The Secant Method for Simultaneous Nonlinear Equations

PHILIP WOLFE, *The RAND Corporation, Santa Monica, California*

Abstract. A procedure for the simultaneous solution of a system of not-necessarily-linear equations, a generalization of the secant method for a single function of one variable, is given.

1. Secant Method for n Equations

This note is concerned with a computational procedure for the solution of the simultaneous equations

$$f_i(x) = 0, \qquad i = 1, \cdots, n, \tag{1}$$

where $x = (x_1, \cdots, x_n) \in E_n$ and each f_i is a computable function of x. (For the later theoretical discussion of the procedure, the f_i will be supposed to have second derivatives, but no derivatives are calculated.)

The procedure consists in the iteration of the following step, at the beginning of which it is assumed that the $n + 1$ "trial solutions" $x^1, \cdots, x^{n+1}$ are at hand.

Find $\pi_1, \cdots, \pi_{n+1}$ such that

$$\sum_{j=1}^{n+1} \pi_j = 1 \tag{2}$$

and

$$\sum_{j=1}^{n+1} \pi_j f_i(x^j) = 0 \quad \text{for } i = 1, \cdots, n; \tag{3}$$

let

$$\bar{x} = \sum_{j=1}^{n+1} \pi_j x^j, \tag{4}$$

and obtain a new set of trial solutions by replacing, with $\bar{x}$, some x^j for which $\| x^j \|$ is maximal, where

$$\| x \| = \sum_{i=1}^{n} | f_i(x) |^2. \tag{5}$$

Note that for the case $n = 1$ the above is just the ordinary secant method for finding the root of a single function f by finding that point on the line through two trial points $(x^1, f(x^1))$, $(x^2, f(x^2))$ on the graph of f which lies on the x-axis. Note also that in case the f_i are all linear, $\bar{x}$ is immediately the solution. Finally, note that the choice of norm $\| x \|$ in (5) was pretty arbitrary; it is not known whether other norms would be more effective.

2. Convergence of the Method

In the formulas below, let z be the solution of the system (1); let G_i be the vector of first partial derivatives[1] of f_i with respect to $x_1, \cdots, x_n$; let $2Q_i$ be the matrix of second partial derivatives[1] of f_i; and let $\cong$ denote equality to within terms of second order of the quantities $(x^j - z)$. Using (2) liberally,

$$\begin{aligned} f_i(\bar{x}) &\cong G_i(\bar{x} - z) + (\bar{x} - z)Q_i(\bar{x} - z) \\ &= \textstyle\sum_j \pi_j G_i(x^j - z) + (\bar{x} - z)Q_i(\bar{x} - z) \\ &\cong \textstyle\sum_j \pi_j [f_i(x^j) - (x^j - z)Q_i(x^j - z)] \\ &\qquad\qquad + (\bar{x} - z)Q_i(\bar{x} - z) \\ &= \textstyle\sum_j \pi_j f_i(x^j) - \sum_j \pi_j x^j Q_i x^j + \bar{x}Q_i\bar{x} \\ &= -\textstyle\sum_j \pi_j (x^j - \bar{x})Q_i(x^j - \bar{x}) \qquad \text{(using (3))}. \end{aligned}$$

Thus $\| \bar{x} \|$ is of second order in the quantities $x^j - \bar{x}$. This unfortunately does not yield a quadratic type of convergence, since the behavior of the π_j is not known; if, however, the solutions of (2), (3) were to remain bounded throughout the process, convergence of an order higher than one could be shown.

3. Computational Scheme

The bulk of computational labor in this process is the solution of the equations (2), (3) at each iteration. Since the equations are only partly altered in each step, the necessary work is less than that done in solving $n + 1$ equations from scratch.

Given the $n + 1$ trial solutions, form the $n + 1$ columns

$$A_j = \begin{bmatrix} f_1(x^j) \\ \vdots \\ f_n(x^j) \\ 1 \end{bmatrix}, \quad j = 1, \cdots, n + 1$$

of the $(n + 1)$-order matrix A of the coefficients of the linear equations. Initially, the inverse of A is formed from these data; in general, suppose that A^{-1} is at hand. The symbol $'$ will denote transpositon (row vector $\rightarrow$ column vector) below, and $\pi = (\pi_1, \cdots, \pi_{n+1})'$. The equations (2), (3) may be written

$$A\pi = (0, 0, \cdots, 0, 1)',$$

so that

$$\pi = A^{-1}(0, \cdots, 0, 1)',$$

whence π is the right-hand column of A^{-1}.

Next form $\bar{x}$ from (4), and the vector

$$p = (f_1(\bar{x}), \cdots, f_n(\bar{x}), 1)',$$

the new column of coefficients for the next matrix $A*$,

[1] At $x = z$.

which is to replace the jth column of A, where x^j is the trial solution dropped in this iteration. The new inverse is calculated from the old by pivoting:

Calculate $q = A*^{-1}p$. Then

$$(A*^{-1})_{jk} = (A^{-1})_{jk}/q_j \qquad \text{for all } k,$$

and

$$(A*^{-1})_{lk} = (A^{-1})_{lk} - (A^{-1})_{jk} \cdot (q_l/q_k) \qquad \begin{cases} \text{for } \neq j, \\ \text{all } k. \end{cases}$$

4. Computational Experience

A FORTRAN II program has been written for trying out this procedure. Its input consists of n, a set of trial solutions or a signal that such a set should be generated, and programs which calculate the f_i. A variety of problems with $n = 2$ have been solved. The process has converged for these in a manner like that which Jeeves [1] has shown for the case $n = 1$, namely that the error at a given step is proportional to the product of the errors at the two previous steps—convergence of order $\frac{1}{2}(\sqrt{5} + 1)$.

SAMPLE: $n = 2$, $f_1(x, y) = x^2 + x - y^2 + 1$, $f_2(x, y) = y(1 - 2x)$ (the real and imaginary parts of $z^2 + z + 1$).

		Points x	*Points* y	*Norm*
Initial	1	−0.600000	1.100000	0.370000
	2	−0.300000	1.100000	1.518400
	3	−0.600000	1.400000	0.250900
	4	−0.516058	0.923358	0.011351
	5	−0.503347	0.870741	0.000101
	6	−0.500884	0.866819	0.423×10^{-5}
	7	−0.499988	0.865996	0.306×10^{-8}
	8	−0.500000	0.866025	0.106×10^{-12}

REFERENCE

[1] T. A. JEEVES, Secant modification of Newton's method, *Comm. Assoc. Comp. Mach. 1*, No. 8 (1958), 9–10.

Automatic Programming Systems

The following are additions to the ACM Library. For the previous status, please refer to the May 1959 issue of *Communications*, page 16.

Computer	*In ACM Library*	*Do Not Have*
709	Commercial Translator FORC 2 FORTRAN	S.O.S. 9 PAC
704		SURGE
705 III	Commercial Translator	
705	Autocoder III FORTRAN	
650	TASS	
7070	Autocoder Commercial Translator FORTRAN	
1103		SLAP
1105	AIMACO	
DATATRON 205, 220		FORTRAN
G-15	POGO	
TRANSAC		TAC ALTAC (FORTRAN)
H-800	ARGUS HBC	FORTRAN
NCR 304	NEAT STEP	
RCA 501	Automatic Assembly	
ICT 1400	CODEL	