Scientific Research Publishing

Advances in Pure Mathematics, 2018, 8, 485-493
http://www.scirp.org/journal/apm
ISSN Online: 2160-0384
ISSN Print: 2160-0368

Least Squares Method from the View Point of Deep Learning

Kazuyuki Fujii[1,2]

[1]International College of Arts and Sciences, Yokohama City University, Yokohama, Japan,
[2]Department of Mathematical Sciences, Shibaura Institute of Technology, Saitama, Japan
Email: fujii@yokohama-cu.ac.jp

How to cite this paper: Fujii, K. (2018) Least Squares Method from the View Point of Deep Learning. *Advances in Pure Mathematics*, **8**, 485-493.
https://doi.org/10.4236/apm.2018.85027

Received: April 10, 2018
Accepted: May 26, 2018
Published: May 29, 2018

Abstract

The least squares method is one of the most fundamental methods in Statistics to estimate correlations among various data. On the other hand, Deep Learning is the heart of Artificial Intelligence and it is a learning method based on the least squares. In this paper we reconsider the least squares method from the view point of Deep Learning and we carry out the computation thoroughly for the gradient descent sequence in a very simple setting. Depending on the values of the learning rate, an essential parameter of Deep Learning, the least squares methods of Statistics and Deep Learning reveal an interesting difference.

Keywords

Least Squares Method, Statistics, Deep Learning, Learning Rate, Linear Algebra

1. Introduction

The least squares method in Statistics plays an important role in almost all disciplines, from Natural Science to Social Science. When we want to find properties, tendencies or correlations hidden in huge and complicated data we usually employ the method. See for example [1].

On the other hand, Deep Learning is the heart of Artificial Intelligence and will become a most important field in Data Science in the near future. As to Deep Learning see for example [2] [3] [4] [5] [6].

Deep Learning may be stated as a successive learning method based on the least squares method. Therefore, to reconsider it from the view point of Deep Learning is very natural and we carry out the calculation thoroughly of the successive approximation called gradient descent sequence.

When the learning rate changes a difference in method between Statistics and Deep Learning gives different results.

Theorem I and II in the text are our main results and a related problem (exercise) is presented for readers. Our results may give a new insight to both Statistics and Data Science.

First of all let us explain the least squares method for readers in a very simple setting. For n pieces of two dimensional real data

$$\{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$$

we assume that their scatter plot is like Figure 1

Then a model function is linear

$$f(x) = ax + b. \tag{1}$$

For this function the error (or loss) function is defined by

$$E(a,b) = \frac{1}{2}\sum_{i=1}^{n}\{y_i - (ax_i + b)\}^2. \tag{2}$$

The aim of least squares method is to minimize the error function (2) with respect to $\{a,b\}$. A little calculation gives

$$E(a,b) = \frac{1}{2}\left\{\sum_{i=1}^{n} x_i^2 a^2 + 2\sum_{i=1}^{n} x_i ab + nb^2 - 2\sum_{i=1}^{n} x_i y_i a - 2\sum_{i=1}^{n} y_i b + \sum_{i=1}^{n} y_i^2\right\}.$$

Then the equations for the stationality

$$\frac{\partial E}{\partial a} = \frac{\partial E}{\partial b} = 0 \tag{3}$$

give a linear equation for a and b

$$\begin{pmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & n \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{pmatrix} \tag{4}$$

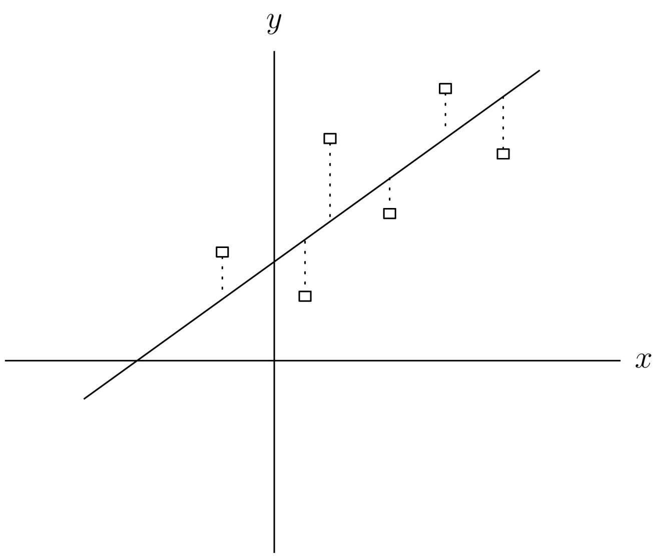

Figure 1. Scatter plot 1.

and its solution is given by

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & n \end{pmatrix}^{-1} \begin{pmatrix} \sum_{i=1}^n x_i y_i \\ \sum_{i=1}^n y_i \end{pmatrix}. \tag{5}$$

Explicitly, we have

$$\begin{aligned} a &= \frac{n\left(\sum_{i=1}^n x_i y_i\right) - \left(\sum_{i=1}^n x_i\right)\left(\sum_{i=1}^n y_i\right)}{n\left(\sum_{i=1}^n x_i^2\right) - \left(\sum_{i=1}^n x_i\right)^2}, \\ b &= \frac{-\left(\sum_{i=1}^n x_i\right)\left(\sum_{i=1}^n x_i y_i\right) + \left(\sum_{i=1}^n x_i^2\right)\left(\sum_{i=1}^n y_i\right)}{n\left(\sum_{i=1}^n x_i^2\right) - \left(\sum_{i=1}^n x_i\right)^2}. \end{aligned} \tag{6}$$

To check that a and b give the minimum of (2) is a good exercise.

Note We have an inequality

$$n\left(x_1^2 + x_2^2 + \cdots + x_n^2\right) \geq \left(x_1 + x_2 + \cdots + x_n\right)^2 \tag{7}$$

and the equal sign holds if and only if

$$x_1 = x_2 = \cdots = x_n.$$

Since $\{x_1, x_2, \cdots, x_n\}$ are data we may assume that $x_i \neq x_j$ for some $i \neq j$. Therefore

$$n\left(x_1^2 + x_2^2 + \cdots + x_n^2\right) - \left(x_1 + x_2 + \cdots + x_n\right)^2 > 0$$

gives

$$\begin{vmatrix} \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & n \end{vmatrix} > 0. \tag{8}$$

2. Least Squares Method from Deep Learning

In this section we reconsider the least squares method in Section 1 from the view point of Deep Learning.

First we arrange the data in Section 1 like

$$\text{Input data}: \{(x_1, 1), (x_2, 1), \cdots, (x_n, 1)\}$$

$$\text{Teacher signal}: \{y_1, y_2, \cdots, y_n\}$$

and consider a simple neuron model in [7] (see **Figure 2**)

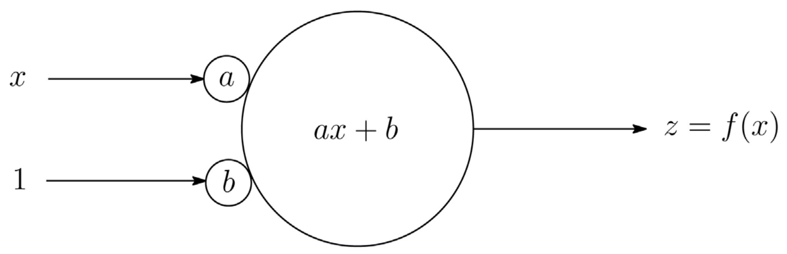

Figure 2. Simple neuron model 1.

Here we use the linear function (1) instead of the sigmoid function $z = \sigma(x)$. In this case the square error function becomes

$$L(a,b) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - f(x_i)\right)^2 = \frac{1}{2}\sum_{i=1}^{n}\left\{y_i - (ax_i + b)\right\}^2 .$$

We usally use $L(a,b)$ instead of $E(a,b)$ in (2).

Our aim is also to determine the parameters $\{a,b\}$ in order to minimize $L(a,b)$. However, the procedure is different from the least squares method in Section 1. This is an important and interesting point.

For later use let us perform a little calculation

$$\begin{aligned}\frac{\partial L}{\partial a} &= -\sum_{i=1}^{n}\left(y_i - f(x_i)\right)\frac{\partial f}{\partial a} = -\sum_{i=1}^{n}\left(y_i - f(x_i)\right)x_i,\\ \frac{\partial L}{\partial b} &= -\sum_{i=1}^{n}\left(y_i - f(x_i)\right)\frac{\partial f}{\partial b} = -\sum_{i=1}^{n}\left(y_i - f(x_i)\right).\end{aligned} \tag{9}$$

We determine the parameters $\{a,b\}$ successively by the gradient descent method, see for example [8]. For $t = 0,1,\cdots,n,\cdots$

$$\begin{pmatrix} a(0) \\ b(0) \end{pmatrix} \to \cdots \to \begin{pmatrix} a(t) \\ b(t) \end{pmatrix} \to \begin{pmatrix} a(t+1) \\ b(t+1) \end{pmatrix} \to \cdots$$

and

$$\begin{aligned} a(t+1) &= a(t) - \epsilon\frac{\partial L}{\partial a(t)},\\ b(t+1) &= b(t) - \epsilon\frac{\partial L}{\partial b(t)}, \end{aligned} \tag{10}$$

where

$$L = L\left(a(t),b(t)\right) = \frac{1}{2}\sum_{i=1}^{n}\left\{y_i - \left(a(t)x_i + b(t)\right)\right\}^2$$

and $\epsilon\,(0 < \epsilon < 1)$ is small enough. The initial value $\left(a(0),b(0)\right)^{\mathrm{T}}$ is given appropriately. As will be shown shortly in Theorem I, their explicit values are not important.

Comment The parameter ϵ is called the *learning rate* and it is very hard to choose ϵ properly as emphasized in [7]. In this paper we provide an estimation (see Theorem II).

Let us write down (10) explicitly:

$$\begin{aligned} a(t+1) &= a(t) + \epsilon\sum_{i=1}^{n}\left\{y_i - \left(a(t)x_i + b(t)\right)\right\}x_i \\ &= \left(1 - \epsilon\sum_{i=1}^{n}x_i^2\right)a(t) - \epsilon\left(\sum_{i=1}^{n}x_i\right)b(t) + \epsilon\sum_{i=1}^{n}x_i y_i \end{aligned}$$

and

$$\begin{aligned} b(t+1) &= b(t) + \epsilon\sum_{i=1}^{n}\left\{y_i - \left(a(t)x_i + b(t)\right)\right\} \\ &= -\epsilon\left(\sum_{i=1}^{n}x_i\right)a(t) + (1 - n\epsilon)b(t) + \epsilon\sum_{i=1}^{n}y_i . \end{aligned}$$

These are cast in a vector-matrix form

$$\begin{aligned}\begin{pmatrix} a(t+1) \\ b(t+1) \end{pmatrix} &= \begin{pmatrix} \left(1-\epsilon\sum_{i=1}^{n} x_i^2\right) a(t) - \epsilon\left(\sum_{i=1}^{n} x_i\right) b(t) + \epsilon\sum_{i=1}^{n} x_i y_i \\ -\epsilon\left(\sum_{i=1}^{n} x_i\right) a(t) + (1-n\epsilon) b(t) + \epsilon\sum_{i=1}^{n} y_i \end{pmatrix} \\ &= \begin{pmatrix} 1-\epsilon\sum_{i=1}^{n} x_i^2 & -\epsilon\sum_{i=1}^{n} x_i \\ -\epsilon\sum_{i=1}^{n} x_i & 1-n\epsilon \end{pmatrix} \begin{pmatrix} a(t) \\ b(t) \end{pmatrix} + \begin{pmatrix} \epsilon\sum_{i=1}^{n} x_i y_i \\ \epsilon\sum_{i=1}^{n} y_i \end{pmatrix} \\ &= \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \epsilon \begin{pmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & n \end{pmatrix} \right\} \begin{pmatrix} a(t) \\ b(t) \end{pmatrix} + \epsilon \begin{pmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{pmatrix}. \end{aligned} \tag{11}$$

For simplicity by setting

$$\boldsymbol{x}(t) = \begin{pmatrix} a(t) \\ b(t) \end{pmatrix}, \quad A = \begin{pmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & n \end{pmatrix}, \quad \boldsymbol{f} = \begin{pmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{pmatrix}$$

we have a simple equation

$$\boldsymbol{x}(t+1) = (E - \epsilon A)\boldsymbol{x}(t) + \epsilon \boldsymbol{f} \tag{12}$$

where E is a unit matrix. Due to (8) the matrix A is invertible ($\exists A^{-1}$), that is, we exclude the trivial and uninteresting case $x_1 = x_2 = \cdots = x_n$.

The solution is easy and given by

$$\boldsymbol{x}(t) = (E-\epsilon A)^t \boldsymbol{x}(0) + \left\{ E - (E-\epsilon A)^t \right\} A^{-1} \boldsymbol{f}. \tag{13}$$

Note Let us consider a simple difference equation

$$x(n+1) = ax(n) + b \quad (a \neq 1)$$

for $n = 0,1,2,\cdots$. Then, the solution is given by

$$x(n) = a^n x(0) + \frac{a^n - 1}{a-1} b = a^n x(0) + \frac{1-a^n}{1-a} b.$$

Check this.

Comment The solution (13) gives

$$\lim_{t\to\infty} \boldsymbol{x}(t) = A^{-1}\boldsymbol{f} \tag{14}$$

if

$$\lim_{t\to\infty} (E-\epsilon A)^t = O \tag{15}$$

where O is a zero matrix. (14) is just the equation (5).

Let us evaluate (13) further. For the purpose we make some preparations from Linear Algebra [9]. For simplicity we set

$$A = \begin{pmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & n \end{pmatrix} \equiv \begin{pmatrix} \alpha & \beta \\ \beta & \delta \end{pmatrix}, \quad \boldsymbol{f} = \begin{pmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{pmatrix} \equiv \begin{pmatrix} e \\ f \end{pmatrix}$$

and want to diagonalize A.

The characteristic polynomial of A is

$$f(\lambda)=|\lambda E-A|=\begin{vmatrix}\lambda-\alpha & -\beta \\ -\beta & \lambda-\delta\end{vmatrix}=(\lambda-\alpha)(\lambda-\delta)-\beta^2$$
$$=\lambda^2-(\alpha+\delta)\lambda+\alpha\delta-\beta^2$$

and the solutions are given by

$$\lambda_\pm=\frac{\alpha+\delta\pm\sqrt{(\alpha+\delta)^2-4(\alpha\delta-\beta^2)}}{2}=\frac{\alpha+\delta\pm\sqrt{(\alpha-\delta)^2+4\beta^2}}{2}. \tag{16}$$

It is easy to see

$$\lambda_+>\lambda_->0,$$
$$\lambda_+>1 \quad (\delta=n \text{ and } n\geq 2). \tag{17}$$

We set the two eigenvectors of matrix A, corresponding to λ_+ and λ_-, in a matrix form

$$\tilde{Q}=\begin{pmatrix}\lambda_+-\delta & \beta \\ \beta & \lambda_--\alpha\end{pmatrix}.$$

It is easy to see

$$(\lambda_+-\delta)^2+\beta^2=(\lambda_--\alpha)^2+\beta^2\equiv\Lambda^2$$

from (16) and we also set

$$Q=\frac{1}{\Lambda}\begin{pmatrix}\lambda_+-\delta & \beta \\ \beta & \lambda_--\alpha\end{pmatrix}. \tag{18}$$

Then it is easy to see

$$Q^{\mathrm{T}}Q=QQ^{\mathrm{T}}=E\Rightarrow Q^{\mathrm{T}}=Q^{-1}.$$

Namely, Q is an orthogonal matrix. Then the diagonalization of A becomes

$$A=Q\begin{pmatrix}\lambda_+ & 0 \\ 0 & \lambda_-\end{pmatrix}Q^{\mathrm{T}}. \tag{19}$$

By substituting (19) into (13) and using

$$E-\epsilon A=Q\begin{pmatrix}1-\epsilon\lambda_+ & 0 \\ 0 & 1-\epsilon\lambda_-\end{pmatrix}Q^{\mathrm{T}}$$

we finally obtain

Theorem I A general solution to (12) is

$$\begin{pmatrix}a(t) \\ b(t)\end{pmatrix}=Q\begin{pmatrix}(1-\epsilon\lambda_+)^t & 0 \\ 0 & (1-\epsilon\lambda_-)^t\end{pmatrix}Q^{\mathrm{T}}\begin{pmatrix}a(0) \\ b(0)\end{pmatrix}$$
$$+Q\begin{pmatrix}\dfrac{1-(1-\epsilon\lambda_+)^t}{\lambda_+} & 0 \\ 0 & \dfrac{1-(1-\epsilon\lambda_-)^t}{\lambda_-}\end{pmatrix}Q^{\mathrm{T}}\begin{pmatrix}e \\ f\end{pmatrix}. \tag{20}$$

This is our main result.

Lastly, let us show how to choose the learning rate $\epsilon(0<\epsilon<1)$, which is a very important problem in Deep Learning. Let us remember

$$\lambda_+ > \lambda_- > 0 \text{ and } \lambda_+ > 1$$

from (17). From (15) the equations

$$\lim_{t\to\infty}(E-\epsilon A)^t = O \Leftrightarrow \lim_{t\to\infty}(1-\epsilon\lambda_+)^t = \lim_{t\to\infty}(1-\epsilon\lambda_-)^t = 0$$

determine the range of ϵ. Noting

$$|1-x| < 1 (\Leftrightarrow 0 < x < 2) \Rightarrow \lim_{n\to\infty}(1-x)^n = 0$$

we obtain

Theorem II The learning rate ϵ must satisfy an inequality

$$0 < \epsilon\lambda_+ < 2 \Leftrightarrow 0 < \epsilon < \frac{2}{\lambda_+}. \tag{21}$$

From (21) ϵ becomes very small when λ_+ is large enough. It is easy to see that the second condition $\lim_{t\to\infty}(1-\epsilon\lambda_-)^t = 0$ is automatically satisfied.

Under Theorem II we can recover (14)

$$\lim_{t\to\infty}\begin{pmatrix} a(t) \\ b(t) \end{pmatrix} = O \begin{pmatrix} \frac{1}{\lambda_+} & 0 \\ 0 & \frac{1}{\lambda_-} \end{pmatrix} O^{\mathrm{T}} \begin{pmatrix} e \\ f \end{pmatrix} = A^{-1}\boldsymbol{f}$$

by (19).

Comment For example, if we choose ϵ like

$$\frac{2}{\lambda_+} < \epsilon < 1$$

then we cannot recover (14), which shows a difference between Statistics and Deep Learning. Let us emphasize that the choice of the initial values $\{a(0), b(0)\}$ is irrelevant when the convergence condition (21) is satisfied.

As a result, how to choose ϵ properly in Deep Learning becomes a very important problem when the number of data is huge. As far as we know the result like Theorem II has not been obtained.

3. Problem

In this section we present the outline of a simple generalization of the results in Section 2. The actual calculation is left as a problem (exercise) to readers.

For n pieces of three dimensional real data

$$\{(x_1, y_1, z_1), (x_2, y_2, z_2), \cdots, (x_n, y_n, z_n)\}$$

we assume that its scatter plot is like Figure 3

Then a model function is linear

$$f(x, y) = ax + by + c \tag{22}$$

and the error (or loss) function is defined by

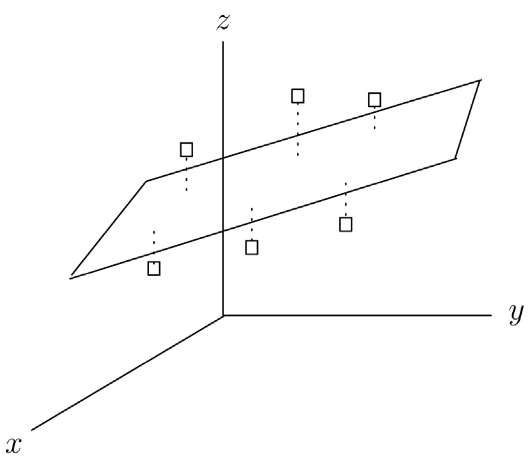

Figure 3. Scatter plot 2.

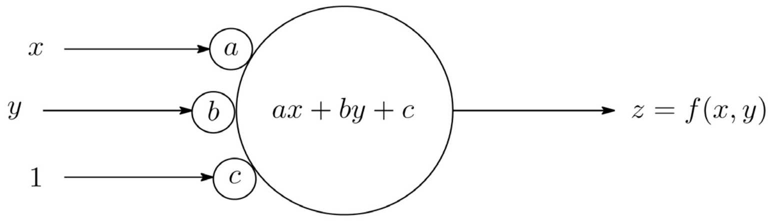

Figure 4. Simple neuron model 2.

$$E(a,b,c)=\frac{1}{2}\sum_{i=1}^{n}\left(z_i-f\left(x_i,y_i\right)\right)^2=\frac{1}{2}\sum_{i=1}^{n}\left\{z_i-\left(ax_i+by_i+c\right)\right\}^2. \tag{23}$$

The aim of least squares method is to minimize the error function (22) with respect to $\{a,b,c\}$.

As we want to treat the least squares method above from the view point of Deep Learning, we again arrange the data like

$$\text{Input data}:\left\{\left(x_1,y_1,1\right),\left(x_2,y_2,1\right),\cdots,\left(x_n,y_n,1\right)\right\}$$

$$\text{Teacher signal}:\left\{z_1,z_2,\cdots,z_n\right\}$$

and consider another simple neuron model (see Figure 4)

Then we present

Problem Carry out the corresponding calculation as given in Section 2.

4. Concluding Remarks

In this paper we discussed the least squares method from the view point of Deep Learning and carried out calculation of the gradient descent thoroughly. A difference in methods between Statistics and Deep Learning delivers different results when the learning rate ϵ is changed. The result of Theorem II is the first one as far as we know.

Deep Learning plays an essential role in Data Science and maybe in almost all fields of Science. Therefore it is desirable for undergraduates to master it as soon as possible. To master it they must study Calculus, Linear Algebra and Statistics from Mathematics. However we don't know a good and compact textbook leading to Deep Learning.

I am planning to write a comprehensive textbook in the near future [10].

Acknowledgements

We wish to thank Ryu Sasaki for useful suggestions and comments.

References

[1] Wikipedia: Least Squares. https://en.wikipedia.org/wiki/Least_squares

[2] Wikipedi: Deep Learning. https://en.wikipedia.org/wiki/Deep_learning

[3] Goodfellow, I., Bengio, Y. and Courville, A. (2016) Deep Learning. The MIT Press, Cambridge.

[4] Patterson, J. and Gibson, A. (2017) Deep Learning: A Practitioner's Approach. O'Reilly Media, Inc., Sebastopol.

[5] Okaya, T. (2015) Deep Learning (In Japanese). Kodansha Ltd., Tokyo.

[6] Amari, S. (2016) Brain·Heart·Artificial Intelligence (In Japanese). Kodansha Ltd., Blue Backs B-1968, Tokyo.

[7] Fujii, K. (2018) Mathematical Reinforcement to the Minibatch of Deep Learning. *Advances in Pure Mathematics*, **8**, 307-320. https://doi.org/10.4236/apm.2018.83016

[8] Wikipedia: Gradient Descent. https://en.wikipedia.org/wiki/Gradient_descent

[9] Kasahara, K. (2000) Linear Algebra (In Japanese). Saiensu Ltd., Tokyo.

[10] Fujii, K. Introduction to Mathematics for Understanding Deep Learning. In Preparation.