

University of Vienna

Department of Medical Computer Sciences

Section of Clinical Biometrics

Section Head: Prof. M. Schemper

A-1090 VIENNA, Spitalgasse 23

Phone: (+43)(1) 40400/6688

Fax: (+43)(1) 40400/6687

e-mail: [imc@akh-wien.ac.at](mailto:imc@akh-wien.ac.at)

Technical Report 1/2001

**An SPLUS library  
to perform Cox regression  
without convergence problems**

Meinhard PLONER and Georg HEINZE

e-mail: [meinhard.ploner@akh-wien.ac.at](mailto:meinhard.ploner@akh-wien.ac.at)

[georg.heinze@akh-wien.ac.at](mailto:georg.heinze@akh-wien.ac.at)

## Abstract

The phenomenon of monotone likelihood is observed in the fitting process of a Cox model if the likelihood converges while at least one parameter estimate diverges to  $\pm$  infinity. Monotone likelihood primarily occurs in small samples with several unbalanced and highly predictive covariates, and with a high percentage of censoring. A procedure by Firth (1993) originally developed to reduce the bias of maximum likelihood estimates provides an ideal solution to monotone likelihood (cf. Heinze & Schemper, 2001). It produces finite parameter estimates by means of penalized maximum likelihood estimation. Corresponding Wald tests and confidence intervals are available but it was shown that penalized likelihood ratio tests and profile penalized likelihood confidence intervals are often preferable.

This Technical Report presents an SPLUS library to apply Firth's procedure to Cox regression. The present report contains the complete User's Guide to this library including syntax, computational methods and examples.

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>The library ‘coxphf’</b>	<b>5</b>
2.1	Overview . . . . .	5
2.2	Installation of the library . . . . .	6
2.3	Breast cancer example . . . . .	6
<b>3</b>	<b>The main function ‘coxphf’</b>	<b>9</b>
3.1	Syntax . . . . .	9
3.2	Required arguments . . . . .	9
3.3	Optional arguments . . . . .	9
3.4	Returned object . . . . .	10
3.5	Specific methods . . . . .	11
<b>4</b>	<b>The function ‘coxphftest’</b>	<b>11</b>
4.1	Overview . . . . .	11
4.2	Syntax . . . . .	12
4.3	Required arguments . . . . .	12
4.4	Optional arguments . . . . .	12
4.5	Returned object . . . . .	12
4.6	Specific methods . . . . .	13
<b>5</b>	<b>The function ‘coxphfplot’</b>	<b>13</b>
5.1	Overview . . . . .	13
5.2	Syntax . . . . .	13
5.3	Required arguments . . . . .	13
5.4	Optional arguments . . . . .	14
5.5	Returned object . . . . .	15

# 1 Introduction

For almost three decades the semiparametric proportional hazards regression model of Cox (1972) has been first choice for the analysis of survival data. The straightforward interpretation of the estimated parameters as log relative risks favored its popularity in medical research. Parameters are estimated by maximizing the (log) likelihood function (maximum likelihood method). However, it is also known that there are certain situations where finite maximum likelihood parameter estimates do not exist, i. e. where the likelihood converges to a finite value while at least one parameter estimate diverges to  $\pm\infty$  (Bryson & Johnson, 1981). This phenomenon is due to special conditions in a data set and is known as ‘monotone likelihood’. The probability of occurrence of monotone likelihood is too high to be negligible (Heinze & Schemper, 2001; Heinze, 1999).

A new procedure that arrives at finite estimates for the parameters by a modification of the score function has been proposed for Cox regression by Heinze & Schemper (2001). This modification was originally developed by Firth (1993, 1992a, 1992b) to reduce the bias of maximum likelihood estimates in generalized linear models. These estimates are biased away from zero (cf. e. g. Schaefer, 1983; Cordeiro & McCullagh, 1991) and the occurrence of infinite parameter estimates in situations of separation can be interpreted as an extreme consequence of this property. The behaviour of the new procedure has been extensively studied by Heinze & Schemper (2001) and Heinze (1999).

In statistical software packages for Cox regression the convergence of the model fitting algorithm is usually based on the log likelihood (SAS Institute, 1999b; MathSoft, 1997). The resulting relative risk estimates are based on that iteration where the log likelihood changes by less than a very small prespecified value (e. g.  $10^{-6}$ ). In case of monotone likelihood the algorithm converges but nonexistence of finite estimates is often overlooked. The resulting estimates are completely arbitrary and thus extremely inaccurate (Heinze & Schemper, 2001) and misleading.

Heinze & Ploner (2001) present SAS (SAS Institute, 1999a) and SPLUS (MathSoft, 1997) programs that apply the methods of Heinze & Schemper (2001). While the SAS program has been extensively described by Heinze (1999), this Technical Report contains the complete User’s Guide for the SPLUS library. In § 2, we give an overview of the library and its installation on a PC. A breast cancer example shows the simplicity of the library function call. In §§ 3–5 we describe the main function and two additional functions of the library by means of syntax and examples.

## 2 The library ‘coxphf’

### 2.1 Overview

The S-PLUS library `coxphf` provides a comprehensive tool to facilitate the application of Firth’s modified score procedure in Cox regression analysis. It has been written on a PC for use with the operating system Windows and is available at the web-site <http://www.akh-wien.ac.at/imc/biometrie/fc>.

As the core routine `fcdll.dll` is written in FORTRAN and compiled for Windows-32 compatible PC, it can also be called from other programs, provided the input data sets follow the structure outlined in Heinze (1999), § 4.2.10.

The call of the main function of the library follows the structure of the related standard Cox-function `coxph`, requiring a `data.frame` and a `formula` for the model specification. The resulting object belongs to the new class `coxphf`, which inherits from class `coxph` and includes penalized maximum likelihood (‘Firth-Cox’- or ‘FC’-type) Cox regression parameters, standard errors, confidence limits,  $p$ -values, the value of the maximized penalized log likelihood, the linear predictors, the number of iterations needed to arrive at the maximum and much more. Furthermore, specific methods for the resulting object are supplied. The objects and methods are similar to the `coxph`-counterparts, too. Additionally, a function to plot profiles of the penalized likelihood function and a function to perform penalized likelihood ratio tests have been included.

The summary of the contents of the library can be obtained by calling:

```
> library(coxphf, help=T)
Library COXPHF
```

```
Cox regression using Firth’s modified score function
```

```
Meinhard Ploner and Georg Heinze
Section of Clinical Biometrics
University of Vienna
meinhard.ploner@akh-wien.ac.at
```

```
Functions and datasets in the library:
```

```
coxphf           fits a Cox regression with Firth’s adaptation
coxphfplot       plots the profile penalized likelihood function
```

<code>coxphf</code>	penalized likelihood ratio test
<code>print.coxphf</code>	prints an object of the class <code>&lt;coxphf&gt;</code>
<code>print.coxphf</code>	prints an object of the class <code>&lt;coxphf&gt;</code>
<code>summary.coxphf</code>	requests the summary of an object of the class <code>&lt;coxphf&gt;</code>
<code>breast</code>	breast cancer data set

Copyright University of Vienna  
2000, all rights reserved

The functions of the library `coxphf` access the dynamic link library `FCDLL.DLL` which was written in FORTRAN using Digital Fortran Professional Edition 5.0.A and Microsoft Visual Studio 97. It has been compiled on a PC running the Windows NT 4.0 SP5 operating system. Use of this dynamic link library and the computational algorithms used are described in detail in §§ 4.2.10 and 4.2.11 of Heinze (1999).

## 2.2 Installation of the library

Usually the library is distributed as a packed ZIP-archive. After unpacking the files to the directory `...\splus4\library\` the SPLUS-files are extracted to `...\splus4\library\coxphf\`. The two additional files

- `dforrt.dll`
- `fc.dll`

remain in the directory `...\splus4\library\` and must be moved to the SPLUS home directory. They are needed for the call to the FORTRAN routines. After starting an SPLUS session the library can be loaded directly by putting

```
> library(coxphf)
```

## 2.3 Breast cancer example

Use of the library `coxphf` is exemplified using a breast cancer data set (refer to Heinze, 1999, Tables 16 and 17). Survival times of 100 patients were recorded (74 of them censored) and also values of four potential risk factors: tumour stage (**TS**), nodal status (**N**), histological grading (**G**) and Cathepsin D immunoreactivity (**CD**). For analysis these factors were dichotomised to levels of 0 and 1 (unfavorable). The survival and censoring times are stored in variable `months` with the censoring indicator `cens`, 0 indicating a censored survival time.

Suppose the data has been stored in a S-PLUS `data.frame` named `breast`. To obtain an object of the class `coxphf`, one submits the following commands:

```
> library(coxphf)  ## to load the library
> fit <- coxphf(Surv(months, cens) ~ TS + N + G + CD, data=breast)
```

A short overview of the results, consisting of a table containing variable names, parameter estimates, standard errors, confidence limits and  $p$ -values, can be obtained by using the `print.coxph` method, which is done automatically by putting the name of an object belonging to the specified class:

```
> fit
coxphf(formula = Surv(months, cens) ~ TS + N + G + CD, data = breast)
Model fitted by Penalized ML
Confidence intervals and p-values by Profile Likelihood

      coef  se(coef) exp(coef) lower 0.95  upper 0.95   Chisq      p
TS 1.2244363 0.4916037  3.402248  1.3627466    9.472181 6.983775 0.0082251
N  0.9188841 0.4225731  2.506492  1.1204555    5.832861 5.004411 0.0252828
G  2.4244150 1.4735438 11.295620  1.4656683 1451.941939 6.090656 0.0135898
CD 0.3971130 0.4418552  1.487524  0.6268674    3.511783 0.822321 0.3645024
```

Likelihood ratio test=35.951420357964 on 4 df,  $p=2.9610538088143e-007$ ,  $n=100$

The summary method `summary.coxphf` produces more output, including the covariance-matrix:

```
> summary(fit)
coxphf(formula = Surv(months, cens) ~ TS + N + G + CD, data =
breast)
...
...
Likelihood ratio test=35.951420357964 on 4 df,  $p=2.9610538088143e-007$ ,  $n=100$ 
Wald test = 23.200026593425 on 4 df,  $p = 0.000115491291690861$ 
```

Covariance-Matrix:

	TS	N	G	CD
TS	0.2416741656	-0.0007910992	-0.06806775	-0.07374013
N	-0.0007910992	0.1785679981	-0.04416373	-0.05117875

```
G -0.0680677456 -0.0441637348 2.17133144 -0.02606946
CD -0.0737401323 -0.0511787518 -0.02606946 0.19523602
```

The output object consists of many attributes which can be extracted in the usual manner by the `$`-operator or the specific method, if given:

```
> attributes(fit)
```

```
$names:
```

```
[1] "coefficients"      "alpha"             "var"
[4] "df"                "loglik"            "iter"
[7] "method.ties"       "n"                  "terms"
[10] "y"                 "formula"           "call"
[13] "means"             "linear.predictors" "method"
[16] "method.ci"         "ci.lower"          "ci.upper"
[19] "prob"
```

```
$class: [1] "coxphf"
```

```
> fit$linear.predictors
```

```
[1] 0.91509980 0.08777648 2.23109692 1.31221281 1.83398391 2.23109692
[7] 0.91509980 2.23109692 0.91509980 2.23109692 ...
```

```
> fit$y
```

```
[1] 1.546053 3.519737 8.059211 8.651316 12.138158 12.368421
[7] 13.059211 14.769737 18.848684 20.032895 20.723684 21.644737
[13] 23.355263+ 23.684211 26.611842 27.960526+ 28.355263+ ...
```

```
> fit$var
```

```
          TS          N          G          CD
TS 0.2416741656 -0.0007910992 -0.06806775 -0.07374013
N -0.0007910992 0.1785679981 -0.04416373 -0.05117875
G -0.0680677456 -0.0441637348 2.17133144 -0.02606946
CD -0.0737401323 -0.0511787518 -0.02606946 0.19523602
```

```
> coef(fit)
```

```
          TS          N          G          CD
1.224436 0.9188841 2.424415 0.397113
```

## 3 The main function ‘coxphf’

### 3.1 Syntax

The definition of the main function is:

```
coxphf(formula=attr(data, "formula"), data=sys.parent(),
       pl = T, alpha = 0.05, maxit = 25, maxhs = 3,
       epsilon = 1e-006, maxstep = .5, firth=T)
```

### 3.2 Required arguments

- **formula**: a formula object, with the response on the left of the  $\sim$  operator, and the model terms on the right. The response must be a survival object as returned by the function `Surv`. To obtain as covariates all 1<sup>st</sup>- and 2<sup>nd</sup>-order interactions of A, B and C, specify `Surv(time, status) ~ A*B*C - A:B:C`
- **data**: a `data.frame` where the variables named in the `formula` can be found, i. e. the variables containing time, status and the covariates.

### 3.3 Optional arguments

- **pl**: specifies if confidence intervals and tests should be based on the profile penalized log likelihood (`pl=T`, the default) or on the Wald method (`pl=F`).
- **alpha**: the significance level (1– the confidence level, 0.05 as default)
- **maxit**: maximum number of iterations (default value is 25)
- **maxhs**: maximum number of step-halvings per iterations (default value is 3)
- **epsilon**: specifies the maximum allowed change in penalized log likelihood to declare convergence. Default value is  $10^{-6}$ .
- **maxstep**: specifies the maximum change of (standardized) parameter values allowed in one iteration. Default value is 0.5.
- **firth**: use of Firth’s penalized maximum likelihood (`firth=T`, default) or the standard maximum likelihood method (`firth=F`) for the Cox regression. Note that by specifying `pl=T` and `firth=F` (and probably a lower number of iterations) one obtains profile likelihood confidence intervals for maximum likelihood Cox regression parameters. This feature has not yet been implemented in any major software package.

### 3.4 Returned object

The object returned is of the class `coxphf` and has the following attributes:

- `coefficients`: the coefficients of the parameter in the fitted model.
- `alpha`: the significance level (1– the confidence level) as specified in the input.
- `var`: the variance-covariance-matrix of the parameters.
- `df`: the number of degrees of freedom in the model.
- `loglik`: a vector of the (penalized) log-likelihood of the full and the restricted models.
- `iter`: the number of iterations needed in the fitting process.
- `method.ties`: the method of handling ties (always "breslow").
- `n`: the number of observations.
- `terms`: an object of mode `expression` and class `term` summarizing the formula as described in the help of S-PLUS.
- `y`: the response, i. e. the time alone, or the time with a plus if the observation is censored.
- `formula`: the `formula` object, see S-PLUS help.
- `call`: the `call` object, see S-PLUS help.
- `means`: vector of column means of the X matrix. Subsequent survival curves computed by `plot(survfit(object))` refer to this value.
- `linear.predictors`: a vector with the linear predictor of each observation.
- `method`: depending on the fitting method "Penalized ML" or "Standard ML".
- `method.ci`: the method in calculating the confidence intervals, i.e. "profile likelihood" or "Wald", depending on the argument `pl`.
- `ci.lower`: the lower confidence limits of the parameter.
- `ci.upper`: the upper confidence limits of the parameter.
- `prob`: the  $p$ -values of the specific parameters.

### 3.5 Specific methods

The definition of the methods `print` and `summary` is as follows:

```
print.coxphf(object)
summary.coxphf(object)
```

where `object` represents an object of the class `coxphf`, which is the output of the function `coxphf`.

## 4 The function ‘coxphftest’

### 4.1 Overview

This function performs a penalized likelihood ratio test on some (or all) selected factors. The resulting object is of the class `coxphftest` and includes the information printed by the proper `print` method.

If for the breast-cancer example we would like to test the specific hypothesis  $\beta_G = 1, \beta_{CD} = 0$ , we do as follows:

```
> coxphftest(Surv(months, cens) ~., data=breast, test=~G + CD, values=c(1,0))
coxphftest(formula = Surv(months, cens) ~ ., data = breast, test= ~ G + CD,
  values = c(1, 0))
```

Model fitted by Penalized ML

Factores fixed as follows:

```
TS  N G CD
NA NA 1  0
```

Likelihoods:

```
Restricted model Full model difference
-93.60053 -92.35242  1.248115
```

Likelihood ratio test=2.4962309288226 on 2 df, p=0.287045234423171

Testing the overall null hypothesis of  $\beta_i = 0$  would be performed by the following call:

```
> coxphftest(Surv(months, cens) ~., data=breast)
```

## 4.2 Syntax

The definition of the main function is:

```
coxphftest(formula=attr(data, "formula"), data=sys.parent(),
  test=~., values, maxit = 25, maxhs = 3, epsilon = 1e-006,
  maxstep = .5, firth = T)
```

## 4.3 Required arguments

- **formula**: a formula object, with the response on the left of the  $\sim$  operator, and the model terms on the right. The response must be a survival object as returned by the function `Surv`.
- **data**: a `data.frame` in which to interpret the variables named in the `formula`, i.e the variables containing time, status and the covariates.

## 4.4 Optional arguments

`coxphftest` takes the same arguments as the main function `coxphf`, except for `alpha`. Additionally, we can specify the factors/covariates to be tested and the null hypothesis values:

- **test**: righthand formula of parameters to test (e.g.  $\sim B + D$ ). The default value is  $\sim .$ , testing all the model.
- **values**: null hypothesis values, default value is 0. For testing the specific hypothesis  $\beta_1 = 1, \beta_4 = 2, \beta_5 = 0$  we specify `test= ~ B1 + B4 + B5` and `values=c(1, 2, 0)` or `c(1, 2)`.

## 4.5 Returned object

The object returned is of the class `coxphf` and has the following attributes:

- **testcov**: a vector of the fixed values of each covariate; `NA` stands for a parameter which is not tested.
- **loglik**: a vector of the (penalized) log-likelihood of the full and the restricted models.
- **df**: the number of degrees of freedom in the model.

- `prob`: the  $p$ -value of the test.
- `call`: the `call` object, see S-PLUS help.
- `method`: depending on the fitting method "Penalized ML" or "Standard ML".

## 4.6 Specific methods

The definition of the method `print` is:

```
print.coxphf(object)
```

where `object` represents an object of the class `coxphfetest`, which is the output of the function `coxphfetest`.

## 5 The function ‘`coxphfplot`’

### 5.1 Overview

This function plots the profile likelihood of a specific parameter. In our example we get the profile of the parameter  $G$  as follows:

```
> coxphfplot(Surv(months, cens) ~ ., data=breast, which= ~ G)
```

Fig. 1 shows the graph of the profile penalized likelihood function for parameter  $\beta_G$ .

### 5.2 Syntax

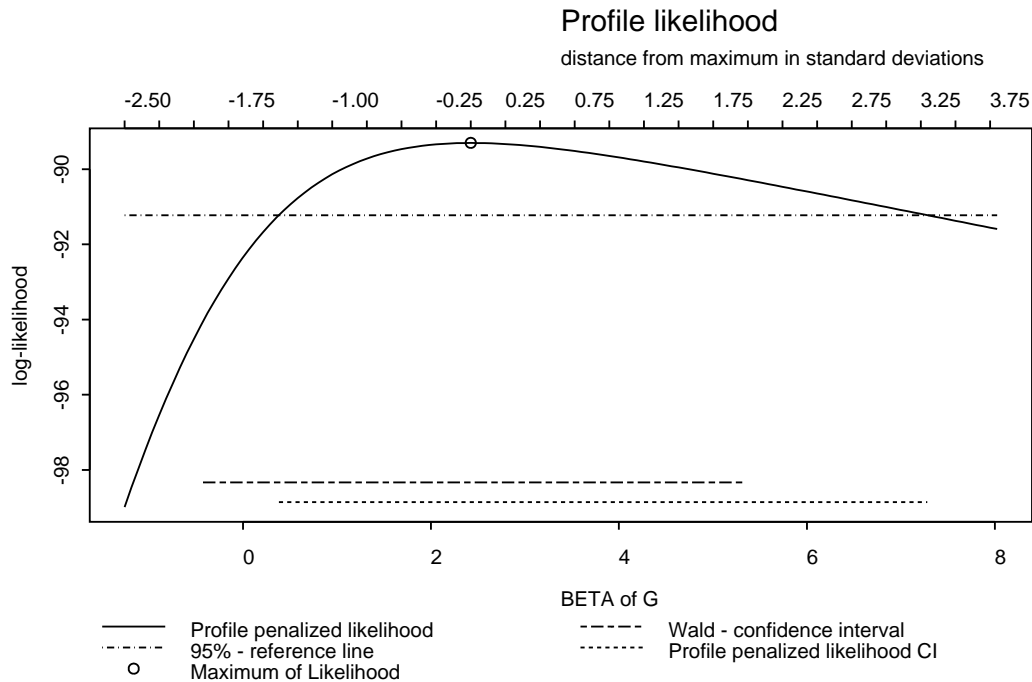
The definition of the main function is

```
coxphfplot(formula=attr(data, "formula"), data=sys.parent(),
  which, pitch=.05, limits, alpha=.05, maxit = 25, maxhs = 3,
  epsilon = 1e-006, maxstep = .5, firth = T, legends=T)
```

### 5.3 Required arguments

- `formula`: a formula object, with the response on the left of the  $\sim$  operator, and the model terms on the right. The response must be a survival object as returned by the function `Surv`.
- `data`: a `data.frame` in which to interpret the variables named in the `formula`, i. e. the variables containing time, status and the covariates.

Figure 1: Profile penalized likelihood function for parameter  $\beta_G$  of breast cancer example.



- **which:** a righthand formula specifying the plotted parameter, interaction or general term, e.g.  $\sim A$  or  $\sim A : C$

## 5.4 Optional arguments

`coxphfplot` takes all arguments of the main function `coxphf`, except for `p1`. In addition, we can specify the following arguments:

- **limits:** vector of the minimum and the maximum on the  $x$ -scale in standard deviations distant from the maximum likelihood. The default values are the extremes of both confidence intervals, Wald and PL, plus or minus half a standard deviation of the parameter, respectively.
- **pitch:** distances between the interpolated points in standard errors of the parameter estimate, the default value is 0.05.
- **legends:** if F, legends on the bottom of the plot would be omitted (default is T).

## 5.5 Returned object

The object returned is a simple `data.frame` containing three columns which allow reproducing the plot. Each row represents one point of the interpolation. The columns are as follows:

- `std`: distance from the maximum of the profile likelihood (in standard errors of the parameter estimate).
- `name`: the value of the parameter for the variable `name` specified in argument `which`.
- `loglik.pen`: the value of the penalized likelihood.

## References

- BRYSON, M. C. & JOHNSON, M. E. (1981). The incidence of monotone likelihood in the Cox model. *Technometrics* **23**, 381–383.
- CORDEIRO, G. M. & MCCULLAGH, P. (1991). Bias correction in generalized linear models. *Journal of the Royal Statistical Society B* **53**, 629–643.
- COX, D. R. (1972). Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society B* **34**, 187–220.
- FIRTH, D. (1992a). Bias reduction, the Jeffreys prior and GLIM. In Fahrmeir, L., Francis, B., Gilchrist, R., & Tutz, G., editors, *Advances in GLIM and Statistical Modelling*, pages 91–100. Springer-Verlag, New York.
- FIRTH, D. (1992b). Generalized linear models and Jeffreys priors: an iterative weighted least-squares approach. In Dodge, Y. & Whittaker, J., editors, *Computational Statistics*, volume 1, pages 553–557, Heidelberg. Physica-Verlag.
- FIRTH, D. (1993). Bias reduction of maximum likelihood estimates. *Biometrika* **80**, 27–38.
- HEINZE, G. (1999). *Technical Report 10: The application of Firth’s procedure to Cox and logistic regression*. Department of Medical Computer Sciences, Section of Clinical Biometrics, Vienna University, Vienna.
- HEINZE, G. & PLONER, M. (2001). SAS and SPLUS programs to perform Cox regression without convergence problems. *To appear in ‘Computer methods and programs in Biomedicine’*.
- HEINZE, G. & SCHEMPER, M. (2001). A solution to the problem of monotone likelihood in Cox regression. *Biometrics* **57**.
- MATHSOFT (1997). *SPLUS 4.0*. MathSoft Inc., Cambridge, MA.
- SAS INSTITUTE (1999a). *SAS Language Reference, Version 8*. SAS Institute Inc., Cary, NC.
- SAS INSTITUTE (1999b). *SAS/STAT User’s Guide, Version 8*. SAS Institute Inc., Cary, NC.
- SCHAEFER, R. L. (1983). Bias correction in maximum likelihood logistic regression. *Statistics in Medicine* **2**, 71–78.