

# NAG Library Function Document

## nag\_prod\_limit\_surviv\_fn (g12aac)

### 1 Purpose

nag\_prod\_limit\_surviv\_fn (g12aac) computes the Kaplan–Meier, (or product-limit), estimates of survival probabilities for a sample of failure times.

### 2 Specification

```
#include <nag.h>
#include <nagg12.h>

void nag_prod_limit_surviv_fn (Integer n, const double t[], const Integer ic[],
    const Integer freq[], Integer *nd, double tp[], double p[], double psig[],
    NagError *fail)
```

### 3 Description

A survivor function,  $S(t)$ , is the probability of surviving to at least time  $t$  with  $S(t) = 1 - F(t)$ , where  $F(t)$  is the cumulative distribution function of the failure times. The Kaplan–Meier or product limit estimator provides an estimate of  $S(t)$ ,  $\hat{S}(t)$ , from sample of failure times which may be progressively right-censored.

Let  $t_i$ ,  $i = 1, 2, \dots, n_d$ , be the ordered distinct failure times for the sample of observed failure/censored times, and let the number of observations in the sample that have not failed by time  $t_i$  be  $n_i$ . If a failure and a loss (censored observation) occur at the same time  $t_i$ , then the failure is treated as if it had occurred slightly before time  $t_i$  and the loss as if it had occurred slightly after  $t_i$ .

The Kaplan–Meier estimate of the survival probabilities is a step function which in the interval  $t_i$  to  $t_{i+1}$  is given by

$$\hat{S}(t) = \prod_{j=1}^i \left( \frac{n_j - d_j}{n_j} \right)$$

where  $d_j$  is the number of failures occurring at time  $t_j$ .

nag\_prod\_limit\_surviv\_fn (g12aac) computes the Kaplan–Meier estimates and the corresponding estimates of the variances,  $\hat{\text{var}}(\hat{S}(t))$ , using Greenwood's formula,

$$\hat{\text{var}}(\hat{S}(t)) = \hat{S}(t)^2 \sum_{j=1}^i \frac{d_j}{n_j(n_j - d_j)}.$$

### 4 References

Gross A J and Clark V A (1975) *Survival Distributions: Reliability Applications in the Biomedical Sciences* Wiley

Kalbfleisch J D and Prentice R L (1980) *The Statistical Analysis of Failure Time Data* Wiley

### 5 Arguments

1: **n** – Integer *Input*

*On entry:* the number of failure and censored times given in **t**.

*Constraint:* **n**  $\geq$  2.

- 2: **t[n]** – const double *Input*  
*On entry:* the failure and censored times; these need not be ordered.
- 3: **ic[n]** – const Integer *Input*  
*On entry:* **ic**[*i* – 1] contains the censoring code of the *i*th observation, for *i* = 1, 2, ..., **n**.  
**ic**[*i* – 1] = 0  
The *i*th observation is a failure time.  
**ic**[*i* – 1] = 1  
The *i*th observation is right-censored.  
*Constraint:* **ic**[*i* – 1] = 0 or 1 for *i* = 1, 2, ..., **n**.
- 4: **freq[n]** – const Integer *Input*  
*On entry:* indicates whether frequencies are provided for each failure and censored time point. If frequencies are provided then **freq** must be dimensioned at least **n**. If the failure and censored times are to be considered as single observations, i.e., a frequency of 1 is to be assumed then **freq** must be set to **NULL**.  
*Constraint:* either **freq** = (Integer\*)0 or **freq**[*i* – 1] ≥ 0, for *i* = 1, 2, ..., **n**.
- 5: **nd** – Integer \* *Output*  
*On exit:* the number of distinct failure times,  $n_d$ .
- 6: **tp[n]** – double *Output*  
*On exit:* **tp**[*i* – 1] contains the *i*th ordered distinct failure time,  $t_i$ , for *i* = 1, 2, ...,  $n_d$ .
- 7: **p[n]** – double *Output*  
*On exit:* **p**[*i* – 1] contains the Kaplan–Meier estimate of the survival probability,  $\hat{S}(t)$ , for time **tp**[*i* – 1], for *i* = 1, 2, ...,  $n_d$ .
- 8: **psig[n]** – double *Output*  
*On exit:* **psig**[*i* – 1] contains an estimate of the standard deviation of **p**[*i* – 1], for *i* = 1, 2, ...,  $n_d$ .
- 9: **fail** – NagError \* *Input/Output*  
The NAG error argument (see Section 3.6 in the Essential Introduction).

## 6 Error Indicators and Warnings

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

### NE\_INT\_ARG\_LT

On entry, **n** =  $\langle value \rangle$ .  
Constraint: **n** ≥ 2.

### 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.

### NE\_INVALID\_CENSOR\_CODE

On entry, **ic**[ $\langle value \rangle$ ] =  $\langle value \rangle$ . The censor code for an observation must be either 0 or 1.

**NE\_INVALID\_FREQ**

On entry, `freq[⟨value⟩] = ⟨value⟩`. The value of frequency for an observation must be  $\geq 0$ .

**7 Accuracy**

The computations are believed to be stable.

**8 Further Comments**

If there are no censored observations,  $\hat{S}(t)$ , reduces to the ordinary binomial estimate of the probability of survival at time  $t$ .

**9 Example**

The remission times for a set of 21 leukemia patients at 18 distinct time points are read in and the Kaplan–Meier estimate computed and printed. For further details see page 242 of Gross and Clark (1975).

**9.1 Program Text**

```

/* nag_prod_limit_surviv_fn (g12aac) Example Program.
 *
 * Copyright 1996 Numerical Algorithms Group.
 *
 * Mark 4, 1996.
 * Mark 8 revised, 2004.
 *
 */

#include <nag.h>
#include <nagx04.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg12.h>

int main(int argc, char *argv[])
{
    FILE      *fpin, *fpout;

    Integer   exit_status = 0, i, *ic = 0, *ifreq = 0, n, nd;
    NagError  fail;
    double    *p = 0, *psig = 0, *t = 0, *tp = 0;

    INIT_FAIL(fail);

    /* Check for command-line IO options */
    fpin = nag_example_file_io(argc, argv, "-data", NULL);
    fpout = nag_example_file_io(argc, argv, "-results", NULL);

    fprintf(fpout, "nag_prod_limit_surviv_fn (g12aac) Example Program Results\n");

    /* Skip heading in data file */
    fscanf(fpin, "%*[\n] ");

    fscanf(fpin, "%ld ", &n);
    if (n >= 2)
    {
        if (!(psig = NAG_ALLOC(n, double)) ||
            !(p = NAG_ALLOC(n, double)) ||
            !(t = NAG_ALLOC(n, double)) ||
            !(tp = NAG_ALLOC(n, double)) ||
            !(ifreq = NAG_ALLOC(n, Integer)) ||
            !(ic = NAG_ALLOC(n, Integer)))
        {
            fprintf(fpout, "Allocation failure\n");
            exit_status = -1;
        }
    }
}

```

```

        goto END;
    }
}
else
{
    fprintf(fpout, "Invalid n.\n");
    exit_status = 1;
    return exit_status;
}

for (i = 0; i < n; ++i)
    fscanf(fpin, "%lf %ld %ld ", &t[i], &ic[i], &ifreq[i]);

/* nag_prod_limit_surviv_fn (g12aac).
 * Computes Kaplan-Meier (product-limit) estimates of
 * survival probabilities
 */
nag_prod_limit_surviv_fn(n, t, ic, ifreq, &nd, tp, p, psig, &fail);
if (fail.code != NE_NOERROR)
{
    fprintf(fpout, "Error from nag_prod_limit_surviv_fn (g12aac).\n%s\n",
            fail.message);
    exit_status = 1;
    goto END;
}

fprintf(fpout, "\n   Time      Survival      Standard\n");
fprintf(fpout, "           probability  deviation\n\n");
for (i = 0; i < nd; ++i)
    fprintf(fpout, " %6.1f%10.3f %10.3f\n", tp[i], p[i], psig[i]);

END:
if (fpin != stdin) fclose(fpin);
if (fpout != stdout) fclose(fpout);
if (psig) NAG_FREE(psig);
if (p) NAG_FREE(p);
if (t) NAG_FREE(t);
if (tp) NAG_FREE(tp);
if (ifreq) NAG_FREE(ifreq);
if (ic) NAG_FREE(ic);
return exit_status;
}

```

## 9.2 Program Data

nag\_prod\_limit\_surviv\_fn (g12aac) Example Program Data  
18  
6.0 1 1 6.0 0 3 7.0 0 1 9.0 1 1 10.0 0 1 10.0 1 1  
11.0 1 1 13.0 0 1 16.0 0 1 17.0 1 1 19.0 1 1 20.0 1 1  
22.0 0 1 23.0 0 1 25.0 1 1 32.0 1 2 34.0 1 1 35.0 1 1

## 9.3 Program Results

nag\_prod\_limit\_surviv\_fn (g12aac) Example Program Results

Time	Survival probability	Standard deviation
6.0	0.857	0.076
7.0	0.807	0.087
10.0	0.753	0.096
13.0	0.690	0.107
16.0	0.627	0.114
22.0	0.538	0.128
23.0	0.448	0.135