ALGORITHM 80
RECIPROCAL GAMMA FUNCTION OF REAL ARGUMENT

WILLIAM HOLSTEN
University of California at San Diego, La Jolla, California


```
real procedure RGR(x); real x; real procedure RGAM;
comment Procedure RGAM computes the real reciprocal
  Gamma function of real x for -1 < x < 1, utilizing Horner's
  method for polynomial evaluation of the approximation poly-
  nomial. RGR extends the range of RGAM by use of the formulae
  (1) 1/Gamma(x-1)=(x-1)/Gamma(x) for x<-1,
  (2) 1/Gamma(x+1)= 1/x×Gamma(x) for x<1.;
  begin real y;
        if x = 0 then begin RGR := 0; go to EXIT end
        if x = 1 then begin RGR := 1; go to EXIT end
        if x < 1 then go to BB;
        y := 1;
AA:     x := x - 1; y := y × x; if x > 1 then go to AA;
        if x = 1 then begin RGR := 1/y; go to EXIT end
        RGR := RGAM(x)/y; go to EXIT;
BB:     if x = -1 then begin RGR := 0; go to EXIT end
        if x > -1 then begin RGR := RGAM(x);
          go to EXIT end
        y := x;
CC:     x := x + 1; if x < -1 then begin y := y × x;
          go to CC end
        RGR := RGAM(x) × y;
EXIT: end RGR;
real procedure RGAM(x); real x; integer i;
  real array B[0:13];
comment The algorithm for this routine was adapted from
  "University of Illinois Digital Computer, Auxiliary Library
  Routine B-17-328", by John Ehrman. Reference may also be
  made to Algorithm 34, dated February, 1961. Approximation
  accuracy is ±2^-35.;
begin real z;
  B[ 0] :=  1.00000 00000 00; B[ 1] := -.42278 43350 92;
  B[ 2] := -.23309 37363 65; B[ 3] := +.19109 11011 62;
  B[ 4] := -.02455 24908 87; B[ 5] := -.01764 52421 18;
  B[ 6] := +.00802 32781 13; B[ 7] := -.00080 43413 35;
  B[ 8] := -.00036 08514 96; B[ 9] := +.00014 56243 24;
  B[10] := -.00001 75279 17; B[11] := -.00000 26257.21;
  B[12] := +.00000 13285 54; B[13] := -.00000 01812 20;
  z: = B[13];
  for i := 12 step -1 until 0 do z := z × x + B[i];
  RGAM := z × x × (x + 1)
end RGAM;
```

REMARKS ON:
ALGORITHM 34 [S14]
GAMMA FUNCTION
[M. F. Lipp, *Comm. ACM 4* (Feb. 1961), 106]
ALGORITHM 54 [S14]
GAMMA FUNCTION FOR RANGE 1 TO 2
[John R. Herndon, *Comm. ACM 4* (Apr. 1961), 180]
ALGORITHM 80 [S14]
RECIPROCAL GAMMA FUNCTION OF REAL ARGUMENT
[William Holsten, *Comm. ACM 5* (Mar. 1962), 166]
ALGORITHM 221 [S14]
GAMMA FUNCTION
[Walter Gautschi, *Comm. ACM 7* (Mar. 1964), 143]
ALGORITHM 291 [S14]
LOGARITHM OF GAMMA FUNCTION
[M. C. Pike and I. D. Hill, *Comm. ACM 9* (Sept. 1966), 684]

M. C. PIKE AND I. D. HILL (Recd. 12 Jan. 1966)
Medical Research Council's Statistical Research Unit,
University College Hospital Medical School,
London, England

Algorithms 34 and 54 both use the same Hastings approximation, accurate to about 7 decimal places. Of these two, Algorithm 54 is to be preferred on grounds of speed.

Algorithm 80 has the following errors:
(1) *RGAM* should be in the parameter list of *RGR*.
(2) The lines
if $x = 0$ **then begin** $RGR := 0$; **go to** *EXIT* **end**
and
if $x = 1$ **then begin** $RGR := 1$; **go to** *EXIT* **end**
should each be followed either by a semicolon or preferably by an **else.**
(3) The lines
if $x = 1$ **then begin** $RGR := 1/y$; **go to** *EXIT* **end**
and
if $x < -1$ **then begin** $y := y \times x$; **go to** *CC* **end**
should each be followed by a semicolon.
(4) The lines
BB: **if** $x = -1$ **then begin** $RGR := 0$; **go to** *EXIT* **end**
and
if $x > -1$ **then begin** $RGR := RGAM(x)$; **go to** *EXIT* **end**
should be separated either by **else** or by a semicolon and this second line needs terminating with a semicolon.
(5) The declarations of **integer** i and **real array** $B[0:13]$ in *RGAM* are in the wrong place; they should come immediately after
begin real z;

With these modifications (and the replaçement of the array B in *RGAM* by the obvious nested multiplication) Algorithm 80 ran successfully on the ICT Atlas computer with the ICT Atlas ALGOL compiler and gave answers correct to 10 significant digits.

Algorithms 80, 221 and 291 all work to an accuracy of about 10 decimal places and to evaluate the gamma function it is therefore on grounds of speed that a choice should be made between them. Algorithms 80 and 221 take virtually the same amount of computing time, being twice as fast as 291 at $x = 1$, but this advantage decreases steadily with increasing x so that at $x = 7$ the speeds are about equal and then from this point on 291 is faster—taking only about a third of the time at $x = 25$ and about a tenth of the time at $x = 78$. These timings include taking the exponential of *log-*

gamma.

For many applications a ratio of gamma functions is required (e.g. binomial coefficients, incomplete beta function ratio) and the use of algorithm 291 allows such a ratio to be calculated for much larger arguments without overflow difficulties.