```
      J1 = I1
      J2 = I2
      IOR = 0
      M = 1
   10 L1 = J1 / 2
      L2 = J2 / 2
      IF(J1.NE.L1 * 2.OR.J2.NE.L2 * 2) IOR = IOR + M
      IF(L1.EQ.0.AND.L2.EQ.0) RETURN
      J1 = L1
      J2 = L2
      M = M + M
      GOTO 10
      END
```

```
      FUNCTION ICNT(I)
C
C        ALGORITHM AS 65.3  APPL.STATIST. (1973), VOL.22, NO.3
C
C        RESULT IS SUM OF DIGITS IN NON-NEGATIVE BINARY INTEGER I. THIS
C        FUNCTION CAN BE CODED MUCH MORE EFFICIENTLY IN ASSEMBLY CODE.
C
      J = I
      ICNT = 0
   10 L = J / 2
      IF(J.NE.L * 2) ICNT = ICNT + 1
      IF(L.EQ.0) RETURN
      J = L
      GOTO 10
      END
```

Algorithm AS 66

The Normal Integral

By I. D. Hill

Medical Research Council, London

Keywords: NORMAL CURVE; TAIL AREA

Language

ISO Fortran

Description and Purpose

Calculates the upper, or lower, tail area of the standardized normal curve corresponding to any given argument.

Numerical Method

The method is that of Adams (1969), but incorporated in a different surrounding structure.

Structure

FUNCTION ALNORM(X, UPPER)

Formal parameters

X	Real	input: the argument value
UPPER	Logical	input: if *.TRUE.* the area found is from *X* to infinity, if *.FALSE.* from minus infinity to *X*

Data constants

The constant *LTONE* should be set to the value at which the lower tail area becomes 1·0 to the accuracy of the machine. $LTONE = (n+9)/3$ gives the required value accurately enough, for a machine that produces n decimal digits in its real numbers.

The constant *UTZERO* should be set to the value at which the upper tail area becomes 0·0 to the accuracy of the machine. This may be taken as the value such that $\exp\{-\frac{1}{2}UTZERO^2\}/(UTZERO\sqrt{2\pi})$ is just greater than the smallest allowable real number.

Related Algorithms

Algorithm AS 2 (Cooper, 1968) exists for the same purpose, but, as Hitchin (1973) has shown, is not always an easy algorithm to use in that both its input and output parameters have to be in array form.

It is believed that the scalar arguments, and functional form, of the current algorithm are much simpler to use.

For the tests mentioned below, AS 2 was modified by adding three extra values to the *CONNOR* array, as suggested in Hill (1969).

Storage

Tests on the Xerox 9300 computer indicate that this algorithm takes only about half as much computer store as does AS 2.

Time

The time taken to find an upper tail area was compared (also on the Xerox 9300) with the following results:

Value of X	*Time in msec* AS 2	AS 66	*Value of X*	*Time in msec* AS 2	AS 66
−20·0	4·5	0·3	0·5	2·9	0·6
−10·0	4·9	0·3	1·0	3·0	0·5
−8·0	4·9	0·2	1·5	3·0	1·0
−6·0	5·6	1·1	2·0	3·0	1·0
−4·0	8·3	1·1	2·5	14·5	1·0
−3·0	12·4	1·1	3·0	12·3	1·0
−2·5	14·5	1·1	4·0	8·3	1·1
−2·0	2·9	1·0	6·0	5·5	1·1
−1·5	3·0	1·1	8·0	4·9	1·1
−1·0	3·0	0·5	10·0	4·8	1·0
−0·5	2·9	0·5	20·0	4·5	0·3
0·0	0·6	0·4			

As noted in Hill (1969), where an array of results is needed AS 2 is a little faster than indicated by the above figures, but not sufficiently so as to affect the comparison to a substantial degree.

Accuracy

For each argument in the series $-20(0{\cdot}01)20$, the ratio was found of the results given by the two algorithms, until the point was reached that a zero result was returned by either.

The maximum and minimum values of this ratio were 1·0000000009 and 0·99999999901.

Precision

For a double precision version:

(1) change *FUNCTION* to *DOUBLE PRECISION FUNCTION*
(2) change *REAL* to *DOUBLE PRECISION*
(3) change the constants in the two *DATA* statements to double precision values
(4) in the statement labelled 30, change *EXP* to *DEXP*

For some compilers, it may be necessary to modify the constants in the two lengthy assignments to *ALNORM*, but since these are *ad hoc* constants (not calculable from any simple formula), and since ISO Fortran does allow expressions to contain a mixture of real and double precision terms (see the last sentence of para 6.1 of the Standard), there seems little point in a modification unless the compiler demands it.

Additional Comments

For full accuracy it is important to use the *UPPER* parameter for the required tail area, and not to resort to

$$1{\cdot}0 - ALNORM(X, .TRUE.)$$

instead of

$$ALNORM(X, .FALSE.)$$

since the former can lead to a severe loss of significant figures.

Expressions may, of course, be used for either parameter. To many users it seems to come naturally to use an expression for a numerical variable, but not for a logical one, yet the latter can sometimes be useful. For instance

$$ALNORM(X, X.GT.0{\cdot}0)$$

can be used to find the smaller of the two tail areas, and twice this value is needed for a two-tail test.

Acknowledgement

I am grateful to Dr J. A. Nelder for suggesting a number of improvements in presentation.

References

Adams, A. G. (1969). Algorithm 39. Areas under the normal curve. *Computer J.*, **12**, 197–198.
Cooper, B. E. (1968). Algorithm AS 2. The normal integral. *Appl. Statist.*, **17**, 186–187.
Hill, I. D. (1969). Remark AS R2. A remark on algorithm AS 2. *Appl. Statist.*, **18**, 299–300.
Hitchin, D. (1973). Remark AS R8. A remark on algorithm AS 4 and AS 5. *Appl. Statist.*, **22**, 428.

```
      FUNCTION ALNORM(X, UPPER)
C
C        ALGORITHM AS 66  APPL. STATIST. (1973) VOL.22, NO.3
C
C        EVALUATES THE TAIL AREA OF THE STANDARDISED NORMAL CURVE
C        FROM X TO INFINITY IF UPPER IS .TRUE. OR
C        FROM MINUS INFINITY TO X IF UPPER IS .FALSE.
C
      REAL LTONE, UTZERO, ZERO, HALF, ONE, CON, Z, Y, X
      LOGICAL UPPER, UP
C
C        LTONE AND UTZERO MUST BE SET TO SUIT THE PARTICULAR COMPUTER
C        (SEE INTRODUCTORY TEXT)
C
      DATA LTONE, UTZERO /7.0, 18.66/
      DATA ZERO, HALF, ONE, CON /0.0, 0.5, 1.0, 1.28/
      UP = UPPER
      Z = X
      IF (Z.GE.ZERO) GOTO 10
      UP = .NOT.UP
      Z = -Z
   10 IF (Z.LE.LTONE.OR.UP.AND.Z.LE.UTZERO) GOTO 20
      ALNORM = ZERO
      GOTO 40
   20 Y = HALF * Z * Z
      IF (Z.GT.CON) GOTO 30
C
      ALNORM = HALF - Z * (0.398942280444 - 0.399903438504 * Y /
     1  (Y + 5.75885480458 - 29.8213557808 /
     2  (Y + 2.62433121679 + 48.6959930692 /
     3  (Y + 5.92885724438))))
      GOTO 40
C
   30 ALNORM = 0.398942280385 * EXP(-Y) /
     1  (Z - 3.8052E-8 + 1.00000615302 /
     2  (Z + 3.98064794E-4 + 1.98615381364 /
     3  (Z - 0.151679116635 + 5.29330324926 /
     4  (Z + 4.8385912808 - 15.1508972451 /
     5  (Z + 0.742380924027 + 30.789933034 / (Z + 3.99019417011))))))
C
   40 IF (.NOT.UP) ALNORM = ONE - ALNORM
      RETURN
      END
```