equal to

$$\frac{E_2 \| A \|_2}{\sqrt{1 - E_1}} \quad \text{if } E_1 < 1.$$

The computation of W and R was done with double-precision inner products.

The results of the tests are summarized as follows:

(a) Both *QR* 2 and *HSI* found the dominant eigenvalues to better relative accuracy, but the same or worse absolute accuracy than the other eigenvalues.

(b) *QR* 2 was on the average 1.8 times faster than *HSI* (*QR* 2 required 2.5 seconds on a Hilbert segment of order 15).

(c) *QR* 2 always found orthogonal eigenvectors ($E_1 \sim 10^{-11}$);

(d) in most cases $E_1 \sim 10^{-11}$ for *HSI* also, but several times *HSI* found two eigenvectors almost parallel ($E_1 \sim 1.0$).

(e) $E_2 \sim 10^{-11}$ for both *QR* 2 and *HSI* with neither being consistently better than the other.

Conclusions. The orthonormalized eigenvectors, speed, and comparable accuracy would recommend *symmetric QR* 2 over the Wilkinson procedures for finding all of the eigenvalues and eigenvectors of a real symmetric matrix. The latter procedures are good for finding selected eigenvalues and eigenvectors.

REMARK ON ALGORITHM 296 [E2] GENERALIZED LEAST SQUARES FIT BY ORTHOGONAL POLYNOMIALS

[G. J. Makinson, *Comm. ACM 10* (Feb. 1967), 87]

G. J. Makinson (Recd. 21 Mar. 1967)
University of Liverpool, Liverpool 3, England

The second sentence of the first comment should read "The weights should be provided inversely proportional to the square of the standard error of the observations."

instead of

"The weights should be provided inversely proportional to the standard error of the observations."

REMARKS ON: ALGORITHM 123 [S15] REAL ERROR FUNCTION, ERF(x)

[Martin Crawford and Robert Techo *Comm. ACM 5* (Sept. 1962), 483]

ALGORITHM 180 [S15] ERROR FUNCTION—LARGE X

[Henry C. Thacher Jr. *Comm. ACM 6* (June 1963), 314]

ALGORITHM 181 [S15] COMPLEMENTARY ERROR FUNCTION—LARGE X

[Henry C. Thacher Jr. *Comm. ACM 6* (June 1963), 315]

ALGORITHM 209 [S15] GAUSS

[D. Ibbetson. *Comm. ACM 6* (Oct. 1963), 616]

ALGORITHM 226 [S15] NORMAL DISTRIBUTION FUNCTION

[S. J. Cyvin. *Comm. ACM 7* (May 1964), 295]

ALGORITHM 272 [S15] PROCEDURE FOR THE NORMAL DISTRIBUTION FUNCTIONS

[M. D. MacLaren. *Comm. ACM 8* (Dec. 1965), 789]

ALGORITHM 304 [S15] NORMAL CURVE INTEGRAL

[I. D. Hill and S. A. Joyce. *Comm. ACM 10* (June 1967), 374]

I. D. Hill and S. A. Joyce (Recd. 21 Nov. 1966)
Medical Research Council,
Statistical Research Unit, 115 Gower Street, London W.C.1., England

These algorithms were tested on the ICT Atlas computer using the Atlas Algol compiler. The following amendments were made and results found:

ALGORITHM 123

(i) **value** x; was inserted.

(ii) $abs(T) \leqslant_{10} -10$ was changed to $Y - T = Y$ both these amendments being as suggested in [1].

(iii) The labels 1 and 2 were changed to $L1$ and $L2$, the **go to** statements being similarly amended.

(iv) The constant was lengthened to 1.12837916710.

(v) The extra statement $x := 0.707106781187 \times x$ was made the first statement of the algorithm, so as to derive the normal integral instead of the error function.

The results were accurate to 10 decimal places at all points tested except $x = 1.0$ where only 2 decimal accuracy was found, as noted in [2]. There seems to be no simple way of overcoming the difficulty [3], and any search for a method of doing so would hardly be worthwhile, as the algorithm is slower than Algorithm 304 without being any more accurate.

ALGORITHM 180

(i) $T := -0.56418958/x/exp(v)$ was changed to $T := -0.564189583548 \times exp(-v)/x$. This is faster and also has the advantage, when v is very large, of merely giving 0 as the answer instead of causing overflow.

(ii) The extra statement $x := 0.707106781187 \times x$ was made as in (v) of Algorithm 123.

(iii) **for** $m := m + 1$ was changed to **for** $m := m + 2$. $m+1$ is a misprint, and gives incorrect answers.

The greatest error observed was 2 in the 11th decimal place.

ALGORITHM 181

(i) Similar to (i) of Algorithm 180 (except for the minus sign).

(ii) Similar to (ii) of Algorithm 180.

(iii) m was declared as **real** instead of **integer**, as an alternative to the amendment suggested in [4].

The results were accurate to 9 significant figures for $x \leqslant 8$, but to only 8 significant figures for $x = 10$ and $x = 20$.

ALGORITHM 209

No modification was made. The results were accurate to 7 decimal places.

ALGORITHM 226

(i) $10 \uparrow m/(480 \times sqrt(2 \times 3.14159265))$ was changed to $10 \uparrow m \times 0.000831129750836$.

(ii) **for** $i := 1$ **step** 1 **until** $2 \times n$ **do** was changed to $m := 2 \times n$; **for** $i := 1$ **step** 1 **until** m **do**.

(iii) $-(i \times b/n) \uparrow 2/8$ was changed to $-(i \times b/n) \uparrow 2 \times 0.125$.

(iv) **if** $i = 2 \times n - 1$ was changed to **if** $i = m - 1$

(v) $b/(6 \times n \times sqrt(2 \times 3.14159265))$ was changed to $b/(15.0397696478 \times n)$.

Tests were made with $m = 7$ and $m = 11$ with the following results:

x	*Number of significant figures correct*		*Number of decimal places correct*	
	$m = 7$	$m = 11$	$m = 7$	$m = 11$
−0.5	7	11	7	11
−1.0	7	10	7	10
−1.5	7	10	8	10
−2.0	7	9	8	10
−2.5	6	9	8	11
−3.0	6	7	8	9
−4.0	5	7	10	11
−6.0	2	1	12	10
−8.0	0	0	11	9

Perhaps the comment with this algorithm should have referred to decimal places and not significant figures. To ask for 11 significant figures is stretching the machine's ability to the limit, and where 10 significant figures are correct, this may be regarded as acceptable.

ALGORITHM 272

The constant .99999999 was lengthened to .9999999999.

The accuracy was 8 decimal places at most of the points tested, but was only 5 decimal places at $x = 0.8$.

ALGORITHM 304

No modification was made. The errors in the 11th significant figure were:

$abs(x)$	$x > 0 \equiv upper$	$x > 0 \not\equiv upper$
0.5	1	1
1.0	1	2
1.5	21[a](5)	2
2.0	25[a](0)	4
3.0	0	0
4.0	2	3
6.0	6	0
8.0	14	0
10.0	23	0
20.0	35	0

[a] Due to the subtraction error mentioned in the comment section of the algorithm. Changing the constant 2.32 to 1.28 resulted in the figures shown in brackets.

To test the claim that the algorithm works virtually to the accuracy of the machine, it was translated into double-length instructions of Mercury Autocode and run on the Atlas using the EXCHLF compiler (the constant being lengthened to 0.39894228040143267793994 6). The results were compared with hand calculations using Table II of [5]. The errors in the 22nd significant figure were:

$abs(x)$	$x > 0 \equiv upper$	$x > 0 \not\equiv upper$
1.0	2	3
2.0	7	1
4.0	2	0
8.0	8	0

Timings. Timings of these algorithms were made in terms of the Atlas "Instruction Count," while evaluating the function 100 times. The figures are not directly applicable to any other computer, but the relative times are likely to be much the same on other machines.

INSTRUCTION COUNT FOR 100 EVALUATIONS

$abs(x)$	Algorithm number							
	123	180	181	209	226 $m = 7$	272	304[a]	304[b]
0.5	58			8	97	24	25	24
1.0	65[c]			8	176	24	29	29
1.5	164	128	127	9	273	25	35	35
2.0	194	78	90	8	387	24	39	39
2.5	252	54	68	10	515	24	131	44
3.0		42	51	9	628	25	97	50
4.0		27	39	9	900[d]	25	67	44
6.0		15	30	6	1400[d]	16	49	23
8.0		9	28	7	2100[d]	18	44	11
10.0		10	25	5	2700[d]	16	38	11
20.0		9	22	5	6500[d]	16	32	11
30.0		9	9	5	10900[d]	16	11	11

[a] Readings refer to $x > 0 \equiv upper$.

[b] Readings refer to $x > 0 \not\equiv upper$.

[c] Time to produce incorrect answer. A count of 120 would fit a smooth curve with surrounding values.

[d] 100 times Instruction Count for 1 evaluation.

Opinion. There are advantages in having two algorithms available for normal curve tail areas. One should be very fast and reasonably accurate, the other very accurate and reasonably fast. We conclude that Algorithm 209 is the best for the first requirement, and Algorithm 304 for the second.

Algorithms 180 and 181 are faster than Algorithm 304 and may be preferred for this reason, but the method used shows itself in Algorithm 181 to be not quite as accurate, and the introduction of this method solely for the circumstances in which Algorithm 180 is applicable hardly seems worth while.

Acknowledgment. Thanks are due to Miss I. Allen for her help with the double-length hand calculations.

REFERENCES:

1. THACHER, HENRY C. JR. Certification of Algorithm 123. *Comm. ACM 6* (June 1963), 316.

2. IBBETSON, D. Remark on Algorithm 123. *Comm. ACM 6* (Oct. 1963), 618.

3. BARTON, STEPHEN P., AND WAGNER, JOHN F. Remark on Algorithm 123. *Comm. ACM 7* (Mar. 1964), 145.

4. CLAUSEN, I., AND HANSSON, L. Certification of Algorithm 181. *Comm. ACM 7* (Dec. 1964), 702.

5. SHEPPARD, W. F. *The Probability Integral.* British Association Mathematical Tables VII, Cambridge U. Press, Cambridge, England, 1939.