then the coefficients are estimated as

$$\begin{aligned} a_0 &= y_0 \\ a_i &= 2y_{0i} - (y_i + 3y_0)/2, \qquad i = 1, \ldots, n \\ b_{ii} &= 2(y_i + y_0 - 2y_{0i}), \qquad i = 1, \ldots, n \\ b_{ij} &= 2(y_{ij} + y_0 - y_{0i} - y_{0j}), \quad i \neq j, \end{aligned}$$

where y_i is the function value at P_i and y_{ij} that at P_{ij}. The estimated minimum is then given by

$$\boldsymbol{x}_{min} = -\boldsymbol{B}^{-1}\boldsymbol{a}$$

and the information matrix is just $\boldsymbol{B}$.

If $\boldsymbol{p}_i$ denotes the co-ordinates of P_i in the original system, and if $\boldsymbol{Q}$ is the $n \times n$ matrix whose ith column is $\boldsymbol{p}_i - \boldsymbol{p}_0$, then the minimum is estimated to be at

$$\begin{aligned} \boldsymbol{p}_{min} &= \boldsymbol{p}_0 + \boldsymbol{Q}\boldsymbol{x}_{min} \\ &= \boldsymbol{p}_0 - \boldsymbol{Q}\boldsymbol{B}^{-1}\boldsymbol{a}. \end{aligned}$$

The minimum value of the function is estimated to be

$$y_{min} = a_0 - \boldsymbol{a}'\boldsymbol{B}^{-1}\boldsymbol{a}.$$

The information matrix in the original co-ordinate system is given by

$$(\boldsymbol{Q}^{-1})'\boldsymbol{B}\boldsymbol{Q}^{-1}$$

so that the variance–covariance matrix is given by

$$\boldsymbol{Q}\boldsymbol{B}^{-1}\boldsymbol{Q}'.$$

If normal equal-variance independent errors are involved and the sum of squares of residuals is minimized, then this matrix must be multiplied by $2\sigma^2$, where as usual σ^2 would be estimated by $y_{min}/(N - n)$, N being the total number of observations, and n the number of parameters fitted.

In estimating $\boldsymbol{B}$ numerically it is necessary to steer a course between two hazards. In one the simplex is so small that $(y_{ij} + y_0 - y_{0i} - y_{0j})$ consists largely of rounding-off errors incurred in calculating the y's. In the other the simplex is so large that the quadratic approximation is poor, and the b's are correspondingly biased. If the method given in this paper is used, the former hazard will usually be the important one, and it may be necessary to enlarge the final simplex before adding the extra points. A possible way of doing this is to double the distance of each point P_i from the centroid until the corresponding function value exceeds that at the centroid by more than a given constant. The choice of this would depend on the rounding-off error attaching to the evaluation of the function, and would need to be at least 10^3 times that rounding error, if acceptable estimates of the b's were to be obtained.

References

FLETCHER, R., and POWELL, M. J. D. (1963). "A rapidly convergent descent method for minimization," *The Computer Journal*, Vol. 6, p. 163.

POWELL, M. J. D. (1962). "An iterative method for finding stationary value of a function of several variables," *The Computer Journal*, Vol. 5, p. 147.

POWELL, M. J. D. (1964). "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *The Computer Journal*, Vol. 7, p. 155.

ROSENBROCK, H. (1960). "An automatic method for finding the greatest or least value of a function," *The Computer Journal*, Vol. 3, p. 175.

SPENDLEY, W., HEXT, G. R., and HIMSWORTH, F. R. (1962). "Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation," *Technometrics*, Vol. 4, p. 441.

Correspondence

To the Editor,
The Computer Journal.

An impossible program

Sir,

A well-known piece of folk-lore among programmers holds that it is impossible to write a program which can examine any other program and tell, in every case, if it will terminate or get into a closed loop when it is run. I have never actually seen a proof of this in print, and though Alan Turing once gave me a verbal proof (in a railway carriage on the way to a Conference at the NPL in 1953), I unfortunately and promptly forgot the details. This left me with an uneasy feeling that the proof must be long or complicated, but in fact it is so short and simple that it may be of interest to casual readers. The version below uses CPL, but not in any essential way.

Suppose $T[R]$ is a Boolean function taking a routine (or program) R with no formal or free variables as its argument and that for all R, $T[R]$ = **True** if R terminates if run and that $T[R]$ = **False** if R does not terminate. Consider the routine P defined as follows

```
rec routine P
    § L : if T[P] go to L
    Return §
```

If $T[P]$ = **True** the routine P will loop, and it will only terminate if $T[P]$ = **False**. In each case $T[P]$ has exactly the wrong value, and this contradiction shows that the function T cannot exist.

Yours faithfully,

Churchill College,
Cambridge.

C. STRACHEY.

Downloaded from http://comjnl.oxfordjournals.org/ at UniversitÃ© Libre de Bruxelles - BibliothÃ¨ques - CP 180 on December 11, 2013