ALGORITHM 283
SIMULTANEOUS DISPLACEMENT OF POLYNOMIAL ROOTS IF REAL AND SIMPLE [C2]

IMMO O. KERNER (Recd. 8 Sept. 1965 and 12 Nov. 1965)
Rechenzentrum Universitaet Rostock

```
procedure Prrs (A, X, n, eps); value n, eps;
  integer n; real eps; array A, X;
comment Prrs (polynomial roots real simple) computes the n
  roots X of the polynomial equation

        A_n x^n + A_{n-1} x^{n-1} + ··· + A_0 = 0

  simultaneously. On entry the array X contains the vector of
  initial approximations to the roots and on exit it contains the
  vector of improved approximations to the roots. The initial
  approximations must be distinct. Accuracy is specified by means
  of a parameter eps. Iteration is continued until the Euclidean
  norm of the correction vector does not exceed eps. The con-
  vergence is quadratic;
begin integer i, k; real x, P, Q;
      eps := eps ↑ 2;
W:  Q := 0;
      for i := 1 step 1 until n do
  begin  x := P := A[n];
    for k := 1 step 1 until n do
    begin x := x × X[i] + A[n − k];
      if k ≠ i then P := P × (X[i]−X[k])
    end;
    X[i] := X[i]−x/P;
    Q := Q + (x/P) ↑ 2
  end;
  if Q > eps then go to W
end
```

CERTIFICATION OF ALGORITHM 9 [D2]
RUNGE-KUTTA INTEGRATION [P. Naur et al., *Comm. ACM 3* (May 1960), 318]

HENRY C. THACHER, JR. (Recd. 28 July 1964 and 22 Nov. 1965)
Argonne National Laboratory, Argonne, Ill.

Algorithm 9 was transcribed into the hardware representation for CDC 3600 ALGOL and run successfully. The following procedure was used for the global procedure *comp*:

```
real procedure comp (a, b, c); value a, b, c; real a, b, c;
begin integer AE, BE, CE;
  integer procedure expon(x); real x;
  comment This function produces the base 10 exponent of x;
  expon := if x = 0 then −999 else
    entier (.4342944819 × ln(abs(x)) + 1);
  comment The number −999 may be replaced by any number
    less than the exponent of the smallest positive number handled
    by the particular machine used, for this algorithm assumes
    that true zero has an exponent smaller than any nonzero
    floating-point number. Users implementing real procedure
    comp by machine code should make sure that this condition
    is satisfied by their program;
  AE := expon(a); BE := expon(b); CE := expon(c);
  if AE < BE then AE := BE; if AE < CE then AE := CE;
  comp := abs(a − b)/10 ↑ AE
end
```

This has the advantage of machine independence, but is highly inefficient compared to machine code.

The procedure was tested using the two following procedures for *FKT*:

```
procedure FKT (X, Y, N, Z); real X; integer N; array
  Y, Z;
comment (dy1/dx) = z1 = y2 , (dy2/dx) = z2 = −y1 . With
  y1(0) = 0, y2(0) = 1, the solution is y1 = sin x, y2 = cos x;
begin Z [1] := Y [2]; Z [2] := − Y [1] end;
procedure FKT (X, Y, N, Z); real X; integer N; array
  Y, Z;
comment (dy1/dx) = 1 + y1^2. For y1(0) = 0, y(x) = tan x;
  Z [1] := 1+Y[1]↑2;
```

The *RK* procedure was used to integrate the differential equations represented by the first *FKT* procedure from $x = 0(0.5)7.0$, with $eps = eta = 10^{-6}$, and with $y_1(0) = 0$, $y_2(0) = 1$. The actual step size h was .0625 for most of the range, but was reduced to .03125 in the neighborhood of $x = k\pi/2$, where one or the other of the solutions is small.

The computed solutions at $x = 7.0$ were: $y_1 = 6.5698602746 \times 10^{-1}$, $y_2 = 7.5390270246 \times 10^{-1}$, with errors -5.71×10^{-7} and 4.48×10^{-7}, respectively.

Results for the second differential equation are summarized in Table I below.

The efficiency of the procedure would be increased slightly on most computers by changing the type of the **own** variable s from **real** to **integer.**

The error is estimated by comparing the results of successive pairs of steps with that of a single double step. This is somewhat more time-consuming than the Kutta-Merson process presented in Algorithm 218 [*Comm. ACM 6* (Dec. 1963) 737–8]. However, the criterion for step-size variation in Algorithm 9 which effectively applies an approximate relative error criterion, *eps*, for $|y| > eta$, and an absolute error criterion $eta \times eps$, for $|y| < eta$, appears superior when the solution fluctuates in magnitude.

REMARK ON ALGORITHM 218 [D2]
KUTTA-MERSON [Phyllis M. Lukehart, *Comm. ACM 6* (Dec. 1963), 737]

G. BAYER (Recd. 25 Oct. 1965)
Technische Hochschule, Braunschweig, Germany

Successive calls of *Kutta Merson* with *first* ≡ **false** do not reach the upper bound $t+h$ if the interval h is unequal to the interval h of the first call with *first* ≡ **true**.

Proposed correction:

```
1) declaration real hc,  instead of own real hc;
2) if first then begin for i := 1 step 1 until n do y0[i] := y[i];
                 hc := h;  ploc := 1;  first := false
              end else hc := h/ploc;
   instead of if first then begin ··· end;
```

TABLE I [ALG. 9]

	η	$x = 0.5$ h_{min}	Absolute error	Relative error	$x = 1.0$ h_{min}	Absolute error	Relative error	$x = 1.5$ h_{min}	Absolute error	Relative error
10^{-7}	10^{-3}	.03125	-1×10^{-9}	-2×10^{-9}	.03125	9×10^{-8}	6×10^{-8}	.00390625	-1×10^{-6}	-8×10^{-8}
10^{-5}	10^{-3}	.125	-5×10^{-7}	-9×10^{-7}	.0625	8×10^{-7}	5×10^{-7}	.0078125	-2×10^{-4}	-1×10^{-5}
10^{-3}	10^{-3}	.25	-1×10^{-5}	-2×10^{-5}	.25	-2×10^{-4}	-1×10^{-4}	.03125	-3×10^{-2}	-2×10^{-3}