letting $Y = \alpha I - X$,
We can rewrite Newton's iteration for (1.1) as

$$(\alpha I - E - Y_i)Y_{i+1} + Y_{i+1}(\alpha I - Y_i) = \alpha^2 I - \alpha E - F - Y_i^2, \qquad i = 0,1,\cdots \qquad (2.1)$$

Theorem (Convergence of Newton's method [1])

For the newton's iteration (2.1) with $Y_0 = 0$, the sequence $\{Y_i\}$ is well defined, $Y_0 \leq Y_1 \leq \cdots$, and $\lim Y_i = S_\alpha$, where $X = \alpha I - S_\alpha$.

Numerical Method for Finding Solution

Remark (Newton's Iteration for (1.1))

$$\begin{cases} H_i X_i + (X_i - E)H_i = -Q(X_i), \\ X_{i+1} = X_i + H_i \end{cases} \quad i = 0, 1, 2, \cdots \qquad (2.2)$$

The algorithm of Pure Newton's method (1.1) is as follows.

Algorithm 1 Newton's Iteration

Given $X_0,\ \varepsilon\ \ and\ i = 0$

While $\delta <\ \varepsilon$ **do**

Solve H_i in the equation (2.1)

$D_{X_i}(H_i) = -Q(X_i)$

$X_{i+1} \leftarrow X_i + H_i$

$i \leftarrow i + 1$

Calculate δ

end

$X \leftarrow X_i$

Numerical Method for Finding Solution

Remark (Newton's Iteration for (1.1))

$$\begin{cases} H_i X_i + (X_i - E)H_i = -Q(X_i), \\ X_{i+1} = X_i + H_i \end{cases} \quad i = 0, 1, 2, \cdots \tag{2.2}$$

The algorithm of Pure Newton's method (1.1) is as follows.

Algorithm 1 Newton's Iteration

Given $\boldsymbol{X_0}$, ε *and* $i = 0$

While $\delta < \varepsilon$ **do**

Solve H_i in the equation (2.1)

$D_{X_i}(H_i) = -Q(X_i)$

$X_{i+1} \leftarrow X_i + H_i$

$i \leftarrow i + 1$

Calculate δ

end

$X \leftarrow X_i$

Numerical Method for Finding Solution

Example. $f(x) = x^4 + xy + (1+y)^2$

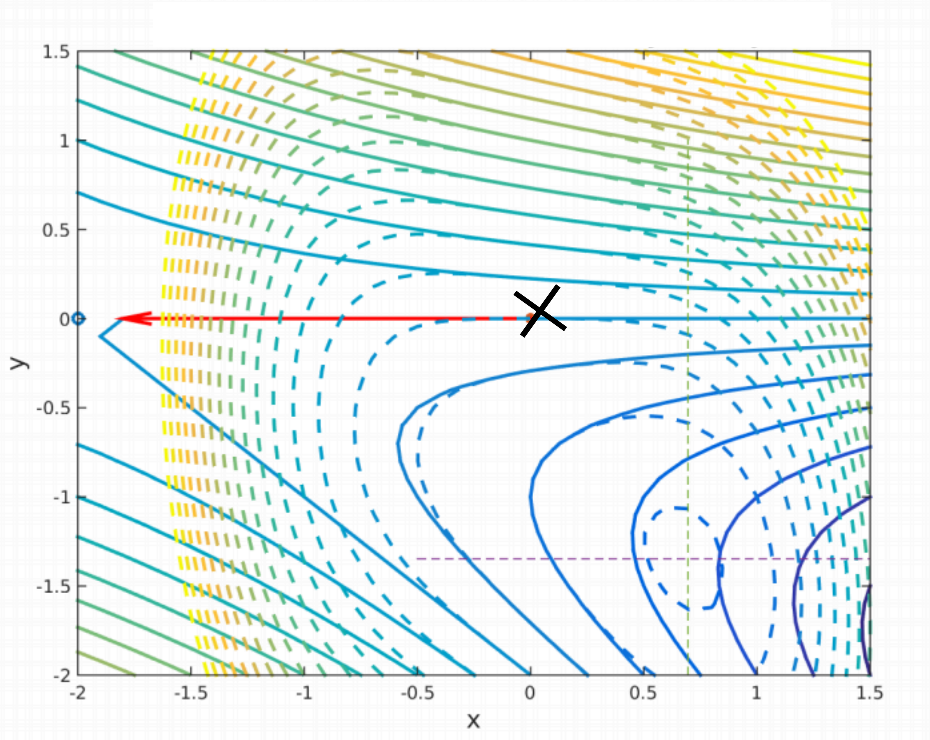


Figure. Failure of Convergence

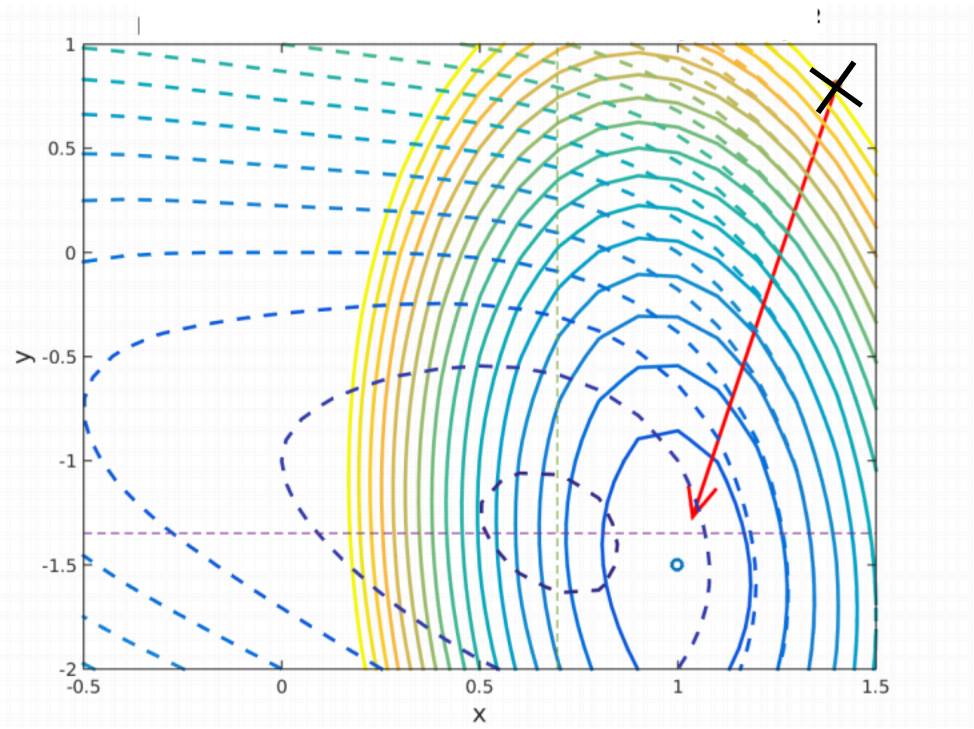
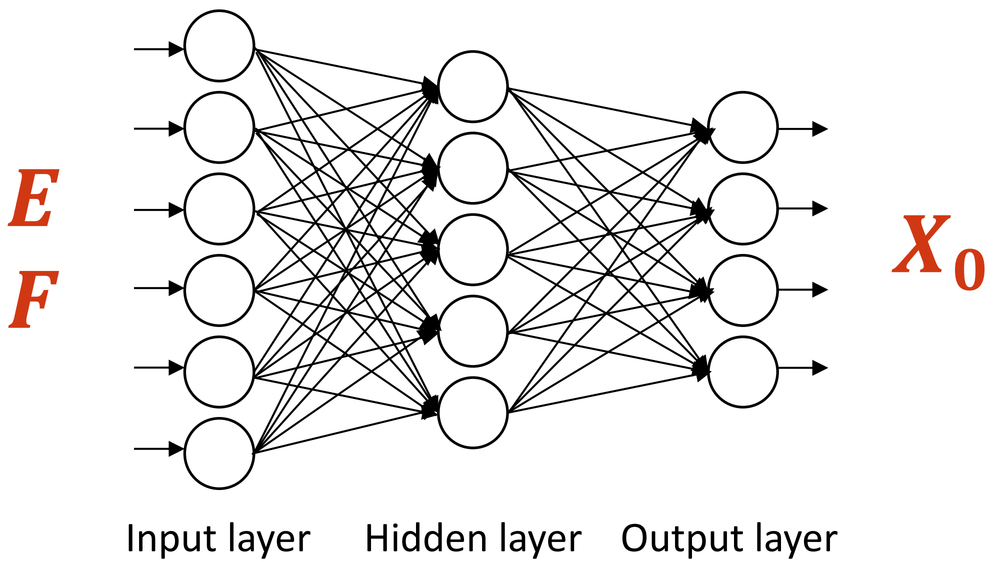


Figure. Success of Convergence

Artificial Neural Network as a Solver

Example. $f(x) = x^4 + xy + (1 + y)^2$



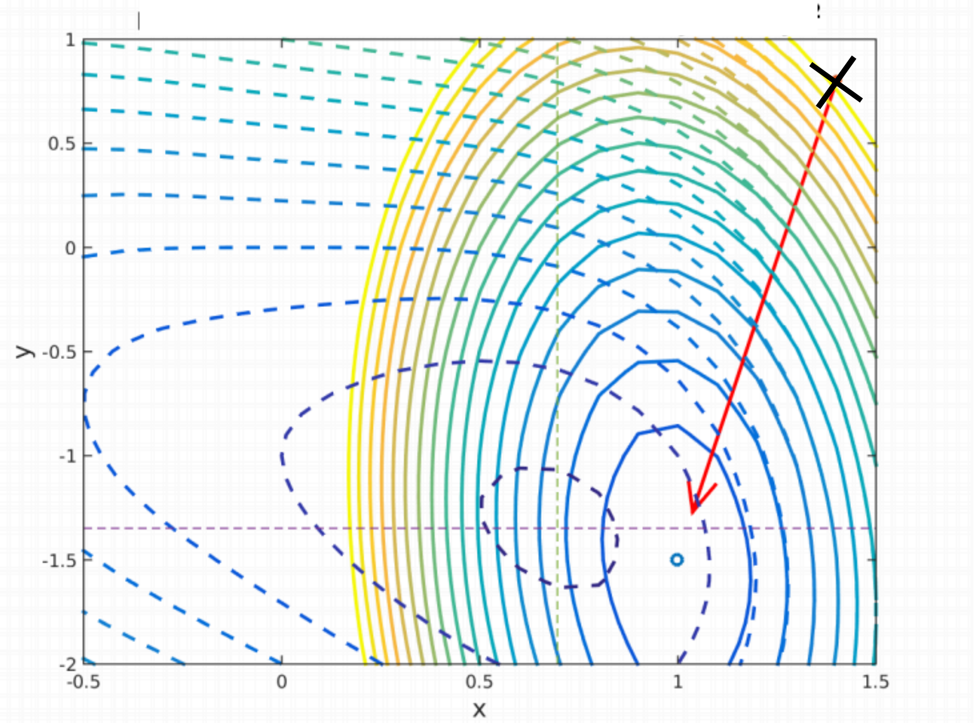


Figure. Success of Convergence

Artificial Neural Network as a Solver

Notation

- Let $L \in \mathbb{N}$ be the number of *layers* except input layer of the network
- Let $N_l \in \mathbb{N}$ be the *dimension* of the output of l-th layer for $l = 1, \cdots, L$
- Let $\sigma : \mathbb{R} \to \mathbb{R}$ be the *activation function* (acts pointwise)

Definition (Standard neural network [10])

Let $d, L \in \mathbb{N}$. A *neural network* Φ *with input dimension d and L+1 layers* is a sequence of matrix-vector tuples

$$\Phi = \big((W_1, b_1), (W_2, b_2), \cdots, (W_L, b_L)\big),$$

where $N_0 = d$ and $N_1, N_2, \cdots, N_L \in \mathbb{N}$, and where each W_l is an $N_l \times N_{l-1}$ matrix and $b_l \in \mathbb{R}^{N_l}$ for $l = 1, \cdots, L$. Then we call Φ a *standard neural network*.

Artificial Neural Network as a Solver

Definition (Standard neural network [10])

Then we define the associated realization of Φ with activation function $\sigma : \mathbb{R} \to \mathbb{R}$ as the map $R_\sigma(\Phi) : \mathbb{R}^d \to \mathbb{R}^{N_L}$ such that

$$R_\sigma(\Phi)(x) = x_L ,$$

where x_L result from scheme given by

$$\begin{aligned} x_0 &:= x, \\ x_l &:= \sigma(W_l\, x_{l-1} + b_l), \text{ for } l = 1, \cdots, L-1, \\ x_L &:= W_L\, x_{L-1} + b_L \end{aligned}$$

Artificial Neural Network as a Solver

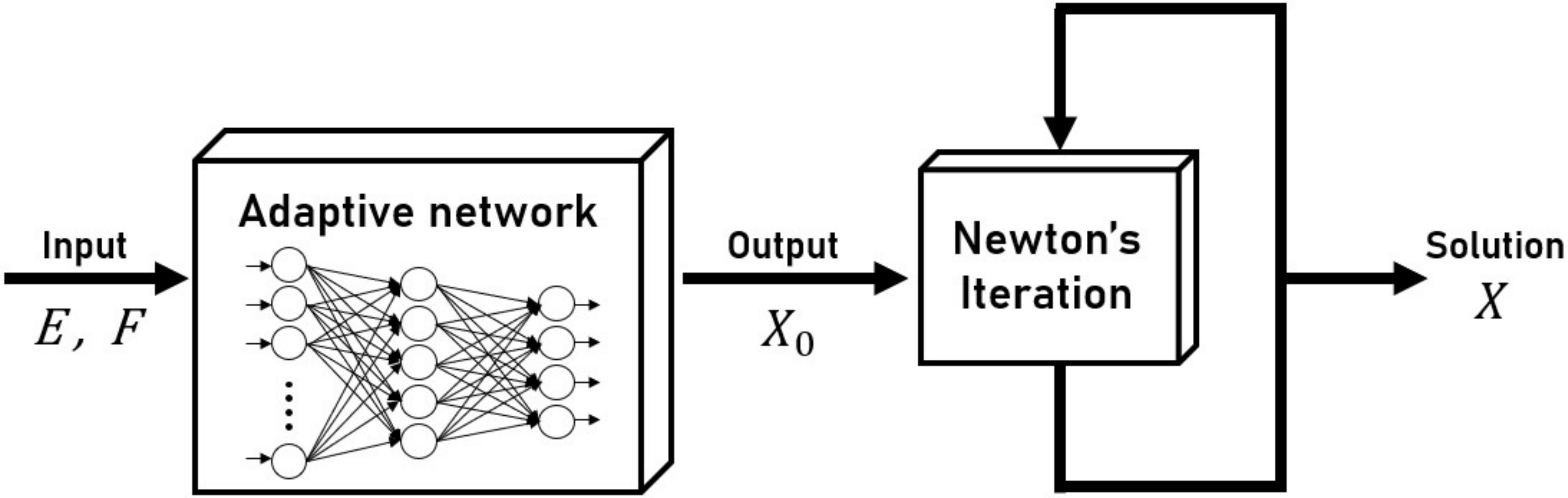


Figure. The diagram of Newton's iteration with ANN

Artificial Neural Network as a Solver

Algorithm 2. Newton's iteration with ANN

Input E, F

Output X

Given $R_\sigma(\Phi), \varepsilon\ and\ i = 0$

$input \leftarrow E, F$

$X_0 \leftarrow R_\sigma(\Phi)(input)$

While $\delta <\ \varepsilon$ **do**

Solve H_i in the equation (2.1)

$D_{X_i}(H_i) = -Q(X_i)$

$X_{i+1}\ \leftarrow X_i + H_i$

$i \leftarrow i + 1$

Calculate δ

end

$X \leftarrow X_i$

Numerical Experiment

- We adopted $L = 2$ and $d = n^2 + n$, where $Q(X) \in \mathbb{R}^{n \times n}$.
- The activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is sigmoid. i.e. $\sigma(x) = (1 + \exp(-x))^{-1}$

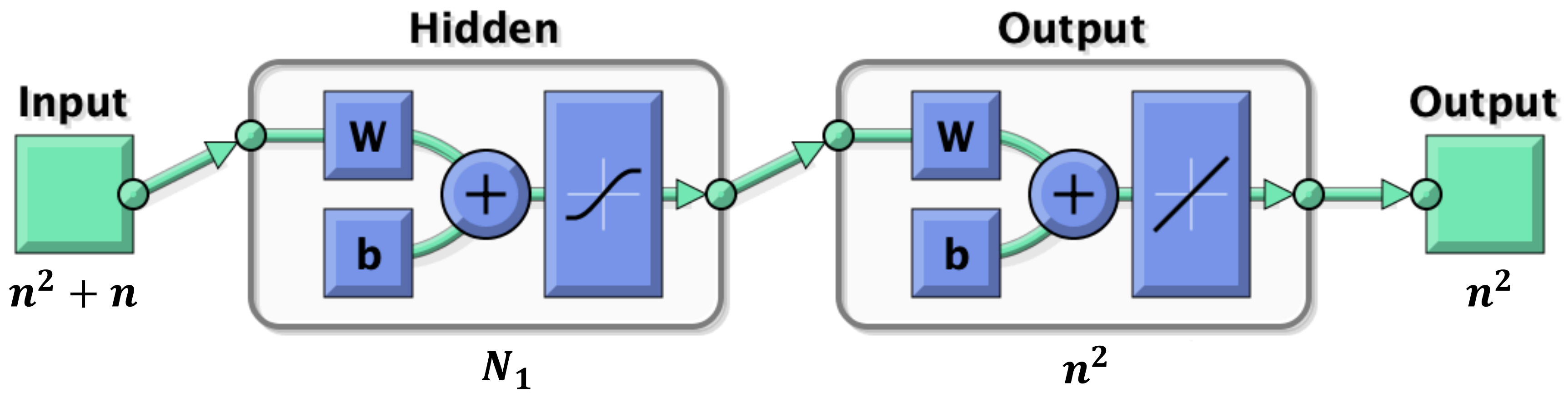


Figure. The diagram of shallow feedforward network.

- We set *tolerance* to $\varepsilon = 10^{-14}$ for break each iteration of numerical experiments.
- All iterations are terminated when the *absolute error* δ is less than the tolerance,

  where $\delta = \|Q(X_i)\|_F$.

Numerical Experiment

Example

Recall the equation (1.1),

$$Q(X) \equiv X^2 - EX - F = 0.$$

Let $N_1 = 5$, and $n = 2, 3, 5, 7, 10$.
In Matlab code, the coeffiecnt E and F are defined as

$$E = diag\big(rand(n, 1)\big),$$
$$F = make\_F(n).$$

For Algorithm 1, random starting matrix X_0 is given.

$$X_0 = rand(n)$$

Algorithm 3. make_F

Given n
While 1 **do**
  $B_0 \leftarrow -rand(n)$
  For $k \leftarrow 1$ **to** n **do**
    $B_0(k,k) \leftarrow rand \times 5$
  end
  If $\mathrm{eig}(B_0) > 0$
    break
  end
end
$F \leftarrow B_0$

Numerical Experiment

n	Algorithm 1 Pure Newton’s iteration	Algorithm 2 Newton’s iteration with ANN
2	2	0
3	12	0
5	20	0
7	39	0
10	100	0

Table. The number of equations does not converge among 100 equations.

Conclusion

In this study, we propose *Newton's iteration with ANN* which promise the convergence, even for higher dimension of matrix equation, so that the incompletion of convergence in Newton's method is complemented.

Future Works

1. Since there is no specific theroem about starting matrix, it could be suggested to guarantee the convergence of newton's iteration.
2. The equation could be extended to every quadratic matrix equation or higher order.
3. With adjustment in the structure, ANN could be a single solver.

Reference

[1] C. Guo, On a quadratic matrix equation associated with an M-matrix, IMA Journal of Numerical Analysis, vol. 23, no. 1 (2003):11-27.

[2] Kim, Hyun-Min. “CONVERGENCE OF NEWTON’S METHOD FOR SOLVING A CLASS OF QUADRATIC MATRIX EQUATIONS.” Honam Journal 30, no. 2 (June 25, 2008): 399–409. doi:10.5831/HMJ.2008.30.2.399.

[3] George J. Davis. 1983. Algorithm 598: an algorithm to compute solvent of the matrix equation AX2 + BX + C = 0. ACM Trans. Math. Softw. 9, 2 (June 1983), 246–254. DOI:https://doi.org/10.1145/357456.357463

[4] J.E. Dennis, Jr. and R. B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Englewood Cliffs, NJ, (1983).

[5] N.J. Higham and H.-M. Kim, Solving a quadratic matrix equation by Newton’s method with exact line searches, SIAM J. Matrix. Anal. Appl., 23:303-316, (2006).

[6] S Agatonovic-Kustrin,R Beresford, Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research, Journal of Pharmaceutical and Biomedical Analysis, Vol. 22, Issue 5, June 2000, Pages 717-727

[7] Okut, H. (2016). Bayesian Regularized Neural Networks for Small n Big p Data.

[8] Roger A Horn, Matrix Analysis: Second Edition 2nd Edition.

[9] Rogers, L. C. G, ”Fluid Models in Queueing Theory and Wiener-Hopf Factorization of Markov Chains.” The Annals of Applied Probability 4, no. 2 (1994): 390-413.

[10] Gühring, Ingo, Gitta Kutyniok and Philipp Petersen, “Complexity bounds for approximations with deep ReLU neural networks in Sobolev norms.” (2019).

[11] Rumelhart, D., Hinton, G. and Williams, R. Learning representations by back-propagating errors. Nature 323, 533–536 (1986). https://doi.org/10.1038/323533a0

[12] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, Backpropagation Applied to Handwritten Zip Code Recognition, Neural Computation Vol1 (1989) p.541-551