

Module 20.7: nag_discrete_dist

Probabilities for Discrete Distributions

`nag_discrete_dist` provides procedures for computing probabilities for various parts of a binomial, Poisson or hypergeometric distribution.

Contents

Procedures

<code>nag_binom_prob</code>	20.7.3
Computes lower tail, upper tail or point probability for a binomial distribution with parameters n and p	
<code>nag_poisson_prob</code>	20.7.5
Computes lower tail, upper tail or point probability for a Poisson distribution with parameter λ	
<code>nag_hypergeo_prob</code>	20.7.7
Computes lower tail, upper tail or point probability for a hypergeometric distribution with parameters n , l , and m	

Examples

Example 1: Calculation of the probability of point k from a binomial distribution	20.7.11
---	---------

Additional Examples	20.7.13
----------------------------------	---------

References	20.7.14
-------------------------	---------

Procedure: nag_binom_prob

1 Description

`nag_binom_prob` returns the lower tail, upper tail or point probability for a binomial distribution with parameters n and p .

2 Usage

USE `nag_discrete_dist`

[*value* =] `nag_binom_prob`(*tail*, *n*, *p*, *k* [, *optional arguments*])

The function result is a scalar of type `real(kind=wp)`.

3 Arguments

3.1 Mandatory Arguments

tail — `character(len=1)`, `intent(in)`

Input: the type of probability to be returned:

- if **tail** = 'L' or 'l', the lower tail probability is returned;
- if **tail** = 'U' or 'u', the upper tail probability is returned;
- if **tail** = 'A' or 'a', the probability of any point **k** is returned.

Constraints: **tail** = 'L', 'l', 'U', 'u', 'A' or 'a'.

n — `integer`, `intent(in)`

Input: the number of independent Bernoulli trials.

Constraints: $n > 0$.

p — `real(kind=wp)`, `intent(in)`

Input: the probability of success in each of the independent Bernoulli trials.

Constraints: $0.0 < p < 1.0$.

k — `integer`, `intent(in)`

Input: the number of successes that defines the required probability.

Constraints: $0 \leq k \leq n$.

3.2 Optional Argument

error — `type(nag_error)`, `intent(inout)`, optional

The NAG *f*90 error-handling argument. See the Essential Introduction, or the module document `nag_error_handling` (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to `nag_set_error` before this procedure is called.

4 Error Codes

Fatal errors (`error%level = 3`):

<code>error%code</code>	Description
301	An input argument has an invalid value.

Failures (error%level = 2):

error%code	Description
201	n is too large to be represented exactly as a real(kind=wp) number.

5 Examples of Usage

A complete example of the use of this procedure appears in Example 1 of this module document.

6 Further Comments

6.1 Mathematical Background

Let X be a random variable from a binomial distribution with parameters n and p ($n > 0$ and $0 < p < 1$). Then

$$P\{X = k\} = \binom{n}{k} p^k (1-p)^{n-k}, \text{ for } k = 0, 1, \dots, n.$$

The mean of the distribution is np and the variance is $np(1-p)$.

6.2 Algorithmic Detail

The procedure computes, for any given values of n , p , and k , the following probabilities:

Lower tail probability = $P\{X \leq k\}$

Upper tail probability = $P\{X > k\}$

Point probability = $P\{X = k\}$.

The computation approach used is described in Knüsel [1] and is similar to the one used for the Poisson distribution.

6.3 Accuracy

The results should be 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).

6.4 Timing

The time taken by the procedure depends on the variance ($= np(1-p)$) and on k . For a given variance, the time is greatest when $k \approx np$ ($=$ the mean), and is then approximately proportional to the square root of the variance.

Procedure: nag_poisson_prob

1 Description

`nag_poisson_prob` returns the lower tail, upper tail or point probability for a Poisson distribution with parameter λ .

2 Usage

USE `nag_discrete_dist`

[*value* =] `nag_poisson_prob`(*tail*, *lambda*, *k* [, *optional arguments*])

The function result is a scalar of type `real(kind=wp)`.

3 Arguments

3.1 Mandatory Arguments

tail — `character(len=1)`, `intent(in)`

Input: the type of probability to be returned:

- if **tail** = 'L' or 'l', the lower tail probability is returned;
- if **tail** = 'U' or 'u', the upper tail probability is returned;
- if **tail** = 'A' or 'a', the probability of any point *k* is returned.

Constraints: **tail** = 'L', 'l', 'U', 'u', 'A' or 'a'.

lambda — `real(kind=wp)`, `intent(in)`

Input: the parameter of the Poisson distribution.

Constraints: $0.0 < \text{lambda} \leq 10^6$.

k — `integer`, `intent(in)`

Input: the value of the variable that defines the required probability.

Constraints: $k \geq 0$.

3.2 Optional Argument

error — `type(nag_error)`, `intent(inout)`, optional

The NAG *f*90 error-handling argument. See the Essential Introduction, or the module document `nag_error_handling` (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to `nag_set_error` before this procedure is called.

4 Error Codes

Fatal errors (`error%level = 3`):

<code>error%code</code>	Description
301	An input argument has an invalid value.

5 Examples of Usage

Assume that all relevant arguments have been declared correctly as described in Section 3, and that input and input/output arguments have been appropriately initialized. This example shows how a call to this procedure returns the lower tail probability of k from a Poisson distribution with parameter `lambda`.

```
prob = nag_poisson_prob('L', lambda, k)
```

6 Further Comments

6.1 Mathematical Background

Let X be a random variable having a Poisson distribution with parameter λ (> 0). Then

$$P\{X = k\} = e^{-\lambda} \frac{\lambda^k}{k!}, \quad k = 0, 1, 2, \dots$$

The mean and variance of the distribution are both equal to λ .

6.2 Algorithmic Detail

This procedure computes, for any given values of λ and k , the following probabilities:

Lower tail probability = $P\{X \leq k\}$

Upper tail probability = $P\{X > k\}$

Point probability = $P\{X = k\}$.

The computation approach used is as described in Knüsel [1].

6.3 Accuracy

The 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).

6.4 Timing

The time taken by the procedure depends on λ and k . For given λ , the time is greatest when $k \approx \lambda$, and is then approximately proportional to $\sqrt{\lambda}$.

Procedure: nag_hypergeo_prob

1 Description

nag_hypergeo_prob returns the lower tail, upper tail or point probability for a hypergeometric distribution with parameters n , l and m .

2 Usage

USE **nag_discrete_dist**

[*value* =] **nag_hypergeo_prob**(*tail*, *n*, *l*, *m*, *k* [, *optional arguments*])

The function result is a scalar of type real(kind=wp). In the case of a multi-precision Library the function result is a scalar of type real(kind=hp), where *hp* is the highest precision in the Library.

3 Arguments

3.1 Mandatory Arguments

tail — character(len=1), intent(in)

Input: the type of probability to be returned:

- if **tail** = 'L' or 'l', the lower tail probability is returned;
- if **tail** = 'U' or 'u', the upper tail probability is returned;
- if **tail** = 'A' or 'a', the probability of any point **k** is returned.

Constraints: **tail** = 'L', 'l', 'U', 'u', 'A' or 'a'.

n — integer, intent(in)

Input: the parameter n of the hypergeometric distribution.

Constraints: $n > 0$.

l — integer, intent(in)

Input: the parameter l of the hypergeometric distribution.

Constraints: $0 < l < n$.

m — integer, intent(in)

Input: the parameter m of the hypergeometric distribution.

Constraints: $0 < m < n$.

k — integer, intent(in)

Input: the number of successes that defines the required probability.

Constraints: $\max(0, l+m-n) \leq k \leq \min(l, m)$.

3.2 Optional Argument

error — type(nag_error), intent(inout), optional

The NAG *f*90 error-handling argument. See the Essential Introduction, or the module document **nag_error_handling** (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to **nag_set_error** before this procedure is called.

4 Error Codes

Fatal errors (error%level = 3):

error%code	Description
301	An input argument has an invalid value.

Failures (error%level = 2):

error%code	Description
201	n is too large to be represented exactly as a real(kind=wp) number.

5 Examples of Usage

Assume that all relevant arguments have been declared correctly as described in Section 3, and that input and input/output arguments have been appropriately initialized. This example shows how a call to this procedure returns the probability of point k from a hypergeometric distribution with parameters n , l and m .

```
prob = nag_hypergeo_prob('a', n, l, m, k)
```

To force the result to be of type real(kind=wp), we recommend using

```
prob = REAL(nag_hypergeo_prob('a', n, l, m, k), KIND=wp)
```

6 Further Comments

6.1 Mathematical Background

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

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

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

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 = lm/n , and the variance = $\frac{lm(n-l)(n-m)}{n^2(n-1)}$.

6.2 Algorithmic Detail

The procedure computes, for any given values of n , l , m and k , the following probabilities:

Lower tail probability = $P\{X \leq k\}$

Upper tail probability = $P\{X > k\}$

Point probability = $P\{X = k\}$.

The computation approach is similar to the one used for the Poisson distribution as described in Knüsel [1].

6.3 Accuracy

The results should be 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).

6.4 Timing

The time taken by the procedure depends on the variance = $\frac{lm(n-l)(n-m)}{n^2(n-1)}$ and on k . For any 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.

Example 1: Calculation of the probability of point k from a binomial distribution

This example program shows how `nag_binom_prob` returns the tail and point probabilities for a binomial distribution with parameters n and p .

1 Program Text

Note. The listing of the example program presented below is double precision. Single precision users are referred to Section 5.2 of the Essential Introduction for further information.

```

PROGRAM nag_discrete_dist_ex01

! Example Program Text for nag_discrete_dist
! NAG fl90, Release 3. NAG Copyright 1997.

! .. Use Statements ..
USE nag_examples_io, ONLY : nag_std_out, nag_std_in
USE nag_discrete_dist, ONLY : nag_binom_prob
! .. Implicit None Statement ..
IMPLICIT NONE
! .. Intrinsic Functions ..
INTRINSIC KIND
! .. Parameters ..
INTEGER, PARAMETER :: wp = KIND(1.0D0)
! .. Local Scalars ..
INTEGER :: k, n
REAL (wp) :: p, prob
CHARACTER (1) :: tail
! .. Executable Statements ..

WRITE (nag_std_out,*) &
'Example Program Results for nag_discrete_dist_ex01'

READ (nag_std_in,*)          ! Skip heading in data file

WRITE (nag_std_out,*)
WRITE (nag_std_out,*) 'tail    n        p        k        probability'
WRITE (nag_std_out,*)

DO

  READ (nag_std_in,*,end=20) tail, n, p, k

  prob = nag_binom_prob(tail,n,p,k)

  WRITE (nag_std_out,'(2X,A1,3x,I4,F8.3,I7,F15.4)') tail, n, p, k, prob

END DO
20  CONTINUE

END PROGRAM nag_discrete_dist_ex01

```

2 Program Data

Example Program Data for `nag_discrete_dist_ex01`

```

'U'    4    0.50    2      : tail, n, p, k
'L'   19    0.44   13
'A'   100    0.75   67
'L'  2000    0.33  700

```

3 Program Results

Example Program Results for nag_discrete_dist_ex01

tail	n	p	k	probability
U	4	0.500	2	0.3125
L	19	0.440	13	0.9914
A	100	0.750	67	0.0170
L	2000	0.330	700	0.9725

Additional Examples

Not all example programs supplied with NAG *f*/90 appear in full in this module document. The following additional examples, associated with this module, are available.

`nag_discrete_dist_ex02`

Calculation of the lower tail, upper tail or point probability for a Poisson distribution with known parameter.

`nag_discrete_dist_ex03`

Calculation of the lower tail, upper tail or point probability for a hypergeometric distribution with known parameters.

References

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