NAG Library Function Document

nag_hypergeom_dist (g01blc)

1 Purpose

nag_hypergeom_dist (g01blc) returns the lower tail, upper tail and point probabilities associated with a hypergeometric distribution.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
void nag_hypergeom_dist (Integer n, Integer l, Integer m, Integer k,
     double *plek, double *pgtk, double *peqk, NagError *fail)
```

3 Description

Let X denote a random variable having a hypergeometric distribution with parameters n, l and m ($n \geq l \geq 0$, $n \geq m \geq 0$). Then

$$\text{Prob}\{X = k\} = \frac{\binom{m}{k}\binom{n-m}{l-k}}{\binom{n}{l}},$$

where $\max(0, l-(n-m)) \leq k \leq \min(l, m)$, $0 \leq l \leq n$ and $0 \leq m \leq n$.

The hypergeometric distribution may arise if in a population of size n a number m are marked. From this population a sample of size l is drawn and of these k are observed to be marked.

The mean of the distribution $= \dfrac{lm}{n}$, and the variance $= \dfrac{lm(n-l)(n-m)}{n^2(n-1)}$.

nag_hypergeom_dist (g01blc) computes for given n, l, m and k the probabilities:

$$\begin{aligned} \textbf{plek} &= \text{Prob}\{X \leq k\} \\ \textbf{pgtk} &= \text{Prob}\{X > k\} \\ \textbf{peqk} &= \text{Prob}\{X = k\}. \end{aligned}$$

The method is similar to the method for the Poisson distribution described in KnÏsel (1986).

4 References

KnÏsel L (1986) Computation of the chi-square and Poisson distribution *SIAM J. Sci. Statist. Comput.* **7** 1022–1036

5 Arguments

1: **n** – Integer *Input*

On entry: the parameter n of the hypergeometric distribution.

Constraint: $\textbf{n} \geq 0$.

2: **l** – Integer *Input*

On entry: the parameter l of the hypergeometric distribution.

Constraint: $0 \leq \mathbf{l} \leq \mathbf{n}$.

3: **m** – Integer *Input*

On entry: the parameter m of the hypergeometric distribution.

Constraint: $0 \leq \mathbf{m} \leq \mathbf{n}$.

4: **k** – Integer *Input*

On entry: the integer k which defines the required probabilities.

Constraint: $\max(0, \mathbf{l} - (\mathbf{n} - \mathbf{m})) \leq \mathbf{k} \leq \min(\mathbf{l}, \mathbf{m})$.

5: **plek** – double * *Output*

On exit: the lower tail probability, $\text{Prob}\{X \leq k\}$.

6: **pgtk** – double * *Output*

On exit: the upper tail probability, $\text{Prob}\{X > k\}$.

7: **peqk** – double * *Output*

On exit: the point probability, $\text{Prob}\{X = k\}$.

8: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_2_INT_ARG_GT

On entry, $\mathbf{k} = \langle value \rangle$ and $\mathbf{l} = \langle value \rangle$.
Constraint: $\mathbf{k} \leq \mathbf{l}$.

On entry, $\mathbf{k} = \langle value \rangle$ and $\mathbf{m} = \langle value \rangle$.
Constraint: $\mathbf{k} \leq \mathbf{m}$.

On entry, $\mathbf{l} = \langle value \rangle$ and $\mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{l} \leq \mathbf{n}$.

On entry, $\mathbf{m} = \langle value \rangle$ and $\mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{m} \leq \mathbf{n}$.

NE_4_INT_ARG_CONS

On entry, $\mathbf{k} = \langle value \rangle$, $\mathbf{l} = \langle value \rangle$, $\mathbf{m} = \langle value \rangle$ and $\mathbf{l} + \mathbf{m} - \mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{k} \geq \mathbf{l} + \mathbf{m} - \mathbf{n}$.

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_ARG_TOO_LARGE

On entry, **n** is too large to be represented exactly as a double precision number.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT_ARG_LT

On entry, $\mathbf{k} = \langle value \rangle$.
Constraint: $\mathbf{k} \geq 0$.

On entry, $\mathbf{l} = \langle value \rangle$.
Constraint: $\mathbf{l} \geq 0$.

On entry, $\mathbf{m} = \langle value \rangle$.
Constraint: $\mathbf{m} \geq 0$.

On entry, $\mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{n} \geq 0$.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

NE_VARIANCE_TOO_LARGE

On entry, the variance $= \frac{lm(n-l)(n-m)}{n^2(n-1)}$ exceeds 10^6.

7 Accuracy

Results are correct to a relative accuracy of at least 10^{-6} on machines with a precision of 9 or more decimal digits, and to a relative accuracy of at least 10^{-3} on machines of lower precision (provided that the results do not underflow to zero).

8 Parallelism and Performance

nag_hypergeom_dist (g01blc) is not threaded in any implementation.

9 Further Comments

The time taken by nag_hypergeom_dist (g01blc) depends on the variance (see Section 3) and on k. For given variance, the time is greatest when $k \approx lm/n$ (= the mean), and is then approximately proportional to the square-root of the variance.

10 Example

This example reads values of n, l, m and k from a data file until end-of-file is reached, and prints the corresponding probabilities.

10.1 Program Text

```
/* nag_hypergeom_dist (g01blc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 *
 */

#include <nag.h>
#include <nag_stdlib.h>
#include <stdio.h>
#include <nagg01.h>

int main(void)
{
  Integer exit_status = 0;
  double plek, peqk, pgtk;
  Integer k, l, m, n;
  NagError fail;

  INIT_FAIL(fail);

  printf("nag_hypergeom_dist (g01blc) Example Program Results\n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  printf("\n    n   l   m   k     plek      pgtk      peqk\n\n");

#ifdef _WIN32
  while ((scanf_s
          ("%" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "%*[^\n]",
           &n, &l, &m, &k)) != EOF) {
#else
  while ((scanf
          ("%" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "%*[^\n]",
           &n, &l, &m, &k)) != EOF) {
#endif
    /* nag_hypergeom_dist (g01blc).
     * Hypergeometric distribution function
     */
    nag_hypergeom_dist(n, l, m, k, &plek, &pgtk, &peqk, &fail);
    if (fail.code != NE_NOERROR) {
      printf("Error from nag_hypergeom_dist (g01blc).\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
    printf(" %4" NAG_IFMT "%4" NAG_IFMT "%4" NAG_IFMT "%4" NAG_IFMT
           "%10.5f%10.5f" "%10.5f\n", n, l, m, k, plek, pgtk, peqk);
  }

END:
  return exit_status;
}
```

10.2 Program Data

```
nag_hypergeom_dist (g01blc) Example Program Data
10   2   5   1    : n, l, m, k
40  10   3   2
155  35 122  22
1000 444 500 220
```

10.3 Program Results

```
nag_hypergeom_dist (g01blc) Example Program Results

    n   l   m   k     plek      pgtk      peqk

   10   2   5   1   0.77778   0.22222   0.55556
   40  10   3   2   0.98785   0.01215   0.13664
  155  35 122  22   0.01101   0.98899   0.00779
 1000 444 500 220   0.42429   0.57571   0.04913
```