NAG C Library Function Document

nag_surviv_cox_model (g12bac)

1 Purpose

nag_surviv_cox_model (g12bac) returns parameter estimates and other statistics that are associated with the Cox proportional hazards model for fixed covariates.

2 Specification

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

void nag_surviv_cox_model (Integer n, Integer m, Integer ns,
          const double z[], Integer tdz, const Integer sz[], Integer ip,
          const double t[], const Integer ic[], const double omega[],
          const Integer isi[], double *dev, double b[], double se[],
          double sc[], double cov[], double res[], Integer *nd, double tp[],
          double sur[], Integer tdsur, Integer ndmax, double tol,
          Integer max_iter, Integer iprint, const char *outfile,
          NagError *fail)
```

3 Description

The proportional hazard model relates the time to an event, usually death or failure, to a number of explanatory variables known as covariates. Some of the observations may be right censored, that is the exact time to failure is not known, only that it is greater than a known time.

Let t_i, for $i = 1, \ldots, n$ be the the failure time or censored time for the ith observation with the vector of p covariates z_i. It is assumed that censoring and failure mechanisms are independent. The hazard function, $\lambda(t, z)$, is the probability that an individual with covariates z fails at time t given that the individual survived up to time t. In the Cox proportional hazards model (Cox (1972b)) $\lambda(t, z)$ is of the form:

$$\lambda(t, z) = \lambda_0(t)\exp(z^T\beta + \omega)$$

where λ_0 is the base-line hazard function, an unspecified function of time, β is a vector of unknown parameters and ω is a known offset.

Assuming there are ties in the failure times giving $n_d < n$ distinct failure times, $t_{(1)} < \ldots < t_{(n_d)}$ such that d_i individuals fail at $t_{(i)}$, it follows that the marginal likelihood for β is well approximated (see Kalbfleisch and Prentice (1980)) by:

$$L = \prod_{i=1}^{n_d} \frac{\exp(s_i^T\beta + \omega_i)}{[\sum_{l\in R(t_{(i)})} \exp(z_l^T\beta + \omega_l)]^{d_i}} \tag{1}$$

where s_i is the sum of the covariates of individuals observed to fail at $t_{(i)}$ and $R(t_{(i)})$ is the set of individuals at risk just prior to $t_{(i)}$, that is it is all individuals that fail or are censored at time $t_{(i)}$ along with all individuals that survive beyond time $t_{(i)}$. The maximum likelihood estimates (MLEs) of β, given by $\hat{\beta}$, are obtained by maximizing (1) using a Newton–Raphson iteration technique that includes step halving and utilizes the first and second partial derivatives of (1) which are given by equations (2) and (3) below:

$$U_j(\beta) = \frac{\partial \ln L}{\partial \beta_j} = \sum_{i=1}^{n_d}[s_{ji} - d_i\alpha_{ji}(\beta)] = 0 \tag{2}$$

for $j = 1, \ldots, p$, where s_{ji} is the jth element in the vector s_i and

$$\alpha_{ji}(\beta) = \frac{\sum_{l\in R(t_{(i)})} z_{jl}\exp(z_l^T\beta + \omega_l)}{\sum_{l\in R(t_{(i)})} \exp(z_l^T\beta + \omega_l)}.$$

Similarly,

$$I_{hj}(\beta) = -\frac{\partial^2 \ln L}{\partial\beta_h \partial\beta_j} = \sum_{i=1}^{n_d} d_i \gamma_{hji} \tag{3}$$

where

$$\gamma_{hji} = \frac{\sum_{l \in R(t_{(i)})} z_{hl} z_{jl} \exp(z_l^T \beta + \omega_l)}{\sum_{l \in R(t_{(i)})} \exp(z_l^T \beta + \omega_l)} - \alpha_{hi}(\beta)\alpha_{ji}(\beta) \qquad h, j = 1, \ldots, p.$$

$U_j(\beta)$ is the jth component of a score vector and $I_{hj}(\beta)$ is the (h, j) element of the observed information matrix $I(\beta)$ whose inverse $I(\beta)^{-1} = [I_{hj}(\beta)]^{-1}$ gives the variance-covariance matrix of β.

It should be noted that if a covariate or a linear combination of covariates is monotonically increasing or decreasing with time then one or more of the β_j's will be infinite.

If $\lambda_0(t)$ varies across ν strata, where the number of individuals in the kth stratum is n_k, $k = 1, \ldots, \nu$ with $n = \sum_{k=1}^{\nu} n_k$, then rather than maximizing (1) to obtain $\hat{\beta}$, the following marginal likelihood is maximized:

$$L = \prod_{k=1}^{\nu} L_k, \tag{4}$$

where L_k is the contribution to likelihood for the n_k observations in the kth stratum treated as a single sample in (1). When strata are included the covariate coefficients are constant across strata but there is a different base-line hazard function λ_0.

The base-line survivor function associated with a failure time $t_{(i)}$, is estimated as $\exp(-\hat{H}(t_{(i)}))$, where

$$\hat{H}(t_{(i)}) = \sum_{t_{(j)} \leq t_{(i)}} \left(\frac{d_i}{\sum_{l \in R(t_{(j)})} \exp(z_l^T \hat{\beta} + \omega_l)} \right), \tag{5}$$

where d_i is the number of failures at time $t_{(i)}$. The residual for the lth observation is computed as:

$$r(t_l) = \hat{H}(t_l)\exp(-z_l^T \hat{\beta} + \omega_l)$$

where $\hat{H}(t_l) = \hat{H}(t_{(i)}), t_{(i)} \leq t_l < t_{(i+1)}$. The deviance is defined as $-2\times$(logarithm of marginal likelihood). There are two ways to test whether individual covariates are significant: the differences between the deviances of nested models can be compared with the appropriate χ^2-distribution; or, the asymptotic normality of the parameter estimates can be used to form z tests by dividing the estimates by their standard errors or the score function for the model under the null hypothesis can be used to form z tests.

4 Parameters

1: **n** – Integer *Input*

On entry: the number of data points, n.

Constraint: **n** ≥ 2.

2: **m** – Integer *Input*

On entry: the number of covariates in array **z**.

Constraint: **m** ≥ 1.

3: **ns** – Integer *Input*

On entry: the number of strata. If **ns** > 0 then the stratum for each observation must be supplied in **isi**.

Constraint: **ns** ≥ 0.

4: **z[n][tdz]** – const double *Input*

On entry: the ith row must contain the covariates which are associated with the ith failure time given in **t**.

5: **tdz** – Integer *Input*

On entry: the second dimension of the array **z** as declared in the function from which nag_surviv_cox_model is called.

Constraint: **tdz** $\geq$ **m**.

6: **sz[m]** – const Integer *Input*

On entry: indicates which subset of covariates is to be included in the model.

If **sz**$[i-1] \geq 1$, the jth covariate is included in the model.

If **sz**$[i-1] = 0$, the jth covariate is excluded from the model and not referenced.

Constraints: **sz**$[j-1] \geq 0$ and at least one and at most $n_0 - 1$ elements of **sz** must be non-zero where n_0 is the number of observations excluding any with zero value of **isi**.

7: **ip** – Integer *Input*

On entry: the number of covariates included in the model as indicated by **sz**.

Constraint: **ip** = number of non-zero values of **sz**.

8: **t[n]** – const double *Input*

On entry: the vector of n failure censoring times.

9: **ic[n]** – const Integer *Input*

On entry: the status of the individual at time t given in **t**.

ic$[i-1] = 0$, indicates that the ith individual has failed at time **t**$[i-1]$.

ic$[i-1] = 1$, indicates that the ith individual has been censored at time **t**$[i-1]$.

Constraint: **ic**$[i-1] = 0$ or 1, for $i = 1, 2, \ldots,$**n**.

10: **omega[n]** – const double *Input*

On entry: if an offset is required then **omega** must contain the value of ω_i, for $i = 1, 2, \ldots,$**n**. Otherwise **omega** must be set as the null pointer, (double*0).

11: **isi[*dim1*]** – const Integer *Input*

Note: the dimension, *dim1*, of the array **isi** must be at least **n** if **ns** > 0.

On entry: if **ns** > 0 the stratum indicators which also allow data points to be excluded from the analysis. If **ns** $= 0$, **isi** is not referenced and can be set to null, (Integer*) 0.

isi$[i-1]$ = k indicates that the ith data point is in the kth stratum, where $k = 1, 2, \ldots,$**ns**.

isi$[i-1] = 0$ indicates that the ith data point is omitted from the analysis.

Constraints: if **ns** > 0, $0 \leq$ **isi**$[i-1] \leq$ **ns** for $i = 1, 2, \ldots,$**n**, and more than **ip** values of **isi**$[i-1]$ > 0.

12: **dev** – double * *Output*

On exit: the deviance, that is $-2\times$(maximized log marginal likelihood).

13: **b[ip]** – double *Input/Output*

On entry: initial estimates of the covariate coefficient parameters β. **b**$[j-1]$ must contain the initial estimate of the coefficient of the covariate in **z** corresponding to the jth non-zero value of **sz**.

Suggested values: In many cases an initial value of zero for **b**$[j-1]$ may be used. For other suggestions see Section 6.

On exit: **b**$[j-1]$ contains the estimate $\hat{\beta}_i$, the coefficient of the covariate stored in the ith column of **z** where i is the jth non-zero value in the array **sz**.

14: **se[ip]** – double *Output*

On exit: **se**$[j-1]$ is the asymptotic standard error of the estimate contained in **b**$[j-1]$ and score function in **sc**$[j-1]$ for $j = 1, 2, \ldots,$**ip**.

15: **sc[ip]** – double *Output*

On exit: **sc**$[j-1]$ is the value of the score function, $U_j(\beta)$, for the estimate contained in **b**$[j-1]$.

16: **cov[ip*(ip+1)2]** – double *Output*

On exit: the variance-covariance matrix of the parameter estimates in **b** stored in packed form by column, i.e., the covariance between the parameter estimates given in **b**$[i-1]$ and **b**$[j-1]$, $j \geq i$, is stored in **cov**$(j(j-1)/2+i)$.

17: **res[n]** – double *Output*

On exit: the residuals, $r(t_l)$, $l = 1, 2, \ldots,$**n**.

18: **nd** – Integer * *Output*

On exit: the number of distinct failure times.

19: **tp[ndmax]** – double *Output*

On exit: **tp**$[i-1]$ contains the ith distinct failure time for $i = 1, 2, \ldots,$**nd**.

20: **sur[ndmax][tdsur]** – double *Output*

Note: the second dimension of the array **sur** must be at least max(**ns**,1).

On exit:

if **ns** $= 0$, **sur**$(i, 1)$ contains the estimated survival function for the ith distinct failure time,

if **ns** > 0, **sur**(i, k) contains the estimated survival function for the ith distinct failure time in the kth stratum.

21: **tdsur** – Integer *Input*

On entry: the second dimension of the array **sur** as declared in the function from which nag_surviv_cox_model is called.

Constraint: **tdsur** $\geq$ max(**ns**,1).

22: **ndmax** – Integer *Input*

On entry: the first dimension of the array **sur** as declared in the function from which nag_surviv_cox_model is called.

Constraint: **ndmax** $\geq$ the number of distinct failure times. This is returned in **nd**.

23: **tol** – double *Input*

On entry: indicates the accuracy required for the estimation. Convergence is assumed when the decrease in deviance is less than **tol**×(1.0+CurrentDeviance). This corresponds approximately to an absolute precision if the deviance is small and a relative precision if the deviance is large.

Constraint: **tol** $\geq$ 10×***machine precision***.

24: **max_iter** – Integer *Input*

On entry: the maximum number of iterations to be used for computing the estimates. If **max_iter** is set to 0 then the standard errors, score functions, variance-covariance matrix and the survival function are computed for the input value of β in **b** but β is not updated.

Constraint: **max_iter** $\geq$ 0.

25: **iprint** – Integer *Input*

On entry: indicates if the printing of information on the iterations is required. If **iprint** $\leq$ 0, there is no printing, if **iprint** $\geq$ 1 then the deviance and the current estimates are printed every **iprint** iterations.

26: **outfile** – const char * *Input*

On entry: the name of the file into which imformation is to be output. If **outfile** is set to NULL or to the string 'stdout', then the monitoring information is output to stdout,

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

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

NE_INT_ARG_LT

On entry, **m** must not be less than 1: **m** = <*value*>.

On entry, **n** must not be less than 2: **n** = <*value*>.

On entry, **ns** must not be less than 0: **ns** = <*value*>.

On entry, **tdsur** must not be less than 1: **tdsur** = <*value*>.

On entry, **max_iter** must not be less than 0: **max_iter** = <*value*>.

NE_2_INT_ARG_LT

On entry, **tdsur** = <*value*> while **ns** = <*value*>.
These parameters must satisfy **tdsur** $\geq$ **ns**.

On entry, **tdz** = <*value*> while **m** = <*value*>.
These parameters must satisfy **tdz** $\geq$ **m**.

NE_REAL_MACH_PREC

On entry, **tol** = <*value*>, ***machine precision***(X02AJC) = <*value*>.
Constraint: **tol** $\geq$ 10.0×***machine precision***.

NE_NOT_APPEND_FILE

Cannot open file **outfile** for appending.

NE_NOT_CLOSE_FILE

Cannot close file **outfile**.

NE_INT_ARRAY_CONS

On entry, **sz**[<*value*>] = <*value*>.
Constraint: **sz**[<*value*>] $\geq$ 0.

On entry, **ic**[<*value*>] = <*value*>.
Constraint: **ic**[<*value*>] = 0 or 1.

On entry, **isi**[<*value*>] $=$ <*value*>.
Constraint: $0 \leq$ **isi**[<*value*>] $\leq$ **ns.**

NE_G12BA_SZ_IP

On entry, **ip** $=$ <*value*> and the number of non zero values of **sz** $=$ <*value*>.
Constraint: **ip** $=$ the number of non-zero values of **sz**.

NE_ARRAY_CONS

The contents of array **ic** are not valid.
Constraint: not all values of **ic** can be 1.

NE_G12BA_SZ_ISI

On entry, the number of values of **sz**$[i] > 0$ is <*value*>, **n** $=$ <*value*> and excluded observations with **isi**$[i] = 0$ is <*value*>.
Constraint: the number of values of non-zero **sz** must be less than **n** $-$ excluded observations.

NE_G12BA_NDMAX

On entry **ndmax** is $=$ <*value*> while the output value of **nd** $=$ <*value*>.
Constraint: **ndmax** $\geq$ **nd**.

NE_G12BA_MAT_SING

The matrix of second partial derivatives is singular. Try different starting values or include fewer covariates.

NE_G12BA_OVERFLOW

Overflow has been detected. Try different starting values.

NE_G12BA_CONV

Convergence has not been achieved in **max_iter** iterations. The progress towards convergence can be examined by using by setting **iprint** to ≥ 1. Any non-covergence maybe due to a linear combination of covariates being monotonic with time. Full results are returned.

NE_G12BA_DEV

In the current iteration 10 step halvings have been performed without decreasing the deviance from the previous iteration. Convergence is assumed.

NE_ALLOC_FAIL

Memory allocation failed.

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 consult NAG for assistance.

6 Further Comments

The routine uses mean centering which involves subtracting the means from the covariables prior to computation of any statistics. This helps to minimize the effect of outlying observations and accelerates convergence.

If the initial estimates are poor then there may be a problem with overflow in calculating $\exp(\beta^T z_i)$ or there may be non-convergence. Reasonable estimates can often be obtained by fitting an exponential model using nag_glm_poisson (g02gcc).

6.1 Accuracy

The accuracy is specified by **tol**.

6.2 References

Cox D R (1972b) Regression models in life tables (with discussion) *J. Roy. Statist. Soc. Ser. B* **34** 187–220

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

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

7 See Also

nag_glm_poisson (g02gcc)

8 Example

The data are the remission times for two groups of leukemia patients (see Gross and Clark (1975) p242). A dummy variable indicates which group they come from. An initial estimate is computed using the exponential model and then the Cox proportional hazard model is fitted and parameter estimates and the survival function are printed.

8.1 Program Text

```
/* nag_surviv_cox_model (g12bac) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * NAG C Library
 */

#include <stdio.h>
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg02.h>
#include <nagg12.h>

/* Table of constant values */

static double c_b42 = 0.;
static Integer c__0 = 0;

int main (void)
{
  double *b=0, *cov=0, dev, df, *omega, *res=0, *sc=0, *se=0, *sur=0, *t=0;
  double tol, *tp=0, *v=0, *y=0, *z=0, *offset=0;
  Integer i, i__1, *ic=0, ip, ip1, iprint, irank, *isi=0, *sz=0;
  Integer j, m, maxit, n, nd, ndmax, ns, tdsur, tdv;
  Integer exit_status=0;
  NagError fail;

#define Z(I,J) z[((I)-1)*m +(J)-1]

  INIT_FAIL(fail);
  Vprintf("g12bac Example Program Results\n");

  /*  Skip heading in data file */
 Vscanf("%*[^\n]");

 Vscanf("%ld %ld %ld %ld %ld ", &n, &m, &ns, &maxit, &iprint);
 ndmax = 42;
 tdsur = MAX(1,ns);
 if (!(z = NAG_ALLOC(n*m, double))
     || !(sz = NAG_ALLOC(m, Integer))
     || !(t = NAG_ALLOC(n, double))
     || !(ic = NAG_ALLOC(n, Integer))
     || !(omega = NAG_ALLOC(n, double))
     || !(isi = NAG_ALLOC(n, Integer))
     || !(res = NAG_ALLOC(n, double))
     || !(sur = NAG_ALLOC(ndmax*tdsur, double))
     || !(tp = NAG_ALLOC(ndmax, double)))
   {
     Vprintf("Allocation failure\n");
     exit_status = -1;
     goto END;
   }

 if (ns > 0)
   {
     for (i = 1; i <= n; ++i)
{
  Vscanf("%lf", &t[i - 1]);
  for (j = 1; j <= m; ++j)
    Vscanf("%lf", &Z(i,j));
  Vscanf("%ld", &ic[i - 1]);
  Vscanf("%ld", &isi[i - 1]);
}
   }
 else
   {
     for (i = 1; i <= n; ++i)
{
  Vscanf("%lf", &t[i - 1]);
  for (j = 1; j <= m; ++j)
    Vscanf("%lf", &Z(i,j));
  Vscanf("%ld", &ic[i - 1]);
}
   }
 for (i = 1; i <= m; ++i)
   Vscanf("%ld", &sz[i - 1]);
 Vscanf("%ld", &ip);
 ip1 = ip +1;
 if (!(b = NAG_ALLOC(ip1, double))
     || !(se = NAG_ALLOC(ip1, double))
     || !(sc = NAG_ALLOC(ip1, double))
     || !(cov = NAG_ALLOC(ip1*(ip1+1)/2, double))
     || !(tdv =  ip1+6)
     || !(v = NAG_ALLOC(n*tdv, double))
     || !(y = NAG_ALLOC(n, double))
     || !(offset = NAG_ALLOC(n, double)))
   {
     Vprintf("Allocation failure\n");
     exit_status = -1;
     goto END;
   }
  tol = 5e-5;
  i__1 = n;
  for (i = 1; i <= n; ++i)
    {
      y[i - 1] = 1.0 - (double) ic[i - 1];
      v[i + 251] = log (t[i - 1]);
      offset[i-1] = log (t[i - 1]);
    }
  g02gcc(Nag_Log, Nag_MeanInclude, n, z, m, m, sz, ip1, y, 0, offset,
  c_b42, &dev, &df, b, &irank, se, cov, v, tdv, tol,
  maxit, c__0, 0, c_b42,  &fail);
  if (fail.code != NE_NOERROR)
    {
      Vprintf("Error from g02gcc.\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
  for (i = 1; i <= ip; ++i)
    b[i - 1] = b[i];
  if (irank != ip + 1)
    Vprintf("%s\n", " WARNING: covariates not of full rank");

  g12bac(n, m, ns, z, m, sz, ip, t, ic, (double *)0,
  isi, &dev, b, se, sc, cov, res, &nd, tp, sur, tdsur, ndmax, tol,
  maxit, iprint, "", &fail);
  if (fail.code != NE_NOERROR)
    {
      Vprintf("Error from g12bac.\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }

  Vprintf("\n");
  Vprintf("%s", " Parameter      Estimate");
  Vprintf("%s\n", "       Standard Error");
  Vprintf("\n");
  for (i = 1; i <= ip; ++i)
    Vprintf("%6ld          %8.4f          %8.4f\n",
     i, b[i - 1], se[i - 1]);
  Vprintf("\n");
  Vprintf("%s%13.4e\n", " Deviance = ", dev);
  Vprintf("\n");
  Vprintf("%s\n", "    Time     Survivor Function");
  Vprintf("\n");
  ns = MAX(ns, 1);
  for (i = 1; i <= nd; ++i)
    {
      Vprintf("%10.0f", tp[i - 1]);
      for (j = 1; j <= ns; ++j)
 Vprintf("     %8.4f%s", sur[(i-1)*tdsur + j-1],
  j%3?"":"\n");
      Vprintf("\n");
    }
 END:
  if (z) NAG_FREE(z);
  if (sz) NAG_FREE(sz);
  if (t) NAG_FREE(t);
  if (ic) NAG_FREE(ic);
  if (omega) NAG_FREE(omega);
  if (isi) NAG_FREE(isi);
  if (res) NAG_FREE(res);
  if (sur) NAG_FREE(sur);
  if (tp) NAG_FREE(tp);
  if (b) NAG_FREE(b);
  if (se) NAG_FREE(se);
  if (sc) NAG_FREE(sc);
  if (cov) NAG_FREE(cov);
  if (v) NAG_FREE(v);
  if (y) NAG_FREE(y);
  if (offset) NAG_FREE(offset);
  return exit_status;
}
```

8.2 Program Data

```
g12bac Example Program Data

42 1 0 20 0

 1 0 0
 1 0 0
 2 0 0
 2 0 0
 3 0 0
 4 0 0
 4 0 0
 5 0 0
 5 0 0
 8 0 0
 8 0 0
 8 0 0
 8 0 0
11 0 0
11 0 0
12 0 0
12 0 0
15 0 0
17 0 0
22 0 0
23 0 0
 6 1 0
 6 1 0
 6 1 0
 7 1 0
10 1 0
13 1 0
16 1 0
22 1 0
23 1 0
 6 1 1
 9 1 1
10 1 1
11 1 1
17 1 1
19 1 1
20 1 1
25 1 1
32 1 1
```

```
32 1 1
34 1 1
35 1 1
   1   1
```

8.3 Program Results

```
g12bac Example Program Results

 Parameter      Estimate       Standard Error

     1           -1.5091            0.4096

 Deviance =    1.7276e+02

    Time     Survivor Function

         1       0.9640
         2       0.9264
         3       0.9065
         4       0.8661
         5       0.8235
         6       0.7566
         7       0.7343
         8       0.6506
        10       0.6241
        11       0.5724
        12       0.5135
        13       0.4784
        15       0.4447
        16       0.4078
        17       0.3727
        22       0.2859
        23       0.1908
```