hence $\mathbf{p}_i$) is determined by

$$\mathbf{s}_i = 0 \quad \text{if} \quad f(\mathbf{p}_{i-1} + \beta_1\mathbf{h}_{i-1} + \beta_2\mathbf{r}_i) \leqq f(\mathbf{p}_{i-1})$$

$$\mathbf{s}_i = \beta_1\mathbf{h}_{i-1} + \beta_2\mathbf{r}_i \quad \text{if} \quad f(\mathbf{p}_{i-1} + \beta_1\mathbf{h}_{i-1} + \beta_2\mathbf{r}_i) > f(\mathbf{p}_{i-1}),$$

where β_1 and β_2 are scalars.

Since history may sometimes be wrong, it is important to selectively ignore it. This is conveniently accomplished by dynamic adjustments of β_1, β_2, γ_1, and γ_2. For example, whenever $f(\mathbf{p}_{i-1} + \mathbf{s}_i) \leqq f(\mathbf{p}_{i-1})$, β_1/β_2 could be tentatively increased and/or decreased to decide empirically whether more or less history is desirable. The particular algorithm described here, however, has $\beta_1 = \beta_2 = 0.5$ so that history is used in generating *every* step. γ_1 and γ_2 are dynamically adjusted so that $\gamma_1 = \gamma_2$ and $|\mathbf{h}_i| = 1$.

Algorithm 3. This algorithm coincides with algorithm 1 until N interrogations of the black box have been made. Thereafter, a model of f is derived (by a second optimizer) from the values of f that are already known. A third optimizer optimizes this model, thereby generating a position vector which is used to bias the random vectors that are used by the optimizer of f. In the simple version examined here, N is taken to be four. Each model of f is a least-squares plane fitted to the four most recently obtained values of f. This model is simple enough that the fitting was done analytically rather than by an optimizer. Assuming that this linear model is a reasonable local approximation to f, algorithm 1 is used to climb five steps on the model, using the most recent position of the f optimizer as a starting point. The resulting position vector, $\mathbf{v}_i$, is used (in this example, without modification) by the f optimizer for its next tentative position vector, i.e. $\mathbf{v}_i - \mathbf{p}_{i-1}$ is used instead of $\mathbf{r}_i$ in the formulae (of algorithm 1) that generate $\mathbf{p}_i$.

The results of applying these algorithms are shown as optimization paths in figure 5. The number of evaluations used to reach a point is given beside that point. Because the points are generated randomly, this sample is insufficient to decide—even for this problem—which of the three algorithms is superior.

REFERENCES

1. Ashby, W. R. Design for an intelligence amplifier. *Automata Studies*, Annals of Mathematics Studies No. 34, Princeton University Press, Princeton, N. J. (1956).
2. Newell, A., Shaw, J. C., and Simon, H. A. Chess playing programs and the problem of complexity. *IBM J. Research 2*, (1958), 320–335.
3. Box, G. E. P. and Wilson, K. B. On the experimental attainment of optimum conditions. *J. Roy. Stat. Soc.* {B}, *13*, (1951), 1–45.
4. Curry, H. B. The method of steepest descent for non-linear minimization problems. *Quart. Appl. Math. 2*, (1944), 258–261.
5. Forsythe, A. I. and Forsythe, G. E. Punched-card experiments with accelerated gradient methods for linear equations. NBS Appl. Math. Series 39, pp. 55–69 (1954).
6. Johnson, S. M. Best exploration for maximum is Fibonaccian. U.S. Air Force Project RAND Memorandum RM-1500, Nov. 18, 1955.
7. Wold, H. *Random Normal Deviates*. Tracts for Computers No. XXV, Cambridge University Press (1954).

Standards

Survey of Punched Card Codes

H. J. Smith and F. A. Williams

IBM Corporation, White Plains, New York

The "Survey of Coded Character Representation" by R. W. Bemer (p. 639) covers the representation of symbols in paper tape, magnetic tape and main storage of a number of different machine systems. Another method of mechanical storage is the punched card. The results of a similar survey for punched card codes are shown in the table on page 642.

Two obvious omissions are the six-row card code of Remington Rand and the Bull code. These codes were not included because the coding techniques are entirely different from that of those shown.

The card code is identified from top to bottom by numbering the rows 12, 11, 0, 1, . . . , 9. The graphic equivalence shown is that given by printing devices in the systems. Some of the codes have a control function as well as graphic significance.

This chart is also presented as staff work for the deliberations of Sub-committee X3.2 of the American Standards Association. It is the most complete information presently available but obviously may contain errors and omissions. Any corrections or additions will be gladly received.

* * *

Notes (to accompany table on page 642):

IBM Type Arrangements: These arrangements are the ones available upon IBM printing equipment. The F arrangement is Share 704 Fortran. The H arrangement is Share 709 Fortran.

M-H: This is the representation of the card code upon these devices when fed through the main frame.

Philco 2000: The graphics n and e print only in Memory Dump Mode.

G-15/CA-2: The manual states "all special characters."

BTM: Uses 0, 1, 6, 8 to represent O, I, G, and S.