Numeric Law Discovery using Neural Networks

Kazumi Saito[1] and Ryohei Nakano[2]

[1]**NTT R&D Management Department, 3-19-2, Nishi-shinjuku, Shinjuku-ku, Tokyo 163-19 Japan**
saito@rdh.ecl.ntt.co.jp
[2]**NTT Communication Science Laboratories, 2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan**
nakano@cslab.kecl.ntt.co.jp

Abstract

This paper proposes a new connectionist approach to numeric law discovery; i.e., neural networks (law-candidates) are trained by using a newly invented second-order learning algorithm based on a quasi-Newton method, called BPQ, and the MDL criterion selects the most suitable from law-candidates. The main advantage of our method over previous work of symbolic or connectionist approach is that it can efficiently discover numeric laws whose power values are not restricted to integers. Experiments showed that the proposed method works well in discovering such laws even from data containing irrelevant variables or a small amount of noise.

1. Introduction

The discovery of a numeric law from a set of data is the central part of scientific discovery systems. Such systems, for example, can detect a relationship between the distance r to the sun and the revolution period T of five planets known to Kepler's third law $T = kr^{3/2}$ (k is a constant).

After the pioneering work of the BACON systems [5, 6], several methods [7, 3, 8, 16, 17] have been proposed. The basic search strategy employed by these methods is much the same: two variables are recursively combined into a new variable by using multiplication, division, or some predefined prototype functions. BACON and FAHRENHEIT [7] use trend detectors to combine variables and employ a heuristic form of depth-first search. ABACUS [3] creates a proportional graph and performs a modified beam search. In IDS [8], correlation analysis is applied, and a beam search is performed. The E* algorithm [16] considers only bivariate functions. Also, Sutton-Matheus' algorithm [17] performs a regression, and the correlation between the squared error and the square of the variable's value [15] is used to combine variables.

These existing methods suffer from the following problems: first, because combining two variables into a new one must be done in order, a combinatorial explosion may occur when complex laws are sought for data consisting of a large number of variables, or the desired laws will be missed when some heuristic search parameters are inappropriate. Second, when some powers appearing in a law are not restricted to integers, the law may remain unknown unless some appropriate prototype functions are prepared in advance. However, a priori information is rarely available. Third, these methods are often criticized for their lack of robustness; noise tolerance is definitely required since real observed data contain noise [8, 16].

We believe a connectionist approach has great potential to solve the above problems. In order to directly learn a generalized polynomial term whose power values are not restricted to integers, a computational unit called a product unit has been proposed [2]; instead of calculating a weighted sum of input values, this unit calculates a weighted product, where each input value is raised to a power determined by a variable weight. However, serious difficulties have been reported when using standard BP [13] to train networks containing these units [9]. Although some heuristic strategies such as multiple learning algorithms have been proposed [9], their improvements over BP have been less than remarkable. Moreover, these earlier studies dealt only with binary data and did not specifically address the problem of numeric law discovery.

In this paper, we propose a connectionist approach called RF5 for discovering numeric laws. Section 2 explains how neural networks are used to discover a class of numeric laws. Section 3 describes a second-order learning algorithm called BPQ which trains the neural networks described in Section 2. Section 4 explains a criterion for selecting the most suitable candidate out of the trained ones. Section 5 evaluates the proposed method RF5 by doing experiments using artificial data, real data, and time series data.

2. Law Discovery using Neural Nets

This section explains a connectionist problem formalization for numeric law discovery, first proposed in [2]. Let $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$ be a set of examples, where $\mathbf{x}_t$ is an n-dimensional input vector and y_t is a target value corresponding to $\mathbf{x}_t$. In this paper, a class of numeric laws expressed as

$$y_t \quad = \quad c_0 + \sum_{i=1}^{h} c_i x_{t1}^{w_{i1}} \cdots x_{tn}^{w_{in}} \tag{1}$$

is considered, where each parameter c_i or w_{ij} is an unknown real number and h is an unknown integer. If a target law consists of periodic function or discontinuous function, we cannot exactly discover it when Eq. (1) is assumed. However, if the range of each input variable is bounded, such a law can be closely approximated to a multivariate polynomial function with a finite number of terms. Moreover, since the power values are not restricted to integer, we can expect that the approximated polynomial function has a fewer number of terms. Hereafter, $(c_0, \cdots, c_h)^T$ and $(w_{i1}, \cdots, w_{in})^T$ are expressed as $\mathbf{c}$ and $\mathbf{w}_i$, respectively, where $\mathbf{a}^T$ means the transposed vector of $\mathbf{a}$. In addition, a vector consisting of all parameters, $(\mathbf{c}^T, \mathbf{w}_1^T, \cdots, \mathbf{w}_h^T)^T$ is simply expressed as $\mathbf{\Phi}$, and $N(= nh + h + 1)$ denotes the dimension of $\mathbf{\Phi}$.

By adding an adequate value to each element of the input vectors, if necessary, without losing generality we can assume $x_{ti} > 0$; then, Eq. (1) is equivalent to

$$y_t \quad = \quad c_0 + \sum_{i=1}^{h} c_i \exp\left(\sum_{j=1}^{n} w_{ij} \ln(x_{tj})\right). \tag{2}$$

Equation (2) can be regarded as the feedforward computation of a three-layer neural network where the activation function of each hidden unit is $\exp(s) = e^s$. Here h, $\mathbf{w}_i$, and $\mathbf{c}$ denote the number of hidden units, the weights between the input units and hidden unit i, and the weights between the hidden units and the output unit, respectively. Hereafter, the output value of hidden unit i is described as $v_i(\mathbf{x}_t; \mathbf{w}_i) = \exp\left(\sum_{j=1}^{n} w_{ij} \ln(x_{tj})\right)$ and then the output value of the output unit is described as $z(\mathbf{x}_t; \mathbf{\Phi}) = c_0 + \sum_{i=1}^{h} c_i v_i(\mathbf{x}_t; \mathbf{w}_i)$. The hidden units defined by $v(\mathbf{x}; \mathbf{w})$ are called *product units* [2]. The discovery of numeric laws subject to Eq. (1) can thus be defined as the following learning problem in neural networks. That is, the problem is to find the $\mathbf{\Phi}$ that minimizes the following objective function:

$$f(\mathbf{\Phi}) = \frac{1}{2}\sum_{t=1}^{m}(y_t - z(\mathbf{x}_t; \mathbf{\Phi}))^2. \qquad (3)$$

3. BPQ Algorithm

In our early experiments and as reported in earlier studies [9], the problem of minimizing Eq. (3) turned out to be quite tough. Thus, in order to efficiently and constantly obtain good results, this paper employs a new second-order learning algorithm called *BPQ* [14]; by adopting a quasi-Newton method [4, 10] as a basic framework, the descent direction, $\Delta\mathbf{\Phi}$, is calculated on the basis of a partial BFGS update and a reasonably accurate step-length, λ, is efficiently calculated as the minimal point of a second-order approximation. In first-order learning algorithms which calculate the search direction as the gradient direction, a large number of iterations are often required until convergence. On the other hand, in existing second-order methods [4, 10] which converge more quickly by using both gradient and curvature information, it is difficult to suitably scale up for large problems, and a large amount of computation is required for calculating the optimal step-length. BPQ can be reasonably scaled up by introducing a storage space parameter, and the computational complexity for calculating the optimal step-length is reasonably small, almost equivalent to that of gradient vector evaluation.

For the problem of minimizing Eq. (3), the partial BFGS update can be directly applied, while the basic procedure for calculating the step-length λ must be slightly modified. In the step-length calculation, since λ is the only variable, we can express $f(\mathbf{\Phi} + \lambda\Delta\mathbf{\Phi})$ simply as $\zeta(\lambda)$. Its second-order Taylor approximation is given as $\zeta(0) + \zeta'(0)\lambda + \frac{1}{2}\zeta''(0)\lambda^2$. When $\zeta'(0) < 0$ and $\zeta''(0) > 0$, the minimal point of this approximation is given by $\lambda = -\zeta'(0)/\zeta''(0)$. Here, the method for coping with the other cases is exactly the same as described in [14]. For three-layer neural networks defined by Eq. (3), we can efficiently calculate both $\zeta'(0)$ and $\zeta''(0)$ as follows. By differentiating $\zeta(\lambda)$ and substituting 0 for λ, we obtain $\zeta'(0) = -\sum_{t=1}^{m}(y_t - z(\mathbf{x}_t; \mathbf{\Phi}))z'(\mathbf{x}_t; \mathbf{\Phi})$, $\zeta''(0) = \sum_{t=1}^{m}\left((z'(\mathbf{x}_t; \mathbf{\Phi}))^2 - (y_t - z(\mathbf{x}_t; \mathbf{\Phi}))z''(\mathbf{x}_t; \mathbf{\Phi})\right)$. Now that the derivative of $z(\mathbf{x}_t; \mathbf{\Phi})$ is defined as $z'(\mathbf{x}_t; \mathbf{\Phi}) = \frac{d}{d\lambda}z(\mathbf{x}_t; \mathbf{\Phi} + \lambda\Delta\mathbf{\Phi})_{|\lambda=0}$, we obtain $z'_t = \Delta c_0 + \sum_{i=1}^{h}(\Delta c_i v_{it} + c_i v'_{it})$ and $z''_t = \sum_{i=1}^{h}(2\Delta c_i v'_{it} + c_i v''_{it})$, where $v'_i(\mathbf{x}_t; \mathbf{w}_i) = v_i(\mathbf{x}_t; \mathbf{w}_i)\sum_{j=1}^{n}\Delta w_{ij}\ln(x_{tj})$, $v''_i(\mathbf{x}_t; \mathbf{w}_i) = v'_i(\mathbf{x}_t; \mathbf{w}_i)\sum_{j=1}^{n}\Delta w_{ij}\ln(x_{tj})$, and Δc_i and Δw_{ij} are modification values corresponding to c_i and w_{ij}, respectively.

Incidentally, we can employ other second-order learning algorithms such as SCG [11] or OSS [1], but BPQ worked the most efficiently among them in our own experience.

4. Criterion for Selection

In general, for a given set of data, we cannot know the optimal number of hidden units in advance. Moreover, since the data is usually corrupted by noise, the law-candidate which minimizes Eq. (3) is not always the best one. We must thus consider a criterion to adequately evaluate the law-candidates discovered by changing the number of hidden units. In this paper, by assuming that the target output values are corrupted by Gaussian noise with a mean of 0 and an unknown standard deviation of σ, finding an adequate number of hidden units is formalized as a model selection problem of the maximum likelihood estimation problem. Thus, we adopt the MDL (Minimum Description Length) criterion [12] for this purpose. The MDL fitness value is defined by

$$\text{MDL} = 0.5m\log(\text{MSE}) + 0.5N\log(m), \qquad (4)$$

where MSE represents the value of the mean squared error defined by

$$\text{MSE} = \frac{1}{m}\sum_{t=1}^{m}(y_t - z(\mathbf{x}_t; \hat{\mathbf{\Phi}}))^2. \qquad (5)$$

Here, $\hat{\mathbf{\Phi}}$ is a set of weights which minimizes Eq. (3), N is the number of parameters in $\mathbf{\Phi}$, and m is the number of examples. Hereafter, our discovery method employing the connectionist problem formalization, the BPQ algorithm and the MDL criterion, is called *RF5* (*Rule extraction from Facts version 5*).

5. Evaluation by Experiments

5.1. Artificial data

The rule discovery method, RF5, was evaluated by using an artificial problem proposed by Sutton and Matheus [17] and our modified version.

The original problem is to restore a law described as

$$y = 2 + 3x_1x_2 + 4x_3x_4x_5, \qquad (6)$$

while a law for the modified problem is described as

$$y = 2 + 3x_1^{-1}x_2^3 + 4x_3x_4^{1/2}x_5^{-1/3}. \qquad (7)$$

Each example is generated as follows: each value of variables $x_1, \cdots, x_5$ is randomly generated in the range of $[0, 1]$, and the corresponding value of y is calculated using Eq. (6) or (7). In these problem, the total number of variables is 9 ($n = 9$). Each value of irrelevant variables $x_6, \cdots, x_9$ is also randomly generated in the range of $[0, 1]$, each value of y is corrupted by adding noise generated according to a normal distribution with a mean of 0 and a standard deviation of 0.1, and the number of examples is set to 200 ($m = 200$).

In the experiments, the initial values for the weights between the input and hidden units were independently generated according to a normal distribution with a mean of 0 and a standard deviation of 1; the initial values for the weights between the hidden and output units were set to 0, but the bias value at the output unit was initially set to the average output value of all training examples. The iteration was terminated when the gradient vector was sufficiently small, i.e., $\|\nabla f(\mathbf{\Phi})\|^2/N < 10^{-8}$, or the total processing time exceeded 100 seconds.

Table 1: Learning statistics (artificial data)

	hidden	MSE value			MDL value			iteration		time	
	unit	best	avg.	s.d.	best	avg.	s.d.	avg.	s.d.	avg.	s.d.
original problem	h = 1	0.160	0.160	0.000	-154.0	-154.0	0.0	68	4	0.31	0.02
	h = 2	0.009	0.009	0.000	-416.0	-416.0	0.0	93	9	0.77	0.07
	h = 3	0.008	0.008	0.000	-405.5	-405.3	0.8	784	82	9.80	1.02
modified problem	h = 1	2.326	2.326	0.000	113.6	113.6	0.0	90	12	0.41	0.05
	h = 2	0.010	0.031	0.207	-403.9	-398.6	53	228	134	1.90	1.11
	h = 3	0.009	0.009	0.000	-388.9	-384.9	1.2	753	127	9.41	1.58

In the experiments, we changed the number of hidden units from 1 to 3 ($h = 1, 2, 3$) and performed 100 trials for each of them. Table 1 shows the basic statistics of MSE values, MDL values, iterations, and processing times (sec.).[1] The best MSE values were minimized when $h = 3$, while the best MDL values were minimized when $h = 2$: this indicates the correct number of hidden units was found again for the both problems. The original and modified laws discovered by RF5 were

$$\begin{aligned} y &= 1.97 + 3.03x_1^{1.00}x_2^{0.97}x_4^{-0.01}x_5^{-0.01}x_7^{0.01}x_8^{-0.01} \\ &\quad +3.88x_1^{-0.03}x_2^{-0.01}x_3^{1.03}x_4^{1.00}x_5^{1.05}x_6^{-0.01}x_7^{-0.02} \\ y &= 2.01 + 3.00x_1^{-1.00}x_2^{3.00} \\ &\quad +3.98x_3^{1.02}x_4^{0.50}x_5^{-0.33}x_6^{-0.01}x_9^{-0.01}. \end{aligned}$$

where the weight values were rounded off to the second decimal place. When the weight values were rounded off to the first decimal place, these laws become

$$\begin{aligned} y &= 2.0 + 3.0x_1^{1.0}x_2^{1.0} + 3.9x_3^{1.0}x_4^{1.0}x_5^{1.0} \\ y &= 2.0 + 3.0x_1^{-1.0}x_2^{3.0} + 4.0x_3^{1.0}x_4^{0.5}x_5^{-0.3}. \end{aligned}$$

Although some weight values were slightly different, laws almost equivalent to the true ones were found. This shows that RF5 is robust and noise tolerant to some degree. Note that without preparing some appropriate prototype functions, existing numeric discovery methods cannot find such laws as described in Eq. (7). This point is an important advantage of RF5 over existing methods.

5.2. Real data

For real data, we used three data sets supporting Hagen-Rubens' law, Kepler's third law, and Boyle's law.[2] In this experiment, since the number of examples for each data set was small, the number of hidden units was fixed at 1. Note that since we consider a constant term c_0, the problem cannot be reduced to a simple regression problem. The trials were performed 10 times for each data set, and the weight values in the discovered laws were rounded off to the second decimal place. The other experimental conditions were exactly the same as before.

Hagen-Rubens' law is a relation among the frequency ν of incident light, the electrical conductivity σ of metal, and the reflectance R. The original law is described as

$$R = 1 - 2\left(\frac{\nu}{\sigma}\right)^{\frac{1}{2}}.$$

The law discovered by RF5 is

$$R = 1.00 - 2.08\nu^{0.57}\sigma^{-0.57}.$$

Although the data has some outstanding outlier examples, the discovered law is very similar to the reference relation.

Kepler's third law is a relation between the distance r to the sun and the revolution period T of five planets. The original law is described as

$$T = 0.41r^{1.5}.$$

The law discovered by RF5 is

$$T = 0.41r^{1.50} + 0.19.$$

An undesired constant term appeared here, but the law is very similar to the reference relation. The value of the constant term is small in the discovered law; as such we can perform another trial using the model where the constant term is omitted.

Boyle's law is a relation between the pressure p and volume V of a quantity of enclosed air. The original law is described as

$$V = 29.30/p.$$

The law discovered by RF5 is

$$V = 29.05p^{-1.08} - 0.61.$$

5.3. Time series data

If time series data $\{x_1, \cdots, x_t, \cdots\}$ are generated according to a nonlinear difference equation such as $kx_t = x_{t-1}(1 - x_{t-1})$ (k is a constant and $x_0 \in [0, 1]$), then, it is known that we can observe a chaotic behavior. Since our target law defined by Eq. (1) is a generalized polynomial function, it is expected that the law of chaotic time series can be identified by RF5. Hereafter, a nonlinear autoregressive function defined by

$$x_t = z(x_{t-1}, \cdots, x_{t-\tau}; \mathbf{\Phi}), \tag{8}$$

is considered, where $z(\cdot)$ denotes the right hand-side of Eq. (2) and τ is the length of delayed inputs. For real time series data, we used a data set recorded from a Far-Infrared-Laser in a chaotic state [18] [3], where we used a series of 1,000 training points (Data Set A) and τ was set to 8.

In the experiments, we changed the number of hidden units from 1 to 7 and performed 10 trials for each of them. The

[1] Our experiments were done on HP/9000/735 computer.

[2] We obtained the sets of Kepler's data and Boyle's data from the UCI repository of machine learning databases.

[3] This data set was used in the Santa Fe Institute Time Series Prediction and Analysis Competition.

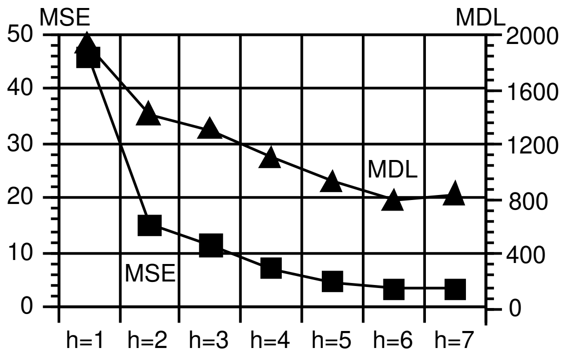

Figure 1: learning results

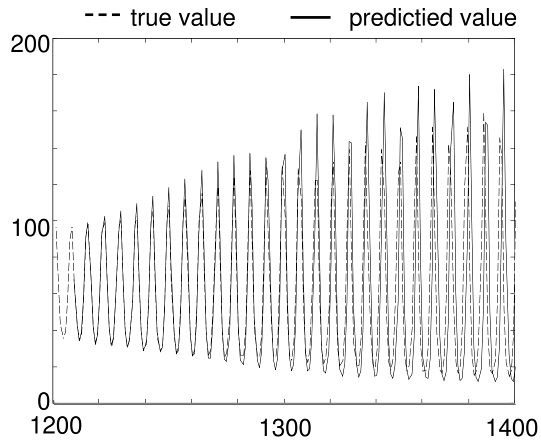

Figure 2: prediction by discovered law

other experimental conditions were exactly the same as before. Figure 1 shows the best MSE values and MDL values. This indicates the best MDL values were minimized when $h = 6$. In this experiments, the law discovered by RF5 was not able to outperform the best result reported in the competition [18], and we must go further to improve our preliminary result. Actually, for the successive 100 test points, the prediction performance of the discovered law was the 9th position among the 14 results submitted to the competition. However, the discovered law shows a good short-term prediction performance for the unknown data. Figure 2 shows a part of the predictions by the discovered law, where only first 8 points were given to the law. We can see that the predicted points are very close to the true ones.

6. Conclusion

To discover an underlying law from a set of numeric data, we have proposed a new connectionist method called RF5. After employing the connectionist problem formalization, RF5 adopts a second-order learning algorithm BPQ for training and the MDL criterion for model selection. Experiments showed that RF5 successfully discovered underlying laws whose power values are not restricted to integers, even if the data contained irrelevant variables or a small amount of noise. In the future, we plan to do further experiments to evaluate the proposed method using a wider variety of problems.

References

[1] R. Battit. First- and second-order methods for learning between steepest descent and newton's method. *Neural Computation*, 4(2):141–166, 1992.

[2] R. Durbin and D. Rumelhart. Product units: a computationally powerful and biologically plausible extension. *Neural Computation*, 1(1):133–142, 1989.

[3] B.C. Falkenhainer and R.S. Michalski. Integrating quantitative and qualitative discovery in the abacus system. In Y. Kodratoff and R.S. Michalski, editors, *Machine learning: an artificial intelligence approach, Volume* III, pages 153–190. Morgan Kaufmann, San Mateo, CA, 1990.

[4] P.E. Gill, W. Murray, and M.H. Wright. *Practical optimization.* Academic Press, London, 1981.

[5] P. Langley. Bacon.1: a general discovery system. In *Proeeedings of the Seeond National Conferenee of the Canadian Soeiety for Computational Studies of Intelligenee*, pages 173–180, 1978.

[6] P. Langley, H.A. Simon, G. Bradshaw, and J. Zytkow. *Scientific discovery: computational explorations of the creative process.* MIT Press, Cambridge, MA, 1987.

[7] P. Langley and J. Zytkow. Data-driven approaches to empirical discovery. *Artificial Intelligence*, 40:283–312, 1989.

[8] P. Langley and J. Zytkow. A robust approach to numeric discovery. In *Proc. seventh International Machine Learning Conference*, pages 411–418, Austin, Texas, 1990.

[9] L.R. Leerink, C.L. Giles, B.G. Horne, and M.A. Jabri. Learning with product units. In G. Tesauro, D.S. Touretzky, and T.K. Lee, editors, *Advances in Neural Information Processing Systems 7.* MIT Press, Cambridge, MA, 1995.

[10] D.G. Luenberger. *Linear and nonlinear programming.* Addison-Wesley, Reading, MA, 1984.

[11] M.F. Møller. Supervised learning on large redundant training sets. *International Journal of Neural Systems*, 4(1):15–25, 1993.

[12] J. Rissanen. *Stochastic complexity in statistical inquiry.* World Scientific, 1989.

[13] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing*, pages 318–362. MIT Press, 1986.

[14] K. Saito and R. Nakano. Partial BFGS update and efficient step-length calculation for three-layer neural networks. *Neural Computation*, 9(1):239–257, 1997.

[15] T.D. Sanger. Basis-function trees as a generalization of local variable selection method for function approximation. In D.S. Touretzky, editor, *Neural Information Processing Systems 3*, pages 707–713. Morgan Kaufmann, San Mateo, CA, 1991.

[16] C. Schaffer. Bivariate scientific function finding in a sampled, real-data testbed. *Machine Learning*, 12(1/2/3):167–183, 1993.

[17] R.S. Sutton and C.J. Matheus. Learning polynomial functions by feature construction. In *Proceedings of the Eighth International Machine Learning Workshop*, pages 208–212, Evanston, IL, 1991.

[18] A.S. Weigend and N.A. Gershenfeld. *Time series prediction: forcasting future and understanding the past.* Addison-Wesley, 1994.