FISHTAIL: Practical F integration on small computers in BASIC and C

PHILLIP L. EMERSON
Cleveland State University, Cleveland, Ohio

Computer algorithms can be used to calculate F significance levels directly, thus avoiding the use of tables and resolving questions that arise when the degrees of freedom fall between the values given in an available published table or when the exact probability level is of interest. An algorithm is presented that finds F probabilities and that represents a good compromise among the goals of accuracy, speed, and compactness. Unlike several others, this method has the capability of interpolation for approximations to significance levels, when noninteger values are given for the degrees of freedom.

The fundamental idea underlying this algorithm was introduced with the *Biometrika* tables of the incomplete beta function, from which the integrals of F , t , and some other distributions can be obtained by the use of transformations (Kendall & Stuart, 1963, pp. 375-378). Following the suggestion of Bock (1975, p. 150), a computer algorithm based on incomplete beta has been developed to bypass the tables. The method uses a finite number of terms of an infinite series that is known to converge to the desired value. The convergence is rapid enough for practical computations, under conditions that can be induced. The method is based on the transformation

$$x = \nu_2/(\nu_2 + \nu_1 F).$$

In this expression, F is Fisher's F statistic with ν_1 and ν_2 degrees of freedom for the numerator and the denominator, respectively. The transformed variate, x , is the variate of a beta distribution with parameters $a = \nu_2/2$ and $b = \nu_1/2$. The series expression for P , the tail area of Fisher's distribution beyond the point F (Bock, 1975, p. 150), is

$$P = C[1 + \sum_{n=1}^{\infty} t_n], \quad (1)$$

where

$$C = x^a(1-x)^b\Gamma(a+b)/[\Gamma(b)\Gamma(a+1)] \quad (2)$$

and

$$t_n = x^n B(a+1, n)/B(a+b, n). \quad (3)$$

In the last expression above, $B(c, d)$ is the beta function defined as

$$B(c, d) = \Gamma(c)\Gamma(d)/\Gamma(c+d).$$

Fortunately, several of the factors in Equation 3 divide out, and the calculations can be performed by taking

$$t_1 = x(a+b)/(a+1),$$

and

$$t_n = t_{n-1}x(a+b+n-1)/(a+n), \quad n > 1.$$

The initial numerical tests of this general method indicated slow convergence, in some cases, which was fortunately circumvented via two modifications of the direct pattern of computations. The first is to compute the natural logarithms of the factors of the coefficient C and then exponentiate the sum of these to obtain the value of C . Stirling's approximation for the logarithm of the gamma function (Kendall & Stuart, 1963, p. 81) is used.

The second tactic that proves helpful, under some conditions, is to find the F tail indirectly as the complement of the tail of $1/F$ (see Kendall & Stuart, 1963, p. 379). One condition for that ploy is apparent from Equation 3. Clearly, small x is conducive to fast convergence, and the inversion makes x smaller when $\nu_1 F < \nu_2$. An additional condition is roughly that the F value not be very large relative to some function of the second and third moments of the F distribution. As $\nu_2 \rightarrow \infty$, the standard deviation of the distribution approaches $\sqrt{2/\nu_1}$, and the skewness is a monotonic function of $1/\nu_1$ (Kendall & Stuart, 1963, p. 378). These considerations, together with many numerical tests, led to the heuristic adoption of $F < 1 + 20/\nu_1 + 10/\sqrt{\nu_1}$ as the additional condition.

A BASIC Version. The listing of a Microsoft BASIC/D implementation is shown in Listing 1. The algorithm is implemented in the three subroutines headed by the labels FTAIL, SERIES, and LOGAMMA in the REM lines. A replaceable main program is included, which simply reads parameter values and prints results on the console. The LOGAMMA routine uses a five-term form of Stirling's approximation, together with a simple operation that improves accuracy when the argument to the gamma function is small. If the argument is less than 5, an integer is added to it that makes the effective argument at least 5. Then the result of the Stirling approximation is reduced by the recursive property of the gamma function.

The LOG() and EXP() functions are assumed to return double-precision values. The Microsoft BASIC (of MS-DOS versions 2.0 and later) supplied with an IBM PC can be instructed to use double-precision transcendental functions, by the execute command BASIC/D rather than merely BASIC. Note, however, that even the BASIC/D transcendental functions do not return correct double-precision values, unless they are called with double-precision arguments.

Although IBM Microsoft BASIC/D has full double-precision capabilities, it still (MS-DOS version 3.1) uses a value of about 10^{-38} or e^{-87} as the criterion on the size of a divisor, for the invocation of the "division-by-zero"

This paper benefited from suggestions by an anonymous reviewer. The author's mailing address is Department of Psychology, Cleveland State University, Cleveland, OH 44115.

LISTING 1
The Microsoft BASIC/D Version of FISHTAIL, Giving Significance Levels of Fisher's F Values

```

90 REM -- --- --- PROGRAM FISHTAIL.BAS --- --- ---
100 INPUT "DF NUMERATOR" ; DFN#
110 INPUT "DF DENOMINATOR" ; DFD#
115 INPUT "F" ; F#
120 GOSUB 500
160 PRINT DFN#,DFD#,F#," PROBABILITY=" ; P# : GOTO 100
480 REM *****
490 REM -- --- --- FTAIL ROUTINE, RETURNING P# --- --- ---
500 IF F#*DFN# < DFD# AND F# < 1 + 20/DFN# + 10/SQR(DFN#) GOTO 570
510 DN# = DFN# : DD# = DFD# : FPRIME# = F# : GOSUB 700
550 P# = S# : GOTO 590
570 DN# = DFD# : DD# = DFN# : FPRIME# = 1/F# : GOSUB 700
580 P# = 1 - S#
590 IF P# <= 1 AND P# >= 0 THEN RETURN
600 IF F# > 1 THEN P# = 0 ELSE IF F# = 1 THEN P# = .5 ELSE P# = 1
610 RETURN
680 REM *****
690 REM -- --- --- SERIES ROUTINE, RETURNING S# --- --- ---
700 SMALL = 1E-11 : EXPMIN = -87 : DN# = DN#/2 : DD# = DD#/2
740 X# = DD#/(DD# + DN#*FPRIME#)
750 A# = DN# + DD# : GOSUB 1500 : C# = G#
780 A# = DN# : GOSUB 1500 : C# = C# - G#
810 A# = DD# + 1 : GOSUB 1500 : C# = C# - G#
840 C# = C# + DD#*LOG(X#) + DN#*LOG(1 - X#)
850 IF C# > EXPMIN GOTO 880
860 S# = -1 : RETURN
880 DN# = DN# + DD# : DD# = DD# + 1 : C# = EXP(C#) : ER = SMALL/C#
920 T# = DN#*X#/DD# : N# = 0 : S# = 1 + T# : T1# = 0
960 N# = N# + 1
970 T# = T#*(DN# + N#)*X#/(DD# + N#)
980 S# = S# + T#
990 IF T# < ER AND T# < T1# GOTO 1020
1000 T1# = T#
1010 GOTO 960
1020 S# = S#*C# : RETURN
1480 REM *****
1490 REM -- --- --- LOGAMMA ROUTINE, RETURNING G# --- --- ---
1500 HL2PI# = .9189385332046728# : ARGMIN = 6
2000 N = INT(ARGMIN - A#) : IF N >= 1 THEN A# = A# + N
2030 G# = 1/(A#*A#)
2040 G#=(1#-G#*(1#/30-G#*(1#/105-G#*(1#/140-G#/99))))/(12*A#)
2050 G# = G# + (A# - .5)*LOG(A#) - A# + HL2PI#
2060 IF N < 1 THEN RETURN
2070 FOR I = 1 TO N
2080 A# = A# - 1
2090 G# = G# - LOG(A#)
2100 NEXT I
2110 RETURN
2120 REM *****

```

error exit. This is important in the computation of the value of C in Equation 2. EXPMIN is defined to be -87 in the program, to avoid error terminations of program execution in extreme cases when the resulting probability would be very near 0 or 1. The SERIES routine returns -1 when the value to be exponentiated is less than EXPMIN. In the FTAIL routine, such cases are treated by setting the returned value of P equal to 0, .5, or 1, depending on whether $F > 1$, $F = 1$, or $F < 1$. The middle case probably never occurs, because C is moderate in size when F is near 1. The other two cases give the reasonable results of $P = 0$ or 1 when the true probability is within about 10^{-38} of those values.

On a standard IBM PC, using BASIC/D, the FISHTAIL.BAS program was found to be accurate to nine decimal places. Accuracy to three places was found with the version of Microsoft BASIC not having double-precision transcendental functions. Execution times with the IBM PC were from about 1 to 13 sec, for cases with $\nu_1 + \nu_2$ between 2 and 2,000. The longer times were ob-

tained with large degrees of freedom together with F values near 1.

Interpolation. The F distribution is a special case of a more general distribution with continuously variable parameters, ν_1 and ν_2 . FISHTAIL evaluates the tail of that generalized F distribution. It provides a smooth, gently curved interpolation function, varying in a regular way between integral points. This continuity could be of some use in connection with df adjustments that are sometimes made to compensate for certain violations of the assumptions of analysis of variance (Box, 1954a, 1954b; Greenhouse & Geisser, 1959; Rogan, Keselman, & Mendoza, 1979; Scheffé, 1959). Box (1954a) used interpolations in some of his examples of df adjustments, for noninteger ν_1 and ν_2 . The main violations that have been the targets of these techniques are heterogeneity of variances, and nonuniformity of error correlations in repeated measures designs.¹

Comparisons. The Fisher() algorithm of Dorrer (1968) is fast, for small ν_1 and ν_2 . One of its main advantages

is that it is, a priori, almost as accurate as the square-root and arctangent functions used to compute kernels that are probability levels of a given F , for $(\nu_1, \nu_2) = (1, 1)$, $(1, 2)$, $(2, 1)$, and $(2, 2)$. From these, the probability level for the same F , with any other integer values of (ν_1, ν_2) , is computed recursively by simple arithmetical operations. Because of its simplicity and *prima facie* high accuracy, Fisher() was used to calculate F tail areas for a large number of cases, such as those shown in Table 1. Fisher() was coded in Datalight C with double-precision transcendental functions. The first 16 rows in Table 1 are representative of the 5% critical values given in published F tables. The last 13 cases are included because they were used by Wood and Wood (1986) to illustrate some points concerning the accuracy of their F tail computations. Fifteen decimal places are listed in Table 1 for comparisons of different implementations of the FISHTAIL method. These values are not generally accurate to 15 places, but they approach that accuracy in some cases.² Although Fisher() is accurate and easy to code, its execution times are almost exactly proportional to $\nu_1 + \nu_2$, and are considerably longer than those of FISHTAIL when $\nu_1 + \nu_2$ is much larger than 200.

Wood and Wood (1986) reported good accuracy for their method based on Simpson's rule, but execution times were as long as a minute under some conditions. Accuracy suffers in some cases of $\nu_1 > \nu_2$, but that may be acceptable for most statistical applications. Moreover, the F inversion technique could be used for such cases, since it

Table 1
Approximate F Tail Areas From Dorrier's Fisher() Algorithm

ν_1	ν_2	F	$P(F)$
1	1	161.4	.050007348139564
1	10	4.96	.050087650566468
1	100	3.94	.049890019981430
1	1000	3.85	.050023005677932
10	1	241.9	.049998116427350
10	10	2.97	.050413178117376
10	100	1.92	.050909757262547
10	1000	1.84	.050023569195274
100	1	253.	.050004052864262
100	10	2.59	.049896332407362
100	100	1.39	.050634869595963
100	1000	1.26	.049750074504041
500	1	254.	.050005824432254
500	10	2.55	.049874859506105
500	100	1.30	.053906943373867
500	1000	1.13	.055220938408803
9	130	2.	.044163262922587
9	130	3.	.002780411028556
9	130	4.	.000156042640529
9	130	5.	.000008791323841
9	130	6.	.000000521923822
12	4	2.	.263513866306920
12	4	10.	.019613334771568
12	4	20.	.005342733051063
12	4	30.	.002444552336554
12	4	40.	.001395301686485
12	4	50.	.000900881740792
12	4	60.	.000629298573474
12	4	70.	.000464288671835

Table 2
Decimal Places of Agreement with Dorrier's Fisher() Algorithm

		FISHTAIL Implementation				
ν_1	ν_2	Microsoft BASIC	Microsoft BASIC/D	MS-DOS Datalight C	MS-DOS Mix C	CP/M Mix C
1	1	7	11	11	11	9
1	10	6	10	10	11	8
1	100	4	10	10	10	9
1	1000	3	10	10	10	10
10	1	7	12	12	12	9
10	10	6	10	10	12	10
10	100	4	10	10	10	9
10	1000	3	10	10	10	10
100	1	5	11	11	11	9
100	10	5	11	11	12	10
100	100	6	10	10	12	8
100	1000	4	10	10	10	7
500	1	4	11	11	11	10
500	10	4	11	11	10	9
500	100	5	10	10	12	8
500	1000	3	10	10	10	9
9	130	4	10	10	10	9
9	130	4	10	10	10	9
9	130	5	9	9	9	9
9	130	3	10	10	10	8
9	130	3	10	10	9	7
12	4	6	10	10	10	10
12	4	7	11	11	11	10
12	4	7	12	12	12	11
12	4	8	12	11	12	10
12	4	8	12	12	13	11
12	4	8	13	13	13	11
12	4	9	14	14	13	11
12	4	9	13	13	13	11

Note— F values were as listed in Table 1.

reverses the roles of ν_1 and ν_2 . Still, the BASIC/D version of FISHTAIL is more uniform in both speed and accuracy, and it is more compact. A possible advantage of the Wood and Wood (1986) method and some others, over FISHTAIL, is greater accuracy when only single-precision transcendental functions are available. A nice feature of the program of Wood and Wood (1986), not incorporated in FISHTAIL, is the suppression of nonsignificant digits in the output probability value.

The method of von Collani (1983) for computing the F tail, is packaged with others for chi-square and normal integrals. The F tail calculations appear to perform like those of Dorrier (1968) in that execution times are strongly dependent on the degrees of freedom. Von Collani (1983) examined three other methods of computing F significance levels. He found two (Jaspens, 1965; Ogasawara, 1982) to be lacking in accuracy in cases of small degrees of freedom, and the other (O'Grady, 1981) to be best suited for use on large computers.

Operating Environments. The third and fourth columns of Table 2 illustrate the differences that can be expected from FISHTAIL.BAS, running with single-precision and full double-precision versions of BASIC. These data suggest that three-place accuracy can be expected of FISHTAIL with single-precision arithmetic, and that nine-place accuracy is attainable if double-precision

LISTING 2

A C-Language Version of FISHTAIL, Compilable with the Datalight and Mix Implementations of C

```

/* fishtail.c */
#include <stdio.h>
double exp(),log(),sqrt() ;

#define ARGMIN 6
#define HL2PI .91893853320467275 /* log(2pi)/2 */
double logamma(a) double a ;
{ double g ; int i,n ; if((n=ARGMIN-a) > 0) a += n ; g = 1./(a*a) ;
  g = (1. - g*(1./30. - g*(1./105. - g*(1./140. - g/99.)))/(12.*a) ;
  g += ( (a - .5)*log(a) - a + HL2PI ) ;
  for(i=0; i < n ; i++) { a -= 1. ; g -= log(a) ; }
  return(g) ;
}

#define SMALL (1.e-11)
#define EXPMIN -700.
double series(F,dfn,dfd) double F,dfn,dfd ;
{ double x,c,er,s,n,t1,t ;
  dfn /= 2. ; dfd /= 2. ; x = dfd/(dfd + dfn*F) ;
  c = logamma(dfn+dfd) - logamma(dfn) - logamma(dfd+1.)
    + dfd*log(x) + dfn*log(1.-x) ;
  if( c < EXPMIN ) return( -1.) ; dfn += dfd ; dfd += 1. ;
  c = exp(c) ; er = SMALL/c ; t = dfn*x/dfd ;
  for(n=t1=0.,s=1.+t ; (t > er) || (t > t1) ; )
    { n += 1. ; t1 = t ; t *= ((dfn+n)*x/(dfd+n)) ; s += t ; }
  return(s*c) ;
}

double ftail(F,dfn,dfd) double F,dfn,dfd ;
{ double p ;
  if( (F*dfn >= dfd) || (F > 1. + 20./dfn + 10./sqrt(dfn) ) )
    p = series(F,dfn,dfd) ;
  else p = 1. - series(1./F,dfd,dfn) ;
  if( (p > 1.) || (p < 0.) )
    { if (F > 1.) p = 0. ;
      else if( F == 1. ) p = .5 ;
      else p = 1. ;
    }
  return(p) ;
}

main()
{ double dfn,dfd,F,p ;
  while(1)
    { fputs("\ntype df(num) df(den) F\n",stderr) ;
      scanf("%lf %lf %lf",&dfn,&dfd,&F) ;
      p = ftail(F,dfn,dfd) ;
      printf("%f %f %f ; p = %20.15e\n",dfn,dfd,F,p) ;
    }
}

```

transcendental functions are available. The memory requirements are modest, and FISHTAIL.BAS should run in as little as 64K.

Among some users of small computers, the C language is replacing BASIC as the language of choice for general-purpose programming. C programs tend to be more portable than BASIC, and C has other advantages as well (Kernighan & Ritchie, 1978; Plum, 1983).³ A C language version of FISHTAIL is shown in Listing 2. This program was found to compile and run successfully with Microsoft C, Datalight C,⁴ and Mix C.⁵ The latter two compilers are quite inexpensive, and they run on 8086-compatible computers using generic MS-DOS. Moreover, Mix C is available in another version that runs on 8080-compatible computers using the CP/M operating system.

The last three columns of Table 2 show results from compiling and running the C version of FISHTAIL with (1) Datalight C under MS-DOS on an IBM PC, (2) Mix C under MS-DOS on an IBM PC, and (3) Mix C under Montezuma CP/M⁶ on a TRS-80 Model 4. The

different results in the MS-DOS environment were almost indistinguishable, showing a minimum of nine places of agreement with the Fisher() algorithm. Mix C with CP/M on the TRS-80 gave less accuracy in general, with a minimum of seven places. Execution times of the compiled C programs were about one quarter those of the interpreted BASIC program. For execution of the compiled programs, 64K of memory is adequate. The size of the .obj file produced by the Datalight compiler was 3.6K, and the linked .exe file was 26K. For compiling, Datalight requires 192K and Mix requires 64K.

REFERENCES

- BOCK, R. D. (1975). *Multivariate statistical methods in behavioral research*. New York: McGraw-Hill.
- Box, G. E. P. (1954a). Some theorems on quadratic forms applied to the study of analysis of variance problems: I. Effect of inequality of variances in the one-way classification. *Annals of Mathematical Statistics*, 25, 290-302.
- Box, G. E. P. (1954b). Some theorems on quadratic forms applied to

- the study of analysis of variance problems: II. Effect of inequality of variance and of correlation of errors in the two-way classification. *Annals of Mathematical Statistics*, **25**, 484-498.
- DORRER, E. (1968). Algorithm 322 F -distribution [514]. *Communications of the Association for Computing Machinery*, **11**, 116-117.
- GREENHOUSE, S. W., & GEISSER, S. (1959). On methods in the analysis of profile data. *Psychometrika*, **24**, 95-112.
- JASPEN, N. (1965). The calculation of probabilities corresponding to values of z , t , F , and chi-square. *Educational & Psychological Measurement*, **25**, 877-880.
- KENDALL, M. G., & STUART, A. (1963). *The advanced theory of statistics* (Vol. 1, rev. ed.). New York: Hafner.
- KERNIGHAN, B. W., & RITCHIE, D. M. (1978). *The C programming language*. Englewood Cliffs, NJ: Prentice-Hall.
- OGASAWARA, T. H. (1982). The calculation of the significance level of F , t , and r on the Apple II. *Behavior Research Methods & Instrumentation*, **14**, 492-493.
- O'GRADY, K. E. (1981). Probabilities and critical values for chi-square, t , and F . *Behavior Research Methods & Instrumentation*, **13**, 55-56.
- PLUM, T. (1983). *Learning to program in C*. Englewood Cliffs, NJ: Prentice-Hall.
- ROGAN, J. C., KESELMAN, H. J., & MENDOZA, J. L. (1979). Analysis of repeated measures. *British Journal of Mathematical & Statistical Psychology*, **32**, 269-286.
- SCHEFFÉ, H. (1959). *The analysis of variance*. New York: Wiley.
- VON COLLANI, G. (1983). Computing probabilities for F , t , chi-square, and z in BASIC. *Behavior Research Methods & Instrumentation*, **15**, 543-544.
- Wike, E. L. (1985). *Numbers: A primer of data analysis*. Columbus, OH: Merrill.
- WOOD, D. L., & WOOD, D. (1986). Precise F integration. *Behavior Research Methods, Instruments, & Computers*, **18**, 405-406.

NOTES

1. The df adjustment techniques have not grown very popular in small-scale research, because they are difficult to use (Wike, 1985). The occasional need for interpolation is only one of several nuisances.
2. I explored the limiting accuracy of FISHTAIL with 8-byte double-precision arithmetic, as follows: I changed SMALL from 1e-11 to 1e-20 and increased ARGMIN from 6 to 12. I compiled with Datalight C, and found the agreements with Fisher() to increase, generally, by about 2 decimal places. In the 29 cases in Table 1, there were 6 agreements to 15 places, 9 to 14 places, 10 to 13 places, 3 to 12 places, and only 1 agreement to 11 places. The 11-place agreement was with the largest number of degrees of freedom, $\nu_1 = 500$ and $\nu_2 = 1000$, and the 12-place agreements were with the remaining three cases of $\nu_2 = 1000$. The FISHTAIL and Fisher() computations follow quite different patterns, so results that match are not likely to arise from the same computational errors. Thus, the entries in Table 1 are very likely correct to at least 11 places for large degrees of freedom, and 13 to 15 places for small degrees of freedom ($\nu_1 + \nu_2 \leq 600$).
3. C is a modular structured language that is equally suitable for high-level simulation programming, statistical programming, and low-level real-time hardware programming. It provides advanced high-level features such as dynamic memory allocation, recursive subroutine calls, and self-referencing data structure. In short, C affords more freedom and flexibility than do other well-known languages that are readily available for small computers.
4. Datalight, 8919 NE 142nd Way, Bothell, WA 98011.
5. Mix Software, Inc., 2116 East Arapaho, Suite 363, Richardson, TX 75081.
6. Montezuma Micro, CP/M Support Department, P.O. Box 32027, Dallas, TX 75232.

(Revision accepted for publication December 7, 1987.)