# Chapter Six

## A SOFTWARE TOOL FOR THE STATISTICAL ANALYSIS OF LANGUAGE TESTS*

The statistical software tool described here was designed to assist in the analysis of the results of tests which are administered as part of the entrance exam in English at Adam Mickiewicz University's Institute of English. The major objective of the exam, which in addition to a battery of tests includes an interview, is the selection of the best candidates for English studies. This explains the emphasis placed in the workshop on the statistics of item analysis, that is on evaluating test items in terms of their effectiveness. A description of item analysis will be the focus of the first part of this chapter. Then I will describe the role of the computer in analyzing test results and give some examples of authentic analyses for discussion. The examples come from the 1986 tests the results of which were analyzed using a series of programs I wrote for the Commodore 64 microcomputer. Finally, I will close with a few remarks on some of the hardware and software options.

### OVERVIEW OF THE STATISTICAL ANALYSIS OF LANGUAGE TESTS

Test designers need a number of standard statistical analyses such as the mean and standard deviation for individual tests as well as for the whole battery. These can easily be obtained using a desk calculator, especially one supplied with the major statis-

---

* Michał Jankowski, Adam Mickiewicz University, Poznań

tical functions. For descriptions of traditionally applied measures of central tendency and variability see elswhere in this book or consult (Butler, 1985). In addition to this, some graphic representation of the distribution of the results is desirable. This has been traditionally done for the overall results of the whole battery by entering a tally beside the appropriate score figure on a sheet of paper. The result looks more or less like this:

.
.
.
56 III
55 IIIIIIII
54 IIIII
53 II
.
.
.

For a description of a similar procedure, see [Harris, 1969: 136 -137].

Finally, test designers perform a detailed item analysis. For a more thorough discussion of item analysis than what follows, the reader is referred to [Komorowska, 1974], [Harris, 1969], [Oller, 1979], and [Henning, 1987].

## ITEM ANALYSIS

An ideal item is one that is neither too easy nor too difficult for the particular group of candidates, and also has the 'power to discriminate' between good and poor candidates. Both these factors can be analyzed statistically on the basis of the results of a test. Such an analysis is the essential component of pretesting, the function of which is to identify 'good' and 'bad' test items. Good items are then collected and reused or serve as example for new items. Bad items are revised or discarded.

Item difficulty (ID) is simply the proportion of the sum of

all the scores on a given item to the sum of the highest possible scores, subtracted from 1.

ID = 1 - sum / highest possible sum

For example, if the maximum number of points for a test item was 2 and there were 10 testees (sum of highest possible scores

| Testee | Score |
|---|---|
| 1 102 | 2 |
| 2 035 | 1 |
| 3 011 | 2 |
| 4 111 | 0 |
| 5 199 | 0 |
| 6 045 | 1 |
| 7 001 | 2 |
| 8 122 | 1 |
| 9 010 | 0 |
| 10 124 | 0 |
| | (9) |

Fig. 1

= 2 * 10 = 20) whose scores on one hypothetical item were like those in Figure 1, then this item's difficulty index would be calculated as follows:

ID = 1 - 9 / 20 = 1 - .45 = .55

We can thus say that the item had a medium difficulty as the sum of the subjects' scores on this item equalled almost half of the sum of the highest possible scores.

Item difficulty can vary from 0 to 1. An item's ID is equal to 0 if all the testees obtained the maximum number of points for a given item. Such an item is obviously too easy. An item's ID is equal to 1 if all the testees answered the item incorrectly. Such an item is obviously too difficult.

Ideally, then, items of an examination test should have an ID close to .5, whereas any items with an ID close to 1 or 0 should be avoided.

In addition to the difficulty index, it is necessary to be able to test an item's power to discriminate between good and poor candidates. The scores on an individual item should theoretically correlate with the overall score for the whole test. In other words, for an item to be considered well correlated with

the general results of the test, the order of the testees based on their scores on that individual test item should be identical to the order based on their overall total score. It is very rarely identical, but just how close it is to being identical is what we want to be able to measure. When calculating the discriminating power (DP) of an item we do not look at the order of testees, but at the sums of scores obtained by the testees at the top of the list and at those obtained by the testees at the bottom[1]. Consider the group of testees in Figure 3 which has been ordered according to their overall scores and two hypothetical items, 15 and 16 (out of, say, 25).

Intuitively, we would consider item 15 as one that is correlated with the overall score since more points were scored by the testees in the top half of the list (7/4), whereas item 16 as not correlated since more points were scored in the bottom half (3/7).

To calculate a DP index for an item we calculate the sum of the scores obtained in the top section of the list and subtract from it the sum of the scores obtained in the bottom section and divide the result by the sum of the highest possible scores in a section:

| Testee | Total | ... | 15 | 16 | .... |
|---|---|---|---|---|---|
| 1 102 | 54 | ... | 2 | 1 | .... |
| 2 035 | 52 | ... | 1 | 0 | .... |
| 3 011 | 33 | ... | 2 | 1 | .... |
| 4 111 | 30 | ... | 0 | 1 | .... |
| 5 199 | 21 | ... | 2 | 0 | .... |
| | | | (7) | (3) | |
| 6 045 | 20 | ... | 1 | 1 | .... |
| 7 001 | 15 | ... | 2 | 2 | .... |
| 8 122 | 12 | ... | 1 | 2 | .... |
| 9 010 | 8 | ... | 0 | 2 | .... |
| 10 124 | 2 | ... | 0 | 0 | .... |
| | | | (4) | (7) | |

Fig. 2

[1] Various other names have been suggested for this statistical index (see note 3). The choice of the term 'discriminating power' is not indended to be associated with any specific theory or approach.

DP = (top sum - bottom sum) / sum in section

where 'section' denotes 50, 33.3, or 27.5 per cent of the total number of testees depending on the method employed.

In the example above, the DP for item 15 (using the 50% method) is .3 ((7-4)/10 = 3/10), and for item 16 it is -.4 ((3-7)/10 = -4/10).

DP can vary from +1 to -1 as in (a) and (b) in Figure 3. An even distribution of scores produces a DP close to 0 as in (c) (Figure 3).

| | Testee | Total | ...(a) | (b) | (c) | ... |
|---|---|---|---|---|---|---|
| | 1 102 | 54 | ... 2 | 0 | 1 | ... |
| | 2 035 | 52 | ... 2 | 0 | 0 | ... |
| top | 3 011 | 33 | ... 2 | 0 | 2 | ... |
| | 4 111 | 30 | ... 2 | 0 | 1 | ... |
| | 5 199 | 21 | ... 2 | 0 | 1 | ... |
| | | | (10) | (0) | (5) | |
| | 6 045 | 20 | ... 0 | 2 | 1 | ... |
| | 7 001 | 15 | ... 0 | 2 | 1 | ... |
| bottom | 8 122 | 12 | ... 0 | 2 | 0 | ... |
| | 9 010 | 8 | ... 0 | 2 | 2 | ... |
| | 10 124 | 2 | ... 0 | 2 | 1 | ... |
| | | | (0) | (10 | (5) | |

(a) DP = (10-0)/10 = 1

(b) DP = (0-10)/10 = -1

(c) DP = (5-5)/10 = 0

Fig. 3

It is interesting to observe the relationship between ID and DP in view of the possible values the two indices can take. As can be observed, there are in fact three reasons why an item has no power to disciminate between good and poor testees (DP close to 0): when it is too easy (ID close to 0), when it is too difficult (ID close to 1), and when it has an equal -average difficulty for the top and bottom testees (scores evenly distributed in the top and bottom sections). An item has a negative DP

when for some reason the poorer testees (judged as such on the basis of the rest of the items) score better on that item than the 'good' ones. For all the negative values of DP, ID can also vary from close to 0 to close to 1. An item's ID in its middle range (.5), then, does not automatically mean that the item discriminates well -- calculating it by itself is not enough to evaluate the item as it would not detect items with combinations producing a low DP. We can further calculate that acceptable DP indices (between, say, .3 and 1) regularly coincide with ID indices between .2 and .8. It could be argued then that it is not necessary to calculate the ID index at all as long as we know an item's DP. It seems, however, that this extra information is often necessary in the case of items with a DP index between .4 and .8, whose ID index can be either closer to .2 or closer to .8. This might just make a difference when considering selecting one out of several items with an equal or similar DP.

Despite the obvious advantages of such analyses of item indices, calculations like these are rarely performed due to the fact that they are considerably tedious and time-consuming. If done manually (preferably using a calculator), item analysis of a single test involves the following steps:

1. Arrange the tests in order on the basis of the overall scores.

2. Select top and bottom section tests (the number of tests in both sections should be the same).

3. Copy the scores on individual items on a separate larger sheet of paper as shown in Figure 4 below.

4. Calculate the totals for the top and bottom sections for each item as shown in Figure 4 above.

5. Calculate the ID index for each item:

$$ID = 1 - \frac{\text{top} + \text{bottom}}{\text{subjects} * \text{max}}$$

6. Calculate the DP index for each item:

$$DP = \frac{\text{top} - \text{bottom}}{\text{subjects in section} * \text{max}}$$

(where 'max' denotes the maximum number of points possible to be scored on one item; in this case max equals 1).

| name/code | 1 | 2 | 3 | 4 | ...total |
|---|---|---|---|---|---|
| 25 | 0 | 1 | 0 | 1 | ... 44 |
| 36 | 1 | 1 | 0 | 1 | ... 42 |
| 1 | 0 | 1 | 1 | 0 | ... 36 |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| top total | 26 | 28 | 24 | 26 | ... 586 |
| 32 | 0 | 0 | 1 | 1 | ... 14 |
| 1 | 0 | 1 | 0 | 0 | ... 12 |
| 86 | 1 | 0 | 0 | 0 | ... 8 |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| bottom total | 14 | 12 | 20 | 18 | ... 250 |
| overall total | 40 | 40 | 44 | 42 | ... 836 |

Fig. 4

If we have a single test of say 30 items and 50 subjects, it doesn't seem more time-consuming than other small-size statistical tasks. But what if we have a battery of say 5 tests, 20 - 50 items each and between 200 and 300 subjects? And what if we administer such tests fairly often and are constantly on the lookout for 'good' items? And what if at one point the test designers decide to analyze the tests that are administered as a battery the same way the items of a single test are analyzed in order to evaluate the effectiveness of a particular test type against other types? (After all, a whole test that is part of a battery also has its own relative difficulty and discriminating power). This is not to mention comparisons of tests year after year. This is where a computer can and should help.

## Data Input

The first time raw data is entered into the computer, it is typed in using the standard input device, i.e., the keyboard and usually immediately stored on an external medium such as a floppy disk or magnetic tape[2]. The operator should be able to correct any errors in the data as it is being entered and edit (alter or delete parts of) it at any time afterwards. (The rule is that the same raw data should never be keyed in more than once). The software should have the ability to simulate sample raw data for testing or demonstration purposes in addition to reading authentic data from disk or tape files and the keyboard. The format of a raw data file on an external medium should allow checking the data for errors before analysis, transferring it onto another type of computer or medium, or reading it by another program. Raw authentic data should, of course, be kept on an external medium for any yet-to-be scheduled research project.

## Data Output

The software should be able to direct any of its relevant output to the printer and disk or tape files in addition to the monitor screen. A 'hard copy' of the analyses can then be kept in a file or folder within easy reach so that there is no need to access the computer when they are to be consulted. Disk and tape files can be easily merged with other documents using word processing software when, for example, there is a need to present a paper or report involving any of the results of several analyses[3]. (The rule, again, is not to have to retype any of the data).

---

[2] Raw scores could of course be read in by the computer mechanically using special answer sheets and optical input devices. In the case of multiple choice tests, such a system would also perform the actual evaluation of the test.

[3] Most of the figures presented here have in fact been retrieved from disk and printed after slight editing with a word processor.

## The Analyses

Once raw data has been entered into the computer, there is virtually no limit to what a researcher aided by the computer can do with it. In the case of test analysis, however only a handful of tasks are really necessary.

First, the computer calculates some standard measures such as those in Figure 5 and displays them along with the essential 'housekeeping' information.

```
wst 3 86.inf  (unique identification of the test)
general info
number of subjects: 226
number of items * max: 45 * 1 = 45
mean: 18.38
standard deviation: 8.80
variance: 77.49
range: 39
```

Fig. 5

It should be easy for the user to change the format of this output as well as add any other (or remove any of the existing) statistical analyses.

Second, the computer should be able to give some graphic representation of the results of a test -- usually in the form of a distribution histogram like the one in Figure 6. (This histogram for the multiple choice grammar test administered in 1986, for example, clearly shows that the test was too easy).

Third, the computer calculates item analysis indices and presents them in a format close to that in Figure 7.

Again, if the user so desires, it should be possible to choose between the different methods for calculating item discriminating power or use all of then for some time and then settle on just one.

wst 2a 86.dst

```
  0
  5
 10  ..
 15  ..
 20  .......
 25  .....
 30  ....
 35  ........
 40  ........
 45  ............
 50  ........
 55  ................
 60  ..................
 65  ..............................
 70  ................................
 75  ........................
 80  ..........................
 85  ..............
 90  ..............
 95  ...
100  ..
```

Fig. 6

wst 4 86.ana

| item | id | dp |
|---|---|---|
| 1 | .51 | .43 |
| 2 | .48 | .43 |
| 3 | .63 | .44 |
| 4 | .46 | .27 |
| 5 | .61 | .30 |
| 6 | .62 | .36 |
| 7 | .55 | .40 |
| 8 | .54 | .49 |
| 9 | .63 | .41 |
| 10 | .78 | .35 |

Fig. 7

Using this output the test designer refers back to the test and analyzes the individual items in an attempt to find out why they produced the indices they did.

Consider the following examples of authentic items of a vocabulary test -- one with unacceptable indices and one with fairly good ones. Both were used in 1986.

E x a m p l e 1.

24. I'm sure that in spite of a number of things which he failed to achieve, the --e---- evaluation of his term of office must be positive.

(overall)

ID = .98

DP = .01

E x a m p l e 2.

33. The rise in the price of oil doesn't --f--- us very much because we don't own a car.

(affect)

ID = .65

DP = .55

In the case of multiple choice tests, the computer should also present the analysis of what is called 'response frequency distribution', that is the information on the performance of the distractors in the top and bottom sections. The test designer wants to know, for example, which distractors were more popular among the top section testees than among the bottom ones, which distractors were not popular at all, and so on. Figure 8 shows an example of a response frequency distribution table:

Consider the following examples of authentic items of a multiple-choice grammar test.

E x a m p l e 3.

7. Don't ring me up at 9:30 -- ..... a group of Japanese businessmen.

A) I will have met

B) I will be meeting

C) I will meet

D) I would meet

| item: 7 | @ | a | B | c | d |
|---|---|---|---|---|---|
| id: .19 top: | 1 | 2 | 101 | 8 | 1 |
| dp: .18 btm: | 0 | 17 | 80 | 13 | 3 |

wst 2a 86.ana

| item: 1 | @ | a | b | C | d |
|---|---|---|---|---|---|
| id: .69 top: | 1 | 50 | 4 | 50 | 8 |
| dp: .28 btm: | 1 | 57 | 25 | 18 | 12 |

| item: 2 | @ | A | b | c | d |
|---|---|---|---|---|---|
| id: .19 top: | 0 | 108 | 0 | 5 | 0 |
| dp: .29 btm: | 1 | 75 | 9 | 20 | 8 |

.
.
.

| item: 19 | @ | A | b | c | d |
|---|---|---|---|---|---|
| id: .22 top: | 0 | 104 | 2 | 4 | 3 |
| dp: .28 btm: | 2 | 72 | 22 | 8 | 9 |

| item: 20 | @ | a | B | c | d |
|---|---|---|---|---|---|
| id: .28 top: | 0 | 6 | 102 | 4 | 1 |
| dp: .38 btm: | 2 | 30 | 59 | 11 | 11 |

(@ - no response, right choice indicated in upper case)

Fig. 8

E x a m p l e 4.

15. Without ..... what to begin with they won't be able to move on.

A) being shown

B) showing them

C) to be shown

D) showing

| item: 15 | @ | A | b | c | d |
|---|---|---|---|---|---|
| id: .42 top: | 0 | 88 | 22 | 0 | 3 |
| dp: .41 btm: | 2 | 41 | 51 | 2 | 17 |

Fourth, a plain list of results for a single test like the one in Figure 9 might be desirable as well.

```
wst 3 86.lst

###   name/code          score %%%

001   337                39  086
002   255                36  080
003   231                36  080
004   091                36  080
005   110                35  077
006   041                35  077
007   202                35  077
008   358                34  075
009   318                34  075
010   351                33  073

 .
 .
 .

219   227                01  002
220   350                01  002
221   245                01  002
222   246                00  000
223   029                00  000
224   316                00  000
225   031                00  000
226   229                00  000
```

Fig. 9

```
wst 1-4 86.inf

general info

number of subjects:  226

number of items * max:  5 * 100 = 500

mean:  220.98

standard deviation:  79.39

variance:  6302.24

range:  395
```

Fig. 10: General statistics

Once all the results of individual tests which have been administered to the same group of subjects as a battery have been processed in the manner shown above, it would seem natural to go one step further and have the computer analyze the whole battery as if it were a single test. (We have already entered all the data it needs -- no extra typing is necessary). That way, in addition to obtaining a general distribution histogram and calculating the standard statistics, we can analyze the difficulty of particular tests as well as their discriminating power.

```
wst 1-4 86.dst

  0  .
  5  ...
 10  .......
 15  ......
 20  ................
 25  ..............
 30  ..............
 35  ..................
 40  ..........................
 45  ..............................
 50  ..........................
 55  ..........................
 60  ......................
 65  ........
 70  ....
 75  ..
 80  .
 85
 90
 95
100
```

Fig. 11: Distribution histogram

```
wst 1-4 86.ana

item  id    dp
----------------
  1  .75   .19      (listening comprehension:
                     open-ended questions)
  2  .36   .23      (grammar: multiple choice choice)
  3  .48   .31      (grammar: guided translation)
  4  .59   .27      (vocabulary: gap-filling)
  5  .58   .26      (controlled writing)
```

Fig. 12: Item difficulty and discriminating power indices

The analyses performed on the results of the 1986 battery of five entrance exam tests are shown in Figures 10 through 12.

In addition to the above statistical tasks the software should assist in some of the secretarial jobs involved in the organization of an exam. For example, it should allow the user to substitute the subjects' names for the code numbers and print two lists of final results: one sorted alphabetically and one by final score (again with minimum typing).

## FLEXIBILITY

The essential characteristic of test analysis software (of any statistical software, for that matter) should be its flexibility. Flexibility in this context is understood as the ability of the software to handle variable data without interfering with any of the test designer's work. The designer should not, for example, be forced to abandon a particular testing technique just because the software will be unable to anylyze the results. Specifically, the test designer's freedom should be guaranteed at least in the following areas:

- number of testees
- number of test items
- maximum possible score on an item
- number of tests in a battery
- number of options in a multiple choice test (2 (true / false) to 5, for example)

## CHOICE OF COMPUTER SYSTEM

Test analysis does not require a sophisticated high performance system such as, say, an IBM PC/AT class microcomputer. A system like that would of course speed up all the operations and generally make things easier, especially if there were a lot of the 'secretarial' jobs to be done. In general, however, even a more modest system is enough, as long as it is equipped with a floppy disk drive and a dot-matrix printer, and future data transfer to a larger system will be possible.

## CHOICE OF SOFTWARE

The general purpose statistical packages -- the first natural choice -- seem to be least suitable for our purpose mainly because they lack the flexibility we want. General purpose software packages usually have only some of the functions that are necessary. Besides, they cannot do secretarial tasks and usually require retyping of data[4].

The second choice might be one of the commercially available general purpose data base management and electronic spreadsheet packages. This is especially true of those that have some application programming capability. Systems designed in their environment, however, would require a fair amount of programming if they were to be used by non-specialists. What is more, since they are generally much slower than dedicated software written in a compiled language like Pascal or C, they would have to be run on a faster machine[5].

The best solution as far as software is concerned (as in the case of other specialized applications) is for the test designer to lay out the requirements for the software, design the package with the help of a computer programmer, and then let the programmer write the program(s). The software could then be tested and customized according to the test designer's specifications, and with the appropriate documentation such a system would be easy to maintain and adjust to the particular needs and conditions. Test designers should not, however, attempt to write the programs themselves unless they also happen to be experienced programmers. For an amateur programmer (even one who otherwise is a competent computer user) some of the problems in developing software like this -- such as disk input/output operations, sorting and merging, string manipulation, compacting the data and maintaining its integrity, and compactness of code, to name a few -- might just be too hard to handle.

---

[4] This does not have to be so as I have never actually tested this option.

[5] One package that would probably handle most of the tasks is the dBASE III PLUS data base management system available for IBM PC and compatibles.