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

Technical Report 2/2001

An S-PLUS library to perform logistic regression without convergence problems

Meinhard PLONER

e-mail: meinhard.ploner@akh-wien.ac.at

Abstract

The phenomenon of separation is observed in the fitting process of a logistic model if the likelihood converges while at least one parameter estimate diverges to ± infinity. Separation primarily occurs in small samples with several unbalanced and highly predictive covariates (c.f. Heinze (1999)). 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 S-PLUS library to apply Firth's procedure to logistic regression. The present report contains the complete User's Guide to this library including syntax, computational methods and examples.

Contents

1 Introduction

In logistic regression usually the 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 $\pm\infty$ (Albert & Anderson, 1984). This phenomenon is due to special conditions in a data set and is known as 'separation'. 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 logistic 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 logistic regression the convergence of the model fitting algorithm is usually based on the log likelihood (SAS, 1999; MathSoft, 1997). The resulting odds ratio 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.

While a SAS program that apply the new methods has been extensively described by Heinze (1999), this Technical Report contains the complete User's Guide for the S-PLUS library. In § 2 we describe the estimation procedure used in fitting the models. In § 3, we give an overview of the library and its installation on a PC. An *Epidemiological Study* example shows the simplicity of the library function call. In §§ 4–6 we describe the main function and two additional functions of the library by means of syntax and examples.

2 Estimation

In explaining the details of the estimation process we follow mainly the description in Heinze & Ploner (2001). In general, maximum likelihood estimates are often prone to small sample bias. To reduce this bias, Firth (1993) suggested to maximize the penalized log likelihood $\log L(\beta)^* = \log L(\beta) + 0.5 \log |I(\beta)|$, where $I(\beta)$ is the Fisher information matrix, i. e. minus the second derivative of the log likelihood. Applying this idea to logistic regression, the score function $U(\beta)$ is replaced by the modified score function $U(\beta)^* = U(\beta) + a$, where a has rth entry $a_r = 0.5\text{tr}\left\{I(\beta)^{-1}\left[\partial I(\beta)/\partial\beta_r\right]\right\}$, $r = 1, \ldots, k$. Heinze and Schemper (2001) give the explicit formulae for $I(\beta)$ and $\partial I(\beta)/\partial\beta_r$.

In our programs estimation of β is based on a Newton-Raphson algorithm. Parameter values are initialized usually with 0, but in general the user can specify arbitrary starting values.

With a starting value of $\beta^{(0)}$, the penalized maximum likelihood estimate $\hat{\beta}$ is obtained iteratively:

$$\beta^{(s+1)} = \beta^{(s)} + I(\beta^{(s)})^{-1}U(\beta^{(s)})^* \tag{2·1}$$

If the penalized log likelihood evaluated at $\beta^{(s+1)}$ is less than that evaluated at $\beta^{(s)}$, then $\beta^{(s+1)}$ is recomputed by step-halving. For each entry r of β with $r = 1, \ldots, k$ the absolute step size $|\beta_r^{(s+1)} - \beta_r^s|$ is restricted to a maximal allowed value ξ. These two means should avoid numerical problems during estimation. The iterative process is continued until the parameter estimates converge.

Computation of profile penalized likelihood confidence intervals for parameters follows the algorithm of Venzon and Moolgavkar (1988). For testing the hypothesis of $\gamma = \gamma_0$, let the likelihood ratio statistic $LR = 2\left[\log L(\hat{\gamma}, \hat{\delta})^* - \log L(\gamma_0, \hat{\delta}_{\gamma_0})^*\right]$, where $(\hat{\gamma}, \hat{\delta})$ is the joint penalized maximum likelihood estimate of $\beta = (\gamma, \delta)$, and $\hat{\delta}_{\gamma_0}$ is the penalized maximum likelihood estimate of δ when $\gamma = \gamma_0$. The profile penalized likelihood confidence interval is the continuous set of values γ_0 for which LR does not exceed the $(1-\alpha)100$th percentile of the χ^2_1-distribution. The confidence limits can therefore be found iteratively by approximating the penalized log likelihood function in a neighborhood of β by the quadratic function

$$\tilde{\ell}(\beta + \delta) = \ell(\beta) + \delta' U^* - \frac{1}{2}\delta' I \delta$$

where $U^* = U(\beta)^*$ and $-I = -I(\beta)$. Suppose the confidence limits for parameter β_r are to be computed. Then the increment vector δ for the next iteration is obtained by solving the likelihood equations

$$\frac{\partial}{\partial\delta}\{\tilde{\ell}(\beta + \delta) + \lambda(e_r'\delta - \theta)\} = 0$$

where λ is a Lagrange multiplier, e_r is the rth unit vector, and θ is an unknown constant. The solution is

$$\delta = I^{-1}(U^* + \lambda e_r) \tag{2·2}$$

Let $\ell_0 = \ell_{\max} - \chi^2_{1,1-\alpha}/2$, where $\ell_{\max}$ is the maximized value of the penalized log likelihood, and $\chi^2_{1,\alpha}$ is the α-quantile of the χ^2-distribution with one degree of freedom. By substituting (2·2) into the equation $\tilde{\ell}(\beta + \delta) = \ell_0$, λ can be estimated as

$$\lambda = \pm \left[\frac{2\left\{\ell_0 - \ell(\beta) - \frac{1}{2}U^{*\prime} I^{-1} U^*\right\}}{e_r' I^{-1} e_r} \right]^{\frac{1}{2}}$$

The upper confidence limit for β_r is computed by starting at the penalized maximum likelihood estimate of β and iterating with positive values of λ until convergence is attained. The process is repeated for the lower confidence limit using negative values of λ. Convergence is declared on the current iteration if $|\ell(\beta) - \ell_0| \leq \epsilon$ and $(U^* + \lambda e_r)' I^{-1} (U^* + \lambda e_r) \leq \epsilon$.

In some situations computation of profile penalized likelihood confidence intervals may be time consuming since the iterative procedure outlined above has to be repeated for the lower and for the upper confidence limits of each of the k parameters. In other problems one may not be interested in interval estimation, anyway. In such cases, the user can request computation of Wald confidence intervals and P-values, which are based on the normal approximation of the parameter estimates and do not need any iterative estimation process. Standard errors $\hat{\sigma}_r$, $r = 1, \ldots, k$, of the parameter estimates are computed as the roots of the diagonal elements of the variance matrix $V(\hat{\beta}) = I(\hat{\beta})^{-1}$. A $(1-\alpha) \times 100\%$ Wald confidence interval for parameter β_r is then defined as $[\hat{\beta}_r + \Phi_{\alpha/2}\hat{\sigma}_r, \hat{\beta}_r + \Phi_{1-\alpha/2}\hat{\sigma}_r]$ where Φ_α denotes the α-quantile of the standard normal distribution function. The adequacy of Wald confidence intervals for parameter estimates should be verified by plotting the profile penalized log likelihood (PPL) function. A symmetric shape of the PPL function allows use of Wald intervals, while an asymmetric shape demands profile penalized likelihood intervals (Heinze & Schemper (2001)).

3 The library 'logistf'

3.1 Overview

The S-PLUS library `logistf` provides a comprehensive tool to facilitate the application of Firth's modified score procedure in logistic regression analysis. It has been written on a PC with S-PLUS 4.0 but it should run on the newer versions as well as with other operation systems like UNIX. The library is available at the web-site `http://www.akh-wien.ac.at/imc/biometrie/fl`.

The call of the main function of the library follows the structure of the standard functions as `lm` or `glm`, requiring a `data.frame` and a `formula` for the model specification. The resulting object belongs to the new class `logistf`, which includes penalized maximum likelihood ('Firth-Logistic'- or 'FL'-type) logistic 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. 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(logistf, help=T)
Library LOGISTF

   Logistic 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 data sets in the library:

      logistf           fits a logistic regression with Firth's adaptation
      logistfplot       plots the profile penalized likelihood function
      logistftest       penalized likelihood ratio test
      print.logistf     prints an object of the class <logistf>
      print.logistftest prints an object of the class <logistftest>
```

```
summary.logistf   requests the summary of an object
                  of the class <logistf>
sex               the College Woman data set in collapsed form
sex2              the College Woman data set in casewise form
```


3.2 Installation of the library

Usually the library is distributed as a packed ZIP-archive. After unpacking the files to the directory `...\splus\library\` the S-PLUS-files are extracted to `...\splus\library\logistf\`. After starting an S-PLUS session the library can be loaded directly by submitting

```
> library(logistf)
```

3.3 Example: Epidemiological study

Use of the library `logistf` is exemplified using the *Epidemiological Study* data set of Foxman (1997), which is available as an ASCII file on the world wide web address `http://www.cytel.com/examples/sex.dat`.
The data set consists of 130 college women with urinary tract infections and 109 uninfected controls. The data set include the binary covariates age (`age`), oral contraceptive use (`oc`), condom use (`vic`), lubricated condom use (`vicl`), spermicide use (`vis`) and diaphragm used (`dia`). The main question arising from this data set is how urinary tract infection is related to age and contraceptive use.

Suppose the data has been stored in a S-PLUS `data.frame` named `sex`. Because the observations are stored in the collapsed form, i. e. for every covariate-combination the number of successes and total size is given, we have to modify to obtain one row per observation with an indicator for the events. Instead of 24 rows we obtain a new `data.frame` with 239 rows, the number of the observations:

```
> sex2 <- rbind(cbind(case=1, sex[rep(1:24, sex$uti), 3:8]),
                cbind(case=0, sex[rep(1:24, sex$GrpSize-sex$uti), 3:8]))
```

To obtain an object of the class `logistf`, one submits the following commands:

```
> library(logistf)   ## to load the library
> fit <- logistf(case ~ ., data=sex2)
```

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
 logistf(formula = case ~ ., data = sex2) Model fitted by Penalized ML
 Confidence intervals and p-values by Profile Likelihood

                  coef  se(coef) lower 0.95 upper 0.95       Chisq            p
 (Intercept)  0.1202541 0.4855415 -0.8185777  1.0731445 0.06286298 8.020268e-001
        age -1.1059815 0.4236601 -1.9737949 -0.3074251  7.50773092 6.143472e-003
         oc -0.0688167 0.4437934 -0.9414289  0.7891995  0.02467044 8.751911e-001
        vic  2.2688747 0.5484159  1.2730212  3.4354329 22.93139022 1.678877e-006
       vicl -2.1114083 0.5430823 -3.2608638 -1.1177349 19.10407252 1.237805e-005
        vis -0.7883170 0.4173676 -1.6080866  0.0151937  3.69740975 5.449701e-002
        dia  3.0960078 1.6750220  0.7745682  8.0302936  7.89693139 4.951873e-003

 Likelihood ratio test=50.885509402167 on 7 df, p=9.6765364610008e-009, n=239
```

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

```
> summary(fit)
 logistf(formula = case ~ ., data = sex2)
 ...
 ...
Covariance-Matrix:
            (Intercept)           age           oc               dia
 (Intercept) 0.23575053 -0.0274268412 -0.195208786   ...   -0.026979464
        age -0.02742684  0.1794879029  0.001817975   ...   -0.065571649
         oc -0.19520879  0.0018179746  0.196952606   ...    0.025480432
        vic -0.12109972 -0.0231659050  0.087887752   ...    0.025211511
       vicl -0.02893131  0.0345957183  0.025843014   ...   -0.008037505
        vis -0.04648040  0.0007177051  0.042053095   ...   -0.045265574
        dia -0.02697946 -0.0655716489  0.025480432   ...    2.805698651
```

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"          "df"
  [5] "loglik"       "iter"         "n"            "terms"
  [9] "y"            "formula"      "call"         "linear.predictors"
 [13] "ci.lower"     "predict"      "hat.diag"     "method"
 [17] "ci.upper"     "prob"         "method.ci"

$class:
  [1] "logistf"

> fit$y
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 ...
[109] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 ...

> fit$linear.predictors
  [1] -0.66806291  2.38912879  2.38912879  2.38912879  2.38912879 ...
 [13]  2.38912879  2.38912879  2.38912879  1.60081181  1.60081181 ...

> fit$predict
  [1]  0.3389307  0.9159946  0.9159946  0.9159946  0.9159946 ...
 [13]  0.9159946  0.9159946  0.9159946  0.8321318  0.8321318 ...

> fit$hat.diag
  [1]  0.07102270  0.02264682  0.02264682  0.02264682  0.02264682 ...
 [13]  0.02264682  0.02264682  0.02264682  0.04817810  0.04817810 ...
```

```
> fit$ci.lower
 (Intercept)        age         oc       vic      vicl       vis       dia
  -0.8185777 -1.973795 -0.9414289 1.273021 -3.260864 -1.608087 0.7745682

> fit$loglik
 [1] -157.9821 -132.5394

> coef(fit)
 (Intercept)        age         oc       vic      vicl       vis      dia
   0.1202541 -1.105982 -0.0688167 2.268875 -2.111408 -0.788317 3.096008
```

4 The main function 'logistf'

4.1 Syntax

The definition of the main function is:

```
logistf <- function(formula=attr(data, "formula"), data=sys.parent(),
   pl = T, alpha = 0.05, maxit = 25, maxhs=5, epsilon = .0001,
   maxstep = 10, firth=T, beta0)
```

4.2 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 vector with 0 and 1 or F and T for the model outcome, where the higher value (1 or T) is modeled. It's possible to include contrasts, interactions, nested effects, cubic or polynomial splines and all the S-PLUS features, as well,
  e.g. `Y ~ X1*X2 + ns(X3, df=4)`.
- `data`: a `data.frame` where the variables named in the `formula` can be found, i. e. the variables containing the binary response and the covariates.

4.3 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 5)
- `epsilon`: specifies the maximum allowed change in penalized log likelihood to declare convergence. Default value is 0.0001.
- `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 logistic 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 logistic regression parameters.

- `beta0`: specifies the initial values of the coefficients for the fitting algorithm.

4.4 Returned object

The object returned is of the class `logistf` 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.
- `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-vector, i. e. 1 for successes (events) and 0 for failures.
- `formula`: the `formula` object, see S-PLUS help.
- `call`: the `call` object, see S-PLUS help.
- `linear.predictors`: a vector with the linear predictor of each observation.
- `predict`: a vector with the predicted probability of each observation.
- `hat.diag`: a vector with the diagonal elements of the Hat Matrix.
- `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.

4.5 Specific methods

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

```
print.logistf(object)
summary.logistf(object)
```

where `object` represents an object of the class `logistf`, which is returned by the function `logistf`. The specific `print`-method prints the coefficients, the standard errors, confidence limits and p-values of the model while the `summary`-method additionally shows the covariance-matrix of the coefficients.

5 The function 'logistftest'

5.1 Overview

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

If for the data set we would like to test the specific hypothesis $\beta_{vic} = 2$, $\beta_{vicl} = 0$, we do as follows:

```
> logistftest(case ~ ., sex2, test = ~ vic + vicl - 1, values = c(2, 0))
 logistftest(formula = case ~ ., data = sex2,
     test =  ~ vic + vicl - 1, values = c(2, 0))
 Model fitted by Penalized ML

 Factores fixed as follows:
 (Intercept) age oc vic vicl vis dia
          NA  NA NA   2    0  NA  NA

 Likelihoods:
 Restricted model Full model difference
         -144.497  -132.5394   11.95762

 Likelihood ratio test=23.9152322431463 on 2 df, p=6.4102254145881e-00
```

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

```
> logistftest(case ~ ., data=sex2)
 logistftest(formula = case ~ ., data = sex2) Model fitted by Penalized ML

 Factores fixed as follows:
 (Intercept) age oc vic vicl vis dia
          NA   0  0   0    0   0   0

 Likelihoods:
 Restricted model Full model difference
        -157.9821  -132.5394   25.44275

 Likelihood ratio test=50.885509402167 on 7 df, p=9.6765364610008e-009
```

5.2 Syntax

The definition of the function is

```
 logistftest <- function(formula=attr(data, "formula"), data=sys.parent(),
    test,   values, maxit = 25, maxhs=5, epsilon = .0001,
    maxstep = 10, firth=T, beta0)
```

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 vector with 0 and 1 or F and T for the model outcome, where the higher value (1 or T) is modeled.

- `data`: a `data.frame` where the variables named in the `formula` can be found, i. e. the variables containing the binary response and the covariates.

5.4 Optional arguments

`logistftest` takes the same arguments as the main function `logistf`, 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. `~ B + D - 1`). As default all parameter apart from the intercept are tested. If `-1` is not included in the formula, the intercept would be tested, too!

As alternative to the formula one can give the indexes of the ordered effects to test (a vector of integers). To test only the intercept specify `test = ~ - .` or `test = 1`.

- `values`: null hypothesis values, default values are 0. For testing the specific hypothesis $\beta_1 = 1, \beta_4 = 2, \beta_5 = 0$ we specify `test= ~ B1 + B4 + B5 - 1` and `values=c(1, 2, 0)`.

- `beta0`: usually omitted. If specified, the full model is not fitted, while the restricted model is fitted by starting with this parameter estimates. The option is utilized by the function `coxphfplot`.

5.5 Returned object

The object returned is of the class `logistf` 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. If the argument `beta0` not missing, the full model isn't evaluated.

- `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".

- `beta`: the coefficients on the restricted solution.

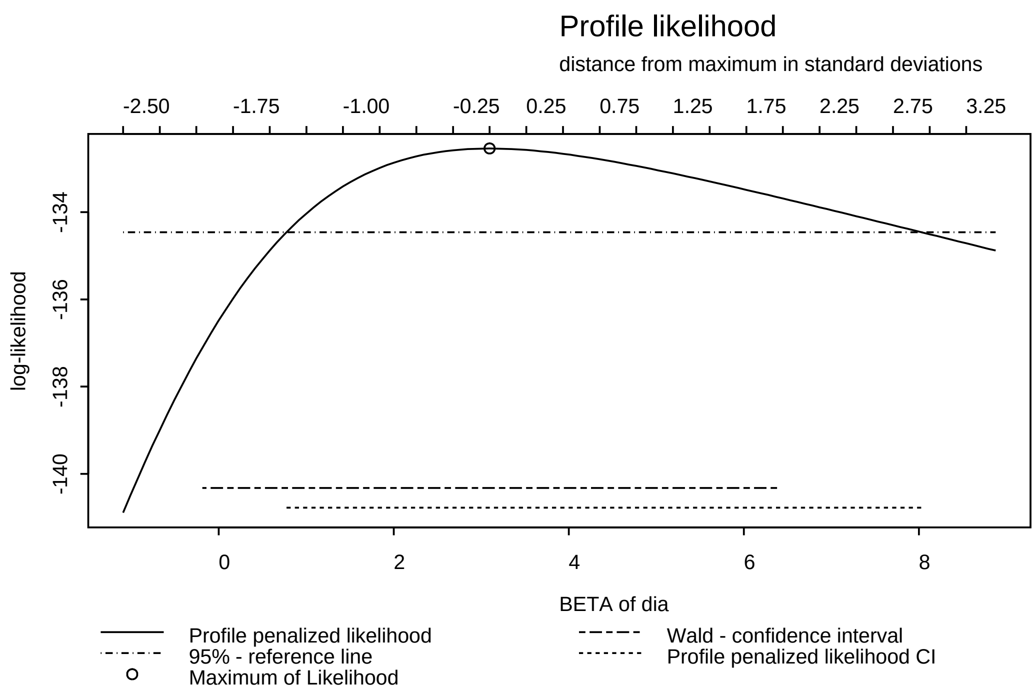
5.6 Specific methods

The definition of the method `print` is:

```
print.logistf(object)
```

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

Figure 1: Profile penalized likelihood function for parameter β_{dia} of *Epidemiological Study* example.



6 The function 'logistfplot'

6.1 Overview

This function plots the profile likelihood of a specific parameter. In our example we get the profile of the parameter β_{dia} as follows:

```
> logistfplot(case ~ ., data=sex2, which= ~ dia - 1)
```

Fig. 1 shows the graph of the profile penalized likelihood function for parameter β_{dia}.

6.2 Syntax

The definition of the function is

```
logistfplot <- function(formula = attr(data, "formula"),
    data = sys.parent(), which, pitch = 0.05, limits, alpha = 0.05,
    maxit = 25, maxhs = 5, epsilon = 0.0001, maxstep = 10, firth = T,
    legends = T)
```

6.3 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 vector with 0 and 1 or F and T for the model outcome, where the higher value (1 or T) is modeled.

- `data`: a `data.frame` where the variables named in the `formula` can be found, i. e. the variables containing the binary response and the covariates.

- `which`: a righthand formula specifying the plotted parameter, interaction or general term, e.g. `~ A - 1` or `~ A : C - 1`. The profile likelihood of the intercept would be obtained by the formula `~ - .`.

6.4 Optional arguments

`logistfplot` takes all arguments of the main function `logistf`, except for `pl`. In addition, we can specify the following arguments:

- `limits`: vector of the minimum and the maximum on the x-scale in standard deviations distant form 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`).

6.5 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

Albert, A. & Anderson, J. A. (1984). On the existence of maximum likelihood estimates in logistic regression models. *Biometrika* **71**, 1-10.

Cordeiro, G. M. & McCullagh, P. (1991). Bias correction in generalized linear models. *Journal of the Royal Statistical Society B* **53**, 629–643.

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.

Foxman, B., Marsh, J., Gillespie, B., Rubin, N., Koopman, J.S. and Spear, S. (1997). Condom Use and First-Time Urinary Tract Infection. *Epidemiology* **8**(6), 637–641.

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. *Computer methods and programs in Biomedicine* , to appear

Heinze, G. & Schemper, M. (2001). A solution to the problem of monotone likelihood in logistic regression. submitted.

MathSoft (1997). *S-PLUS 4.0.* MathSoft Inc., Cambridge, MA.

Ploner, M. & Heinze, G. (2001). *Technical Report 1: An S-PLUS library to perform Cox regression without convergence problems.* Department of Medical Computer Sciences, Section of Clinical Biometrics, Vienna University, Vienna.

SAS (1999). SAS/STAT User's Guide, Version 8, SAS Institute Inc., Cary, NC, 1999.

SAS (1999b). SAS Language Reference, Version 8, SAS Institute Inc., Cary, NC, 1999.

Schaefer, R. L. (1983). Bias correction in maximum likelihood logistic regression. *Statistics in Medicine* **2**, 71–78.

Venzon, D. J. & Moolgavkar, S. H. (1988). A Method for Computing Profile-Likelihood Based Confidence Intervals. *Applied Statistics* **37**, 87–94.