TR3/12

Variable Selection in a Cox Proportional Hazards Model

July 9, 2012

The Numerical Algorithms Group Ltd

Web Site: www.nag.co.uk
www.nag.com
www.nag-j.co.jp
General Inquires: info@nag.co.uk
Technical Support: support@nag.co.uk

Contents

1 Introduction

The Cox 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 $i = 1, 2, \ldots, n$, be the failure time or censored time for the ith observation with the vector of m 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 [1] $\lambda(t, z)$ is of the form:

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

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.

The NAG routine for fitting a Cox proportional hazards model is G12BAF if you are using the Fortran library and g12bac if you are using the C library. In this article we will show how to use these routines to perform the three main approaches for automatic variable selection, that is, choosing which explanatory variables to include in the model. The three approaches described are; forward selection, backward selection and stepwise selection.

When discussing the NAG routines used in these analyses we concentrate on the Fortran library, however example programs and code snippets have been provided for both libraries.

2 Forward Selection

The forward selection process can be described as follows:

1. Start with the null model (that is a model with no explanatory variables),
2. Calculate S_i, a score for each variable not in the model, adjusted for all variables already in the model,
3. Find j such that $S_j \geq S_i$ for all $i, i \neq j$, i.e. we find the variable (not in the model) with the largest value of S. We denote the variable associated with the jth score, z_j,
4. Calculate p, the p-value associated with the S_j,
5. If $p > p_a$ then go to step 8,
6. Add variable z_j to the model.
7. If there are still variables not in the model then go to step 2,
8. Stop.

In cases where, at step 3, there are two or more variables with the highest score, one is chosen. This choice is arbitary.

In order to perform a forward selection procedure one must therefore choose a scoring statistic, S, and a cut-off p_a. When performing a forward selection on a Cox proportional hazards model one well known statistical package uses the Score test statistic for S and has a default value of 0.05 for p_a.

2.1 Score Test Statistic

The Score test statistic, S, is given by:

$$S(\beta_0) = \left(\frac{\partial \ln L}{\partial \beta}(\beta_0)\right)^T \left(\frac{\partial^2 \ln L}{\partial \beta^2}(\beta_0)\right)^{-1} \left(\frac{\partial \ln L}{\partial \beta}(\beta_0)\right)$$

where $\ln L$ is the log-likelihood function. S can then be used to test a hypothesis of the form:

$$H_0 : \beta = \beta_0 \text{ vs } H_1 : \beta \neq \beta_0$$

Under the null hypothesis, H_0, $S(\beta_0)$ has a χ^2_ν-distribution where ν is the number of variables being tested.

2.2 Calculating the Score Test

G12BAF has a number of output parameters, including the variance-covariance matrix, Σ, associated with the estimated values of β, as returned in COV, and the value of the score function, $U(\beta)$, as returned in SC. It should be noted that, although they share a common name, the score function U is not the Score test statistic, S, that we will use during the forward selection process.

Whilst being primarily designed to estimate the covariate coefficient parameters, β, G12BAF can be used to calculate the values of the other output parameters for a given value of β by setting the number of iterations, MAXIT, to zero. It is this feature that we utilize to calculate the Score test statistic.

The score function, returned by G12BAF is given by

$$U = U(\boldsymbol{\beta}) = \frac{\partial \ln L}{\partial \beta}$$

and the covariance matrix by

$$\begin{aligned} \Sigma &= I(\beta)^{-1}, \\ I(\boldsymbol{\beta}) &= -\frac{\partial^2 \ln L}{\partial \beta^2} \end{aligned}$$

therefore the score test statistic, S is given by

$$S = U^T \Sigma U$$

which can be calculated using the following code snippets:

Fortran:
```
! F06PEF: calculate COV * SC
Allocate (covsc(ip))
Call dspmv('Upper',ip,1.0_wp,cov,sc,1,0.0_wp,covsc,1)

! F06EAF: calculate transpose(SC) * COV * SC
! which gives the Score test statistic, S
s = ddot(ip,sc,1,covsc,1)
```

C:
```
/* f16pec: calculate cov * sc,
   using the default error structure, which will terminate if an error
   occurs as we should only ever be supplying valid input arguments, so
   routine should not fail */
covsc = NAG_ALLOC(ip, double);
nag_dspmv(Nag_ColMajor,Nag_Upper,ip,1.0,cov,sc,1,0.0,covsc,1,
          NAGERR_DEFAULT);

/* calculate transpose(sc) * cov * sc, which gives the
   Score test statistic */
for (i = 0, s = 0.0; i < ip; i++) s+= sc[i] * covsc[i];
```

where `ip` is the number of variables in the model. The p-value associated with S can be obtained using:

Fortran:
```
p = g01ecf('Upper',s,df,ifail)
```

C:
```
p = g01ecc(Nag_UpperTail,s,df,&fail);
```

where `df` is the degrees of freedom associated with the Score test statistic, ν.

Given a model that currently contains m parameters, the Score test statistic tends to be used to test two types of hypothesis;

1. $\beta_i = 0$ for $i = 1, 2, \ldots, m$, usually referred to as the global hypothesis as it tests whether all the parameters in the model are zero simultaneously. When testing the global hypothesis $\nu = m$.
2. $\beta_j = 0$, given $\beta_i = \hat{\beta}_i$, for $i \neq j$. This tests that one parameter is zero, given the value of the other parameters and therefore $\nu = 1$. This is the hypothesis used when calculating the p-value in step 4 of the forward selection process.

3 Backward Selection

The backward selection process is:

1. Start with the full model (that is a model containing all the explanatory variables),
2. Calculate W_i, a score for each variable in the model, adjusted for all other variables in the model,
3. Find k such that $W_k \leq W_i$ for all $i, i \neq k$, i.e. we find the variable (in the model) with the smallest value of W. We denote the variable associated with the kth score, z_k,
4. Calculate p, the p-value associated with W_i,
5. If $p < p_d$ then go to step 8,
6. Drop variable z_k from the model,
7. If there are still variables in the model then go to step 2,
8. Stop.

In cases where, at step 3, there are two or more variables with the lowest score, one is chosen. This choice is arbitary.

In order to perform a backward selection procedure one must therefore choose a scoring statistic, W, and a cut-off p_d. When performing a backward selection on a Cox proportional hazards model one well known statistical package uses the Wald test statisic for W and has a default value for p_d of 0.05.

3.1 Wald Test Statistic

The Wald test statistic, W, is given by:

$$W = \left(\hat{\beta} - \beta_0\right)^T \left(\frac{\partial^2 \ln L}{\partial \beta^2}\left(\hat{\beta}\right)\right)\left(\hat{\beta} - \beta_0\right)$$

where, $\hat{\beta}$ are the maximum likelihood estimates of the model parameters, β and $\ln L$ is the log-likelihood function. The test statistic W can then be used to test a hypothesis of the form:

$$H_0 : \beta = \beta_0 \text{ vs } H_1 : \beta \neq \beta_0$$

Under the null hypothesis, W has a χ^2_ν-distribution where ν is the number of variables being tested.

3.2 Calculating the Wald Test Statistic

When calculating the Wald test statistic we again make use of the fact that G12BAF can return the covariance matrix, Σ, for a given vector of parameter estimates, $\hat{\beta}$. The Wald test statistic is therefore given by

$$W = \left(\hat{\beta} - \beta_0\right)^T \Sigma^{-1} \left(\hat{\beta} - \beta_0\right)$$

Rather than inverting the covariance matrix, Σ directly, we use a Cholesky decomposition to obtain the upper triangular matrix, U, such that $\Sigma = U^T U$ and then solve the system of equations $Ux = \hat{\beta} - \beta_0$ for x, before finally calculating $W = x^T x$.

As Σ is a covariance matrix, it is positive semidefinite, although in the majority of cases it will be positive definite. The standard Cholesky factorization only works on a positive definite matrix. The Fortran library has a routine, DPSTRF, which performs a Cholesky factorization with complete pivoting, allowing the (rare) semidefinite cases to be handled. Unfortunately, as of Mark 23, the C library does not have such a routine, although it should be included at Mark 24. Therefore the code for calculating the Wald test statistic differs somewhat between the Fortran and C.

In the Fortran code we decompose the covariance matrix using F07KDF. Because G12BAF stores the covariance matrix in packaged storage, where as DPSTRF requires the matrix in unpacked storage, we first must unpack the matrix

```
! copy COV from packed format to upper triangular format
Allocate (ccov(ip,ip))
k = 0
Do j = 1, ip
  Do i = 1, j
    k = k + 1
    ccov(i,j) = cov(k)
  End Do
End Do
```

before calling the factorization routine

```
! use default tolerance in F07KDF
tol = 0.0_wp

! F07KDF: factorize COV so that COV = transpose(U) * U, where
! CCOV = COV on entry and U on exit
Allocate (work(2*ip),piv(ip))
Call dpstrf('Upper',ip,ccov,ip,piv,rank,tol,work,info)
```

where `ip` is the number of covariates in the current model. In the C code we use nag_dpptrf which performs the Cholesky factorization on a positive definite matrix in packed storage, so this time we do not need to unpack the covariance matrix. We do still need to copy it as nag_dpptrf overwrites the input matrix.

```
/* copy cov */
lcov = ip*(ip+1)/2;
ccov = NAG_ALLOC(lcov, double);
for (i = 0; i < lcov; i++) ccov[i] = cov[i];

/* f07gdc: factorize cov so that cov = transpose(U) * U, where ccov = cov
   on entry and U on exit */
nag_dpptrf(Nag_ColMajor,Nag_Upper,ip,ccov,NAGERR_DEFAULT);
```

Because we are using the default NAG error structure, `NAGERR_DEFAULT` nag_dpptrf will terminate if Σ is semidefinite.

Once the factorization has been performed we use a back solver to obtain $x = U^{-T}\left(\hat{\beta} - \beta_0\right)$:

Fortran:

```
! pivot B into PB
Allocate (pb(ip))
Do i = 1, ip
  pb(i) = b(piv(i))
End Do

! F06YJF: solve CCOV x = PB for x, putting the result in PB
Call dtrsm('Left','Upper','Transpose','NonUnit',rank,1,1.0_wp,ccov,ip, &
           pb,ip)
```

C:

```
/* copy b */
cb = NAG_ALLOC(ip, double);
for (i = 0; i < ip; i++) cb[i] = b[i];

/* f16plc: solve ccov * x = cb for x, putting the result in cb */
nag_dtpsv(Nag_ColMajor,Nag_Upper,Nag_Trans,Nag_NonUnitDiag,ip,1.0,ccov,cb,
          1,NAGERR_DEFAULT);
```

As can be seen above, in the Fortran code we have to take into account the pivotting performed by DPSTRF when copying the parameter estimates, `b`. The copy is required in both cases as both back solvers, DTRSM and nag_dtpsv overwrite `b`, so that `b` holds $\left(\hat{\beta} - \beta_0\right)$ on entry and x on exit.

Finally W is calculated:

Fortran:

```
w = ddot(rank,pb,1,pb,1)
```

C:

```
for (i = 0, w = 0.0; i < ip; i++) w+= cb[i] * cb[i];
```

The C library does not have a documented equivalent of DDOT, which performs the dot product of two vectors, therefore in the C code snippet we need to use a for loop. The p-value associated with W can be obtained using:

Fortran:

```
p = g01ecf('Upper',s,df,ifail)
```

C:

```
p = g01ecc(Nag_UpperTail,s,df,&fail);
```

where `df` is the degrees of freedom associated with the Wald test statistic, ν.

Given a model that currently contains m parameters, the Wald test statistic tends to be used

to test two types of hypothesis;

1. $\beta_i = 0$ for $i = 1, 2, \ldots, m$, usually referred to as the global hypothesis as it tests whether all the parameters in the model are zero simultaneously. When testing the global hypothesis $\nu = m$.
2. $\beta_j = 0$, given $\beta_i = \hat{\beta}_i$, for $i \neq j$. This tests that one parameter is zero, given the value of the other parameters and therefore $\nu = 1$. This is the hypothesis used when calculating the p-value in step 4 of the backward selection process. In this case, the Wald test statistic simplifies to

$$W = \frac{\hat{\beta}_i^2}{\sigma_{ii}}$$

where σ_{ii} is the (i, i)th element if Σ.

4 Stepwise Selection

The last variable selection process we consider is stepwise selection. Stepwise selection can be considered as an amalgamation of the forward and backward processes as at each iteration a forward selection is performed, followed by a backward elimination. The process can be summarised as:

1. Start with the null model (that is a model with no explanatory variables),
2. Calculate S_i, a score for each variable not in the model, adjusted for all variables already in the model,
3. Find j such that $S_j \geq S_i$ for all $i, i \neq j$, i.e. we find the variable (not in the model) with the largest value of S. We denote the variable associated with the jth score, z_j,
4. Calculate p, the p-value associated with the S_j,
5. If $p > p_a$ then go to step 8,
6. Add variable z_j to the model.
7. Calculate W_i, a score for each variable in the model, adjusted for all other variables in the model,
8. Find k such that $W_j \leq W_i$ for all $i, i \neq k$, i.e. we find the variable (in the model) with the smallest value of W. We denote the variable associated with the kth score, z_k,
9. Calculate p, the p-value associated with W_i,
10. If $p \geq p_d$ then
    a. Drop variable z_k from the model,
    b. If $z_k = z_j$, i.e. the variable that was added at this iteration was also dropped, then go to step 12.

11. If there are still variables not in the model then go to step 2,
12. Stop.

In order to perform a stepwise selection procedure one must therefore choose two scoring statistics, S and W, and a two cut-offs p_a and p_d. The scoring statistics could be the same, but the the same well known statistical package again uses the Score statistic for S and the Wald test statisic for W and uses the same a default value of 0.05 for for p_a and p_d.

As we already know how to calculate both the Score test statistic and the Wald test statistic we can perform stepwise selection utilising the same code as we developed before.

5 Further Comments

Both Cox proportional hazards routines, G12BAF and g12bac, require the design matrix to be supplied. In cases where all of the covariates are binary or continuous (i.e. each covariate has a single degree of freedom), the design matrix is the same as a matrix holding the data. In cases where some of the covariates are categorical (i.e. they can take one of a set of discrete values) and are not binary (i.e. the set of discrete values has more than two values), then some preprocessing must be performed to obtain a series of *dummy variables*. A description of dummy variables and the various ways they can be produced can be found in Section 3 of the documentation for G04EAF.

The example code supplied with this whitepaper makes the assumption that all of the covariates being considered have a single degree of freedom. If this is not the case, then some recoding will be necessary. The recoding will need to allow dummy variables to be added or dropped from the model as a block, for example, if a covariate can take one of k possible values, then it can be represented by $k-1$ dummy variables. These $k-1$ dummy variables need to be added or dropped from the model together, as it usually does not make sense to include a subset of them. In addition the degrees of freedom used when calculating p-value will need to the additional degrees of freedom into account.

A side effect of the example code assuming that all of the covariates being considered have a single degree of freedom, i.e. $\nu = 1$ in step 4 of the backward selection process is that the simplified version of the Wald test statistic is all that is required (i.e. $W = \hat{\beta}_i^2/\sigma_{ii}$). However, to demonstrate a more complicated case where the simplification is not possible we have included an example of testing the global null hypothesis using the Wald statistic.

Referenced NAG Routines

Fortran Library

F06EAF	Dot product of two real vectors
DDOT	Dot product of two real vectors
F06PEF	Matrix-vector product, real symmetric packed matrix
DSPMV	Matrix-vector product, real symmetric packed matrix
F06YJF	Solves a system of equations with multiple right-hand sides, real triangular coefficient matrix
DTRSM	Solves a system of equations with multiple right-hand sides, real triangular coefficient matrix
F07KDF	Cholesky factorization of real symmetric positive semidefinite matrix
DPSTRF	Cholesky factorization of real symmetric positive semidefinite matrix
G01ECF	Computes probabilities for χ^2 distribution
G12BAF	Fits Cox's proportional hazard model

C Library

f07gdc	Cholesky factorization of real symmetric positive definite matrix, packed storage
nag_dpptrf	Cholesky factorization of real symmetric positive definite matrix, packed storage
f16pec	Matrix-vector product, real symmetric packed matrix
nag_dspmv	Matrix-vector product, real symmetric packed matrix
f16plc	System of equations, real triangular packed matrix
nag_dtpsv	System of equations, real triangular packed matrix
g01ecc	Probabilities for χ^2 distribution
g12bac	Fits Cox's proportional hazard model

References

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