2.20 Binomial Coefficients

A. Purpose

Compute the binomial coefficient $\binom{n}{k} = \frac{n!}{(n-k)!\,k!}$ which gives the number of ways one can select k items from a set containing n items. Binomial coefficients also appear in a number of other contexts.

B. Usage

B.1 Program Prototype, Single Precision

REAL SBINOM, ANS

INTEGER N, K

Assign values to N and K. $\boxed{\textbf{ANS = SBINOM(N, K)}}$

B.2 Argument Definitions

N, K [in] The input integers n and k as described in ”Purpose” above. One must have $0 \le \text{K} \le \text{N}$.

B.3 Modifications for Double Precision

For double-precision usage change the name SBINOM to DBINOM, and change the REAL declaration to DOUBLE PRECISION.

C. Examples and Remarks

The program DRSBINOM compares the true values with the results of computing the binomial coefficients for two large values of N where SBINOM uses a different algorithm internally. It then finds the first case where results using the Pascal triangle give different results from SBINOM (the true value for the case printed is 67863915, which is slightly closer to the value given by SBINOM). If one is computing binomial coefficients for strictly increasing values of N, one may want to use the Pascal triangle code in DRSBINOM instead of SBINOM for computing the binomial coefficients.

D. Functional Description

For N $\le$ 150, the routine maintains and uses a table of saved factors of factorials, ϕ_j. If the input N is larger than any seen so far, this table is extended. When there is no problem with overflow, $\phi_j = j\phi_{j-1}$. To deal with overflow a second table i_ν is maintained. Initially $\nu = 0$. When adding entry j to the table would cause an overflow, ν is incremented, i_ν is set to $j-1$, and ϕ_j to j.

To compute the binomial coefficients, the code uses

$$\binom{n}{k} = \frac{n!}{(n-k)!\,k!}, \quad j! = \Bigl(\prod_{\ell=1}^{\nu-1} \phi(i_\ell)\Bigr)\phi(j),$$

with ν such that $i_{\nu-1} < j \le i_\nu$. When a factorial is so large it is represented by more than one piece, the pieces are combined in such a way as to avoid overflow. Finally, for large values, the result is refined by setting the result equal to $p \times \lfloor r/p + .5\rfloor$, where r is the originally computed result, p is either one prime or the product of two primes in the interval $\bigl(\max(n-k,\ k),\ n\bigr]$ and $\lfloor\cdot\rfloor$ denotes the integer part.

In the case of IEEE single precision arithmetic, only one piece of the factorial is needed for N $\le$ 33, and for IEEE double precision 150! does not overflow. When only one piece is needed for the factorial and it is already computed, the binomial coefficient is computed with one multiply, one divide, and a few tests.

If N > 150, the log gamma function from Chapter 2.3 is used. In this case

$$\binom{n}{k} = e^{\log\Gamma(n+1) - \log\Gamma(n-k+1) - \log\Gamma(k+1)}$$

It is very easy to shorten the code by replacing the call to the log gamma function for large N with an error message reporting too large an N. It is only slightly more difficult to replace the code for keeping pieces of the factorial with code that calls log gamma whenever N! would overflow. This would lose some precision as illustrated by the difference in the errors for N = 150 and N = 151 in the results.

E. Error Procedures and Restrictions

An error exit is taken if the condition $0 \le \text{K} \le \text{N}$ is not satisfied, or if the result would overflow. Errors are processed using the routines in Chapter 19.2. If one references one of these routines to change the error action, then instead of stopping on an error, the routine will return with a result of −1.

F. Supporting Information

The source language is ANSI Fortran 77.

Entry	**Required Files**
DBINOM	AMACH, DBINOM, DERM1, DERV1, DLGAMA, DGAMMA, ERFIN, ERMSG, IERM1, IERV1
SBINOM	AMACH, ERFIN, ERMSG, IERM1, IERV1, SBINOM, SERM1, SERV1, SLGAMA, SGAMMA

Algorithm and code due to F. T. Krogh, JPL, December 1995.

DRSBINOM

```
c      program DRSBINOM
c>> 1998-05-12 DRSBINOM Krogh Ensure Pascal print output.
c>> 1995-12-15 DRDBINUM Krogh Initial Code.
c--S replaces "?": DR?BINOM, ?BINOM
c
c Test SBINOM for computing binomial coefficients.
c Checks a large value computed with factorials, one a little larger
c that is computed using log gamma, and then finds first that differs
c from that obtained using the Pascal triangle.
c
      integer K, N, NMAX
      parameter (NMAX = 120)
      external SBINOM
      real            BC1, BC2, P(0:NMAX), TP, SBINOM
c             ( 150 )             ( 151 )
c       BC1 = (     )       BC2 = (     )
c             (  30 )             (  29 )
      parameter (BC1 = 32198785340494567031466236484400.E0)
      parameter (BC2 = 9880808670399701168713050486000.E0)
c
   10 format (/'   N   K',12X,'SBINOM',20X, A, 8X, ' Col. 2 - Col. 1')
   20 format (I4, I4,1P,E26.17,E26.17,1P,E16.7)
c
      print 10, ' True '
      TP = SBINOM(150, 30)
      print 20, 150, 30, TP, BC1, BC1 - TP
      TP = SBINOM(151, 29)
      print 20, 151, 29, TP, BC2, BC2 - TP
c
      print 10, 'Pascal'
      P(0) = 1.E0
      do 80 N = 0, NMAX
         do 40 K = 0, N
            TP = SBINOM(N, K)
            if (TP .ne. P(K)) go to 100
   40    continue
c                  Update the Pascal Triangle.
         P(N+1) = 1.E0
         do 60 K = N, 1, -1
            P(K) = P(K) + P(K-1)
   60    continue
   80 continue
  100 print 20, N, K, TP, P(K), P(K) - TP
      stop
      end
```

ODSBINOM

```
   N   K            SBINOM                         True         Col. 2 - Col. 1
 150  30   3.21987793183348741E+31   3.21987841540381525E+31   4.8357033E+24
 151  29   9.88067403414396494E+30   9.88080882937285197E+30   1.3479523E+26

   N   K            SBINOM                        Pascal        Col. 2 - Col. 1
  27  10   8.43628600000000000E+06   8.43628500000000000E+06  -1.0000000E+00
```