CASE 68

Technique for the Evaluation of Programming Ability Based on the MTS

Abstract: In our research, by advancing Takada's research [1] to some extent, we set up persons creating a normal program in a base space, judged directions of two abnormal data groups using the Mahalanobis–Taguchi system (MTS) method with Schmidt's orthogonalization [2], and proposed a method of assessing a programmer's ability with data from questionnaires about lifestyle-related matters.

1. Introduction

In a software development process, a programmer's ability is regarded as an important human factor that affects software quality directly. Currently, many companies have adopted a programmer aptitude test as a method of evaluating a programmer's potential. However, in fact, approximately 30% of them are not making the best of the programmer aptitude test [3]. The main reason is its reliability problem: that the test result is not relevant to a programmer's ability or that the criteria for judgment of aptitude are not clear. Therefore, establishment of an effective method of evaluating a programmer's ability has been desired; in fact, many attempts have been made. Among them is Takada's and other researchers' evaluation method of programming ability using the MTS method.

Using two types of programs, mask calculation and sorting programs, we gathered data for programming ability. Table 1 shows the difference between the two programs. Lifestyle-related data were collected through questionnaires, which consist of 56 questions and the four items of age, gender, programming experience, and computer experience (from now on, this questionnaire's item is simply called an "item").

Combining the data for sorting and mask calculation programs, we obtained 62 sets of data. By adding the data used in Takada's research, the total number of data amounts to 153.

2. Setup of Normal and Abnormal Data

In the study by Takada and others, the people who do not have a high level of programming ability were considered as the base space, since the people with high ability were considered as abnormal. But since the people with low ability are also abnormal, the base space was defined as the people who write normal programs. It is a fact that the programs produced by the people with high ability are generally difficult to use and do not fit well in a large system.

A program created by a testee was judged by two examiners (Suzuki and Takada). A testee who created a normal program was classified as a normal datum, or a datum belonging to a base space, whereas testees making a good and poor programs were categorized as abnormal data. It can be seen from Table 2 that there are not many people in the "good program" category, indicating the difficulty of using these people to construct a base space.

3. Calculation of Mahalanobis Distance Using Schmidt's Orthogonal Variable

A Schmidt orthogonal expansion's calculation process is shown below. Using measurements contained in a base space, we calculated a mean and standard deviation for each item and computed normalized

Taguchi's Quality Engineering Handbook. Genichi Taguchi, Subir Chowdhury and Yuin Wu
Copyright © 2005 Genichi Taguchi, Subir Chowdhury, Yuin Wu.

Table 1
Experimental difference between two programs

	Sorting Program	Mask Calculation Program
Testee	Student	Businessperson and student
Programming time	Limited	Unlimited
Programming language	C	Any language

variables, $X_1, X_2, \ldots, X_{60}$. Then, with $X_1, X_2, \ldots, X_{60}$, we created orthogonalized variables, $x_1, x_2, \ldots, x_{60}$.

$$x_1 = X_1 \tag{1}$$

$$x_2 = X_2 - b_{21}x_1 \tag{2}$$

$$\vdots$$

$$\mathrm{x}_{60} = X_{60} - b_{601}x_1 - b_{602}x_2 - \cdots - b_{6059}x_{59} \tag{3}$$

Next we set each variance of $x_1, x_2, \ldots, x_{60}$ to $V_1, V_2, \ldots, V_{60}$. The normalized orthogonal variables, $Y_1, Y_2, \ldots, Y_{60}$, are expressed as follows:

$$Y_1 = \frac{x_1}{\sqrt{V_1}} \tag{4}$$

$$Y_2 = \frac{x_2}{\sqrt{V_2}} \tag{5}$$

$$\vdots$$

$$Y_{60} = \frac{x_{60}}{\sqrt{V_{60}}} \tag{6}$$

Table 3 summarizes the calculation results above.

Next, for measurements of testees creating good and poor programs, we computed $Y_1, Y_2, \ldots, Y_{60}$, as shown in Table 4. Then the Mahalanobis distance D^2 was computed by $Y_1, Y_2, \ldots, Y_{60}$:

$$D^2 = \frac{1}{60}\left(Y_1^2 + Y_2^2 + \cdots + Y_{60}^2\right) \tag{7}$$

Table 5 shows the calculation results of Mahalanobis Distances for the base space and abnormal data.

4. Selection of Items for Measurement Ability Using the Larger-the-Better Characteristic

As a next step, we judged the direction of a distance and set up each item's order. It is important in using the Schmidt method to determine the order of items based on their importance as well as on cost, but these were unknown because each item in our research was a question. Next, selecting items by a larger-the-better characteristic based on an orthogonal array, after creating response graphs, we placed all items in descending order with respect to the magnitude of a factor effect. (Level 1 was used for creating a standard space. Since level 2 was not used for the purpose, items with decreasing value from left to right are effective for judgment.) Figures 1 and 2 show the results. The former is the item-selection response graph for testees creating a good program, and the latter is that for those creating a poor one.

According to Figure 1, we note that items 28, 2, 20, and 46 are effective for testees creating a good program. In contrast, Figure 2 reveals that effective items for those creating a poor program are 29, 32, 36, and 30. These results highlight that there is an

Table 2
Classification of normal and abnormal data for software programming

	Normal Data, Normal Program	Abnormal Data	
		Good Program	Poor Program
Judgment criterion	A program after removing abnormal data	A program that contains no bugs and understandable to others	A program that contains fatal bugs
Number of Data	102	14	23

Table 3
Database for the base space

	Item 1	Item 2	···	Item 60
Measurement	2 3 ⋮ −2	−1 1 ⋮ −1	···	5 6 ⋮ 7
Average	0.24	0.66	···	6.42
Standard deviation	1.72	1.37	···	4.77
Normalized variable *X*	1.02 1.60 ⋮ −1.30	−1.21 0.25 ⋮ −1.21	···	−0.98 0.54 ⋮ 0.33
Regression coefficient *b*	—	0.12	···	−0.10 −0.12 ⋮ 0.70
Variance *V*	1	0.99	···	0.10
Normalized orthogonal variable *Y*	1.02 1.60 ⋮ −1.30	−1.34 0.06 ⋮ −1.07	···	−1.20 0.04 ⋮ 1.64

Table 4
Normalized orthogonal variables of abnormal data

	Item 1	Item 2	···	Item 60
Measurement				
Good program	−2 2 ⋮ 0	0 0 ⋮ −2	···	5 6 ⋮ 7
Poor program	0 1 ⋮ 1	0 2 ⋮ 1	···	2 3 ⋮ 7
Orthogonal variable				
Good program	−1.30 1.02 ⋮ −0.14	−0.33 −0.61 ⋮ −1.94	···	5.18 −3.98 ⋮ −5.62
Poor program	−0.14 0.44 ⋮ 0.44	−0.47 0.93 ⋮ 0.20	···	−2.50 1.94 ⋮ 5.01

obvious difference between those creating good and poor programs. By performing a Schmidt orthogonal expansion on the data for those creating good and poor programs, we rearranged each item's order.

5. Determination of Item Order by Normalized Orthogonal Variables

In accordance with the procedure below, using orthogonal normalized variables $Y_1, Y_2, \ldots, Y_{60}$, we computed larger-the-better SN ratios $\eta_1, \eta_2, \ldots, \eta_{60}$ for $A_1, A_2, \ldots, A_{60}$, and rearranged each item's order such that we could eliminate items leading to a smaller SN ratio:

$$A_1 = \text{only } Y_1 \text{ is used}$$

$$A_2 = Y_1 \text{ and } Y_2 \text{ are used}$$

$$\vdots$$

$$A_k = Y_1, Y_2, \ldots, Y_{60} \text{ are used}$$

Then we detailed calculation of the SN ratio for testees creating a good program (refer to Table 6) as follows.

SN ratio for A_1:

$$\eta_1 = -10 \log \frac{1}{14}\left(\frac{1}{1.20^2} + \frac{1}{1.20^2} + \cdots + \frac{1}{4.99^2}\right) = -5.90 \text{ dB} \quad (8)$$

SN ratio for A_2:

$$\eta_2 = -10 \log \frac{1}{14}\left(\frac{1}{0.86^2} + \frac{1}{1.38^2} + \cdots + \frac{1}{3.57^2}\right) = -5.21 \text{ dB} \quad (9)$$

SN ratio for A_{60}:

$$\eta_{60} = -10 \log \frac{1}{14}\left(\frac{1}{2.18^2} + \frac{1}{2.52^2} + \cdots + \frac{1}{1.54^2}\right) = 4.07 \text{ dB} \quad (10)$$

Figure 3 shows the relationship between items and SN ratios when using the data for testees creating a good program, and the relationship for those creating a poor program. Up to this point, calculations were made using software from the Oken Company.

Table 5
Mahalanobis distance of abnormal data calculated from a base space constructed by the group of people who wrote normal programs

Distance	Base Space	Good Program	Poor Program
0	0	0	0
0.5	2	0	0
1	49	0	1
1.5	51	1	2
2	0	2	1
2.5	0	4	5
3	0	4	3
3.5	0	0	4
4	0	0	4
4.5	0	1	1
5	0	1	2
5.5	0	0	0
6	0	0	0
6.5	0	1	0
7	0	0	0
7.5	0	0	0
8	0	0	0
8.5	0	0	0
9	0	0	0
9.5	0	0	0
10	0	0	0
Total	102	14	23

6. Directional Judgment by Schmidt Orthogonal Expansion

By taking advantage of the rearranged data, we made a judgment on the direction of abnormal data as to whether the abnormal data were abnormal in a favorable or unfavorable direction: whether abnormally good or abnormally bad. Since we did not have enough data for the base space, it was decided

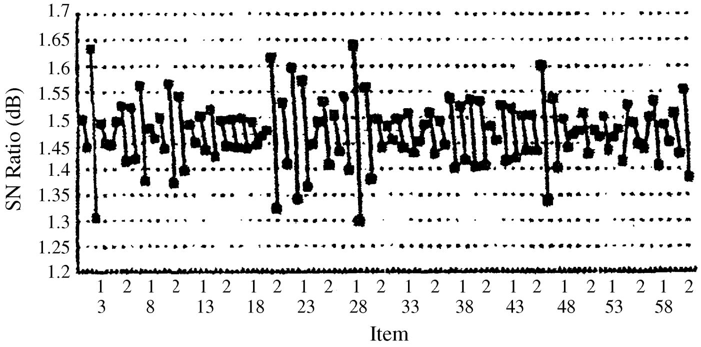

Figure 1
Response graphs for item selection (for testees creating a good program)

to judge one side at a time rather than judging the sides at the same time. As a first step, we made a separate directional judgment for each piece of the data for those creating good and poor programs.

After each type was evaluated and classified into two levels, we set up the levels of a signal, m, as follows:

m_1: most favorable = 2 (8 testees matched)

m_2: favorable = 1 (6 testees matched)

m_3: unfavorable = −1 (8 testees matched)

m_4: most unfavorable = −2 (15 testees matched)

As a next step, we calculated the normalized orthogonal variables Y's for each item for those creating good and poor programs. Because the number of items was 60, we calculated Y_1, Y_2, ... , Y_{60}. We rearranged the data for those creating a good program and computed the normalized orthogonal variables for abnormal data (Table 7).

Note the following calculation example of β for the first item after rearranged for those creating a good program: For the first item we computed the average of Y_1's, denoted by $\overline{Y}_1$ for the data belonging to each of m_1 and then computed the same for m_2, m_3, and m_4. Then, for signals, setting the true value as m and the output normalized orthogonal variables as Y, we applied a zero-proportional equation $Y = \beta m$. Computing each $\overline{Y}_1$ belonging to each of m_1, m_2, m_3, and m_4, we estimated β_1 by the least squares method.

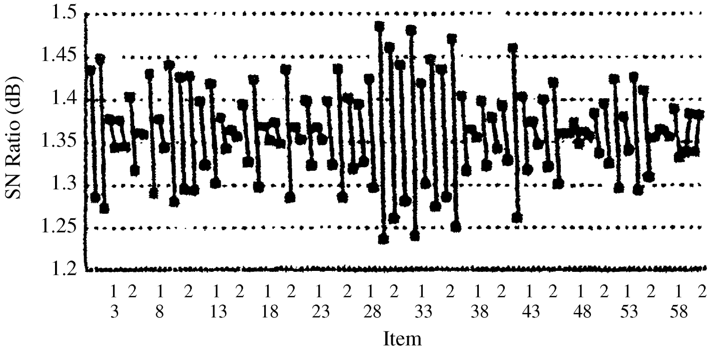

Figure 2
Response graphs for item selection (for testees creating a poor program)

Table 6
Rearranging data for testees creating a good program

No.	A_1	A_2	···	A_{60}
1	1.20	0.86	···	2.18
2	1.20	1.38	···	2.52
⋮	⋮	⋮	···	⋮
8	0.69	1.94	···	1.53
9	1.16	0.96	···	2.03
10	0.22	0.18	···	1.72
⋮	⋮	⋮	···	⋮
14	4.99	3.57	···	1.54

$\overline{Y_1}$ for data belonging to m_1:

$$\overline{Y_{1m1}} = \frac{-1.20 + \cdots + 0.69}{8} = -0.38 \qquad (11)$$

$\overline{Y_1}$ for data belonging to m_2:

$$\overline{Y_{1m2}} = \frac{1.16 + \cdots + (-4.99)}{6} = -0.49 \qquad (12)$$

$\overline{Y}$ for data belonging to m_3:

$$\overline{Y_{1m3}} = \frac{-0.73 + \cdots + 0.22}{8} = -0.02 \qquad (13)$$

$\overline{Y_1}$ for data belonging to m_4:

$$\overline{Y_{1m4}} = \frac{-0.26 + \cdots + 1.16}{15} = 0.34 \qquad (14)$$

$$\beta_1 = \frac{m_1\overline{Y_{1m1}} + m_2\overline{Y_{1m2}} + m_3\overline{Y_{1m3}} + m_4\overline{Y_{1m4}}}{m_1^2 + m_2^2 + m_3^2 + m_4^2}$$

$$= -0.19 \qquad (15)$$

In a similar manner, we computed $\beta_2, \beta_3, \ldots, \beta_{60}$. Then, for $Y_1, Y_2, \ldots, Y_{60}$ (abnormal data), whose directions we attempted to judge, for example, if we define the direction of the first item as M_1, we know the direction of M_1 by using the relationship $Y_1 = \beta_1 M_1$ because all of $\beta_1, \beta_2, \ldots, \beta_{60}$ are known. Similarly, determining $M_2, \ldots, \mathrm{M}_{60}$ for $Y_2, \ldots, Y_{60}$ with $\beta_2, \ldots, \beta_{60}$, we calculated the sum of M's as the direction of the data's distances, M. Since we defined the signals for testees creating a good program as +2 and +1, and those for testees creating a poor program as −2 and −1, if testees creating a good program have a positive direction and testees creating a poor program have a negative direction, our judgment method can be regarded to work properly. Table 8 shows the judgment result for both of the cases of using the data for testees creating good or poor programs.

Table 8 reveals that in the case of using the data for those creating a good program, 8 of 14 testees creating a good program and 12 of 23 creating a poor program were judged correctly. On the other hand, in the case of using the data for testees creating a poor program, 8 of 14 creating a good program and 13 of 23 creating a poor program were

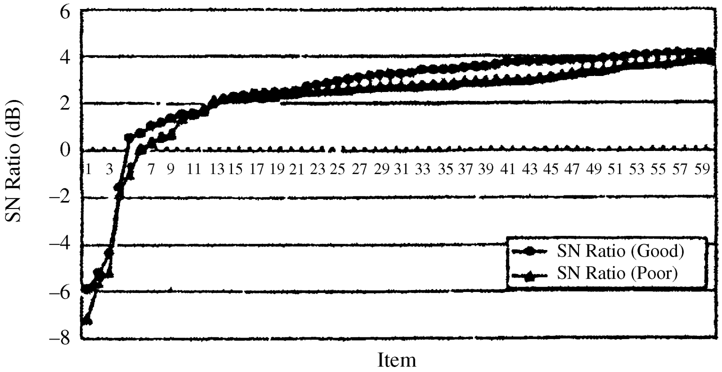

Figure 3
Rearrangement of items by Schmidt orthogonal expansion

Table 7
Normalized orthogonal variables of abnormal data

			Normalized Orthogonal Variable			
	No.	*m* Level	Y_1	Y_2	···	Y_{60}
Testees creating good program	1	m_1	−1.20	−0.35	···	−2.84
	2	m_1	−1.20	−0.35	···	−5.38
	⋮	⋮	⋮	⋮	···	⋮
	8	m_1	0.69	−2.04	···	−0.98
	9	m_2	1.16	1.59	···	−0.85
	10	m_2	0.22	−1.98	···	2.87
	⋮	⋮	⋮	⋮	···	⋮
	14	m_2	−4.99	−1.39	···	0.92
Testees creating poor program	1	m_2	−0.73	1.07	···	1.71
	2	m_3	0.22	−1.98	···	0.53
	⋮	⋮	⋮	⋮	···	⋮
	15	m_3	0.22	0.23	···	2.06
	16	m_4	−0.26	−0.46	···	−3.67
	17	m_4	0.22	0.96	···	−1.69
	⋮	⋮	⋮	⋮	···	⋮
	23	m_4	1.16	0.85	···	−2.19

judged correctly. However, the data do not necessarily indicate an accurate judgment.

Item Selection for Programming Ability Evaluation

To improve the accuracy in judgment, by using the SN ratio, we evaluated item selection for programming ability evaluation whether or not outputs for signals from −2 to +2 were stable (the linearity of $Y = \beta M$). For the first item obtained after rearranging the data for testees creating a good program, we calculated the SN ratio as below.

Total variation:

$$S_T = \overline{Y_{1M1}}^2 + \overline{Y_{1M2}}^2 + \overline{Y_{1M3}}^2 \text{ pl } \overline{Y_{1M4}}^2 = 0.50 \qquad (16)$$

Variation of β:

$$S_\beta = \frac{(M_1\overline{Y_{1M1}} + M_2\overline{Y_{1M2}} + M_3\overline{Y_{1M3}} + M_4\overline{Y_{1M4}})^2}{r}$$

$$= 0.36 \qquad (17)$$

Effective divider:

$$r = M_1^2 + M_2^2 + M_3^2 + M_4^2 = 10 \qquad (18)$$

Error variation:

$$S_e = S_T - S_\beta = 0.14 \qquad (19)$$

Error variance:

$$V_e = \frac{S_e}{f} = 0.05 \qquad (f = 3) \qquad (20)$$

SN ratio:

$$\eta = 10 \log \frac{(1/r)\ (S_\beta - V_e)}{V_e} = -1.62 \text{ dB} \qquad (21)$$

Similarly, we computed the SN ratios for items 2 through 60. By picking up stable items from all of the SN ratios (no item with an SN ratio of less than −10 dB was used), we calculated the directions of distances again. Table 9 shows the result.

The table demonstrates that in the case of using the data for those creating a good program, 12 of 14 testees creating a good program and 17 of 23 creating a poor program were judged correctly. The number of items used for this analysis was 17.

On the other hand, in the case of using the data for testees creating a poor program, for 12 of 14

Table 8
Directional judgment using all items

Good Program No.	*M* for Good Program	*M* for Poor Program	Poor Program No.	*M* for Good Program	*M* for Poor Program
1	2125.61	780.12	1	12.51	391.49
2	3973.87	−527.55	2	118.96	−10.84
3	−4842.26	428.87	3	7785.43	316.39
4	−1027.02	140.29	4	−2097.13	−1702.27
5	−3437.98	897.78	5	−3453.95	1185.03
6	−3258.63	631.64	6	−2755.32	−1036.88
7	5805.01	−594.37	7	−1602.08	966.77
8	5062.20	−753.75	8	−1070.22	−731.54
9	−2984.79	−1713.75	9	7412.83	192.16
10	4531.27	464.39	10	−2571.95	713.33
11	1095.78	−321.21	11	1006.82	1890.32
12	4209.38	1084.77	12	905.96	−216.42
13	−1011.24	−670.23	13	6814.81	−2039.88
14	2580.05	170.22	14	1184.24	−143.26
			15	−2647.86	−1052.03
			16	5393.82	121.56
			17	589.65	−964.47
			18	3003.96	−248.19
			19	−350.94	385.61
			20	456.86	10.69
			21	3193.06	−71.63
			22	−3413.29	253.36
			23	−1104.85	−287.06

creating a good program and 15 of 23 creating a poor program, were correctly judged. The number of items used was 11. Consequently, these results prove that the accuracy in judgment was improved compared with the former analysis.

After obtaining the results that single-directional judgment produced satisfactory accuracy, we attempted two-directional judgment. More specifically, after combining the data for testees creating good and poor programs into a single set of abnormal data, we selected items by using an orthogonal array, rearranged them according to the response graphs shown in Figure 4, and reordered them by a Schmidt orthogonal expansion (figure 5) such that items leading to a smaller SN ratio can be eliminated.

Table 9
Directional judgment using only stable items

Good Program No.	*M* for Good Program	*M* for Poor Program	Poor Program No.	*M* for Good Program	*M* for Poor Program
1	11.92	48.48	1	−60.26	−19.46
2	0.64	6.49	2	39.77	4.19
3	104.51	72.78	3	−80.13	−38.91
4	10.62	13.21	4	−10.76	−33.91
5	95.44	7.22	5	−28.88	−4.31
6	−11.47	14.04	6	14.21	7.94
7	16.24	24.18	7	26.12	18.67
8	34.97	5.35	8	−88.30	−25.71
9	29.00	−31.36	9	−44.31	19.67
10	37.79	22.85	10	−72.51	−53.59
11	40.22	32.68	11	−16.22	−3.78
12	26.22	−1.93	12	−24.20	−25.94
13	33.57	14.21	13	−13.18	−17.04
14	87.14	61.13	14	−52.91	3.27
			15	−12.71	2.45
			16	10.68	11.67
			17	32.93	−12.76
			18	−66.87	−93.26
			19	−59.76	−33.73
			20	−61.35	−22.68
			21	−15.74	−16.04
			22	74.36	10.28
			23	−38.94	−26.52

Then, based on the rearranged order of the items, we made a directional judgment focused on the SN ratio, indicating a linear trend. Table 10 shows the result of directional judgment on abnormal data.

Table 10 shows that 13 of 14 testees creating a good program and 19 of 23 creating a poor program were judged correctly. Eleven items were used for this analysis. Although initially, we attempted to make a separate directional judgment on each group of testees creating good and poor programs, because of the small number of data used, the two groups' combined double-directional judgment resulted in higher accuracy.

7. Results

Observing the Mahalanobis distances of normal (base space) and abnormal data, it was found that

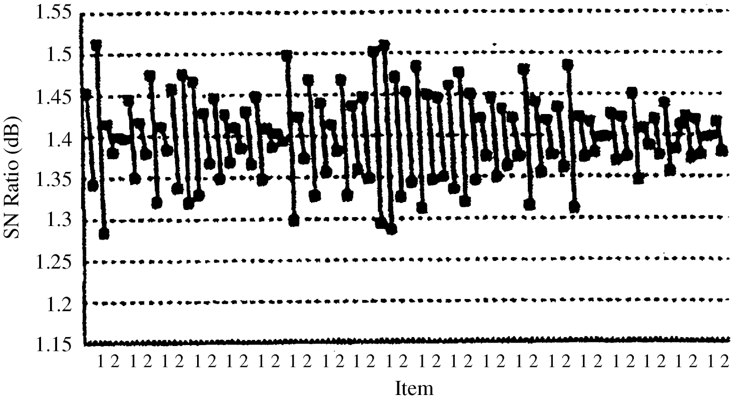

Figure 4
Response graphs for item selection

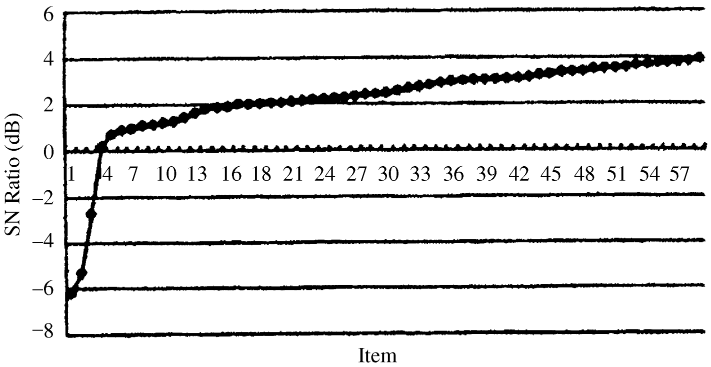

Figure 5
Rearrangement of items by Schmidt orthogonal expansion

the center of distribution of each case was separate; therefore, it is possible to distinguish between normal and abnormal data. Regarding the directional judgment for abnormal data, its accuracy was poor if all items were used. As an alternative approach, the SN ratio was calculated to observe the linearity of $Y = \beta M$. Then those items with good linearity were used. It means that the items useful for discrimination of normal/abnormal data were not necessarily useful for the judgment of direction unless they have linearity on $y = \beta m$. As a result, judging accuracy was improved and the direction of all abnormal data was judged correctly for almost all the data.

In our experiment, two types of abnormal data were used. By taking advantage of a Schmidt orthogonal expansion, with direction expressed by a positive or negative sign, we judged successfully whether the abnormal data were abnormal in a favorable or unfavorable direction. According to this result, after creating a base space with data for persons creating a normal program, we can assess each programmer's ability by comparing abnormal data for persons creating good and poor programs with the base space. The remaining issue is to form a more stable base space consisting of a larger number of data. Additionally, while we classified a signal, m, into four groups, to improve accuracy in directional judgment, we should set up a separate signal for a different programmer.

Table 10
Directional judgment using abnormal data

Good Program No.	Direction of Distance *M*	Poor Program No.	Direction of Distance *M*
1	53.17	1	−40.63
2	18.92	2	9.31
3	55.64	3	−45.33
4	3.11	4	−18.44
5	28.49	5	−23.91
6	7.04	6	19.25
7	17.66	7	−7.10
8	12.49	8	−38.26
9	−29.90	9	17.12
10	15.67	10	−35.41
11	23.18	11	−5.86
12	0.47	12	−29.15
13	7.77	13	−12.16
14	60.18	14	9.74
		15	4.83
		16	−3.24
		17	−4.79
		18	−61.08
		19	−29.90
		20	−11.70
		21	−16.68
		22	−3.69
		23	−22.33

References

1. Kei Takada, Kazuhito Takahashi, and Hiroshi Yano, 1999. A prediction on ability of programming from questionnaire using MTS method. *Quality Engineering*, Vol. 7, No. 1, pp. 65–72.
2. Genichi Taguchi, 1997. Application of Schmidt's orthogonalization in Mahalanobis space. *Quality Engineering*, Vol. 5, No. 6, pp. 11–15.
3. Hiroshi Miyeno. 1991. *Aptitude Inspection of Programmers and SE*. Tokyo: Ohmsha, pp. 105–106.
4. Takayuki Suzuki, Kei Takada, Muneo Takahashi, and Hiroshi Yano, 2000. A technique for the evaluation of programming ability based on MTS. *Quality Engineering*, Vol. 8, No. 4, pp. 57–64.

This case study is contributed by Takayuki Suzuki, Kei Takada, Muneo Takahashi, and Hiroshi Yano.