Poisson Regression: nagdmc_poisson_reg

Purpose

nagdmc_poisson_reg computes a regression model with p parameters, either binomial or poisson errors and a variety of link functions.

Declaration

```
#include <nagdmc.h>

void nagdmc_poisson_reg(long rec1, long nvar, long nrec, long dblk, double data[],
                        void (*dfun)(long, long, double [], char *, int *), char *comm,
                        long chunksize, long nxvar, long xvar[], long yvar, long iwts,
                        long ioff, char link, double a, double *dev, long *df,
                        double b[], double se[], double cov[], double model[],
                        double scale, double tol, double eps, long maxit, int *info);
```

Parameters

1: **rec1** – long *Input*

On entry: the index in the data of the first data record used in the analysis.

Constraint: **rec1** ≥ 0.

2: **nvar** – long *Input*

On entry: the number of variables in the data.

Constraint: **nvar** > 1.

3: **nrec** – long *Input*

On entry: the number of consecutive records, beginning at **rec1**, used in the analysis.

Constraint: **nrec** > 1.

4: **dblk** – long *Input*

On entry: the total number of records in the data block.

Constraint: **dblk** $\geq$ **rec1** + **nrec**.

5: **data**[**dblk** $*$ **nvar**] – double *Input*

On entry: the data values for the jth variable (for $j = 0, 1, \ldots,$ **nvar**-1) are stored in **data**[$i*$**nvar**$+j$], for $i = 0, 1, \ldots,$ **dblk** $- 1$. When the data function is used, **data** is not referenced.

6: **dfun** – function supplied by user *External Procedure*

On entry: the pointer to a data function supplied by the user.

Constraint: if **dfun** is a valid pointer, **data** must be 0.

The specification of **dfun** is:

```
void dfun(long irec, long chunksize, double x[], char *comm, int *ierr)
```

1: **irec** – long *Input*

On entry: the index in the data of the first record returned.

2: **chunksize** – long *Input*

On entry: the number of consecutive records returned.

3: **x**[**chunksize**$*$**nvar**] – double *Output*

On exit: data values for the jth variable (for $j = 0, 1, \ldots,$ **nvar** $- 1$) must be returned in **x**[$i *$ **nvar** $+ j$], for $i = 0, 1, \ldots,$ **chunksize** $- 1$.

> 4: **comm** – `char *` *Input*
>
> *On entry:* a communication parameter allowing additional information to be passed to **dfun**. This parameter is passed 'as is' through the calling function.
>
> 5: **ierr** – `int *` *Output*
>
> *On exit:* if the value pointed to by **ierr** on return is greater than 100, the NAG DMC function will terminate immediately and **info** will point to this value.

7: **comm** – `char *` *Input*

On entry: a communication parameter allowing additional information to be passed to **dfun**. This parameter is passed 'as is' through the calling function.

8: **chunksize** – `long` *Input*

On entry: if the data function is used, the function inputs no more than **chunksize** data records at a time; otherwise **chunksize** is not referenced.

Constraint: if **dfun** $\neq 0$, **chunksize** ≥ 1.

9: **nxvar** – `long` *Input*

On entry: the number of independent variables. If **nxvar** $= 0$ then all variables in the data, excluding **yvar** and, if ≥ 0, **iwts** and **ioff**, are treated as independent variables.

Constraint: $0 \leq$ **nxvar** $<$ **nvar**.

10: **xvar**[**nxvar**] – `long` *Input*

On entry: the indices indicating the position in **data** in which values of the independent variables are stored. If **nxvar** $= 0$ then **xvar** must be 0, and the indices of independent variables are given by $j = 0, 1, \ldots,$ **nvar** $- 1$; $j \neq$ **yvar** and **iwts** or **ioff**.

Constraints: if **nxvar** > 0, $0 \leq$ **xvar**$[i] <$ **nvar**, for $i = 0, 1, \ldots,$ **nxvar** $- 1$; otherwise **xvar** must be 0.

11: **yvar** – `long` *Input*

On entry: the index in **data** in which values of the dependent variable are stored.

Constraints: $0 \leq$ **yvar** $<$ **nvar**; if **nxvar** > 0, **yvar** $\neq$ **xvar**$[i]$, for $i = 0, 1, \ldots,$ **nxvar** $- 1$.

12: **iwts** – `long` *Input*

On entry: if **iwts** $= -1$, no weights are used; otherwise **iwts** is the index in **data** in which the weights are stored.

Constraints: $-1 \leq$ **iwts** $<$ **nvar**; **iwts** $\neq$ **yvar**; and if **nxvar** > 0, **iwts** $\neq$ **xvar**$[i]$, for $i = 0, 1, \ldots,$ **nxvar** $- 1$.

13: **ioff** – `long` *Input*

On entry: the index in **data** in which the offset values are stored. If **ioff** $= -1$, no offsets are used.

Constraint: **ioff** $<$ **nvar**.

14: **link** – `char` *Input*

On entry: indicates which link function to use. Values of **link** can be upper or lower case.

- 'I' : Identity link function.
- 'L' : Log link function.
- 'S' : Square root link function.
- 'R' : Reciprocal link function.
- 'E' : Power link function.

Constraint: **link** = 'I', 'i', 'L', 'l', 'S', 's', 'R', 'r', 'E' or 'e'.

15: **a** – `double` *Input*

On entry: if **link** = 'E' then **a** is the power used, otherwise **a** is not referenced.

Constraint: **a** $\neq 0$

16: **dev** – `double` *Output*

On exit: the deviance from the fitted model.

17: **df** – `long *` *Output*

On exit: the degrees of freedom for the deviance.

18: **b**[p] – `double` *Output*

On exit: the parameter estimates. **b**[0] is the mean parameter. **b**[i] is the coefficient of the ith variable included in the model, for $i = 1, 2, \ldots, p-1$. If **nxvar** > 0 then the order the independent variables are added to the model is defined by **xvar**, otherwise the order is defined by indices in the data.

19: **se**[p] – `double` *Output*

On exit: the standard errors of the parameters in **b**.

20: **cov**[$p*(p+1)/2$] – `double` *Output*

On exit: the first $p*(p+1)/2$ elements of **cov** contain the upper triangular part of the variance-covariance matrix of the p parameters in **b**. They are stored packed by column, i.e., the covariance between the parameter estimate given in **b**[i] and the parameter estimate given in **b**[j], $j \geq i$, is stored in **cov**[$j(j+1)/2+i$], for $i = 0, 1, \ldots, p-1$ and $j = i, i+1, \ldots, p-1$.

21: **model**[$(3*p*(p+1))/2+$**nvar**$+14$] – `double` *Output*

On exit: if not 0, information on the fitted model for use in the functions described in 'See Also'.

22: **scale** – `double` *Input*

On entry: the scale parameter used to scale the standard errors of the parameter estimates. If **scale** $= 0.0$, a default value of 1.0 is used.

Constraint: **scale** ≥ 0.0.

23: **tol** – `double` *Input*

On entry: the convergence tolerance for the training. If **tol** is equal to 0.0, a default value of 0.00001 is used.

Constraint: **tol** ≥ 0.0.

24: **eps** – `double` *Input*

On entry: the value of the criterion used for model pruning. If **eps** $= 0.0$, a default value of $1e^{-10}$ is used.

Constraint: **eps** ≥ 0.0.

25: **maxit** – `long` *Input*

On entry: the maximum number of iterations (passes through the data) to be used in training. If **maxit** $= 0$, a default value of 10 is used.

Constraint: **maxit** ≥ 0.

26: **info** – `int *` *Output*

On exit: **info** gives information on the success of the function call:

-4: a model value has reached a boundary.

0: the function successfully completed its task.

i; $i = 1, 2, \ldots, 6, 8, 9, \ldots, 15, 22, 23, 24, 25$: the specification of the ith formal parameter was incorrect.

41: invalid value for a weight.

42: invalid value for response variable.

45: model has not converged.

57: there are no degrees of freedom for the error estimates.

58: the fit is exact, no error estimates.

59: more variables than observations.

98: there is an underlying computational problem (this is an unlikely error exit).

99: the function failed to allocate enough memory.

> 100: an error occurred in a function specified by the user.

Notation

nrec the number of observations, n.
nxvar the number of independent variables, $p-1$.
xvar the independent variables, X, excluding the mean.
yvar the dependent variable, y.
bdvar if **bdvar** ≥ 0, **bdvar** is the index in the data that defines the binomial denominator, t.
iwts if **iwts** ≥ 0, **iwts** is the index in the data that defines the weights, W.
ioff if **ioff** ≥ 0, **ioff** is the index in the data that defines the offset, o.
link character flag indicating which link function $g(.)$ to use.
b the parameter estimates, $\hat{\beta}$.

Description

nagdmc_poisson_reg fits a generalized linear model with poisson errors. The model consists of the following elements.

(a) A set of n observations, y_i, from a poisson distribution

$$\frac{\mu^y e^{-y}}{y!}$$

(b) X, an n by p matrix of independent variables, In most linear regression models the first term is taken as a mean term or an intercept, i.e., $X_{i,1} = 1$, for $i = 1, 2, \ldots, n$; this is assumed in NAG DMC.

(c) A linear model:

$$\eta = \sum \beta_j x_j.$$

(d) A link function $\eta = g(\mu)$, linking the linear predictor, η, and the mean of the distribution, $\mu = \pi t$. The possible link functions are

(i) exponent link: $\eta = \mu^a$, for constant a,
(ii) identity link: $\eta = \mu$,
(iii) log link: $\eta = \log \mu$,
(iv) square root link: $\eta = \sqrt{\mu}$,
(iii) reciprocal link: $\eta = \frac{1}{\mu}$.

(e) A measure of fit, the deviance:

$$\sum_{i=1}^{n} \text{dev}(y_i, \hat{\mu}_i) = \sum_{i=1}^{n} 2 \left[y_i \log \left(\frac{y_i}{\hat{\mu}_i} \right) - (y_i - \hat{\mu}_i) \right].$$

The linear parameters are estimated by iterative weighted least squares. An adjusted dependent variable, z, is formed,

$$z = \eta + (y - \mu) \frac{d\eta}{d\mu},$$

and a working weight, w,

$$w = \left(\tau \frac{d\eta}{d\mu} \right)^2 \text{ where } \tau = \sqrt{\mu}.$$

At each iteration an approximation to the estimate of β, $\hat{\beta}$, is found by the weighted least squares regression of z on X with weights w.

NAG DMC uses a QR decomposition of $w^{\frac{1}{2}}X$, i.e.,

$$w^{\frac{1}{2}}X = QR,$$

where R is a p by p triangular matrix and Q is an n by p column orthogonal matrix. If R is of full rank then $\hat{\beta}$ is the solution to

$$R\hat{\beta} = Q^T w^{\frac{1}{2}} z.$$

If R is not of full rank a solution is obtained by means of a singular value decomposition (SVD) of R.

$$R = Q_* \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} P^T,$$

where D is a k by k diagonal matrix with non-zero diagonal elements, k being the rank of R and $w^{\frac{1}{2}}X$. This gives the solution

$$\hat{\beta} = P_1 D^{-1} \begin{pmatrix} Q_* & 0 \\ 0 & I \end{pmatrix} Q^T w^{\frac{1}{2}} z,$$

P_1 being the first k columns of P, i.e., $P = (P_1 P_0)$.

The iterations are continued until there is only a small change in the deviance.

The initial values for the algorithm are obtained by taking

$$\hat{\eta} = g(y).$$

The fit of the model can be assessed by examining and testing the deviance, in particular, by comparing the difference in deviance between nested models, i.e., when one model is a sub-model of the other. The difference in deviance between two nested models has, asymptotically, a χ^2 distribution with degrees of freedom given by the difference in the degrees of freedom associated with the two deviances.

The parameter estimates, $\hat{\beta}$, are asymptotically Normally distributed with variance-covariance matrix:

$$C = R^{-1} R^{-1^T} \quad \text{in the full rank case, otherwise}$$
$$C = P_1 D^{-2} P_1^T.$$

The residuals and influence statistics can also be examined.

The estimated linear predictor $\hat{\eta} = X\hat{\beta}$ can be written as $Hw^{\frac{1}{2}}z$ for an n by n matrix H. The ith diagonal elements of H, h_i, give a measure of the influence of the ith values of the independent variables on the fitted regression model. These are known as leverages.

The fitted values are given by $\hat{\mu} = g^{-1}(\hat{\eta})$ and the deviance residuals by r:

$$r_i = \text{sign}(y_i - \hat{\mu}_i)\sqrt{\text{dev}(y_i, \hat{\mu}_i)}.$$

An option allows prior weights to be used with the model.

If part of the linear predictor can be represented by a variable with a known coefficient then this can be included in the model by using an offset, o:

$$\eta = o + \sum \beta_j x_j.$$

If the model is not of full rank the solution given will be only one of the possible solutions but all solutions will give the same predicted values.

References and Further Reading

Cook R D and Weisberg S (1982) *Residuals and Influence in Regression* Chapman and Hall.

McCullagh P and Nelder J A (1983) *Generalized Linear Models* Chapman and Hall.

Plackett R L (1974) *The Analysis of Categorical Data* Griffin.

See Also

nagdmc_extr_reg	computes fitted values, residuals and leverages for a regression.
nagdmc_loglinear_reg	simplified version of **nagdmc_poisson_reg** using a log link and a restricted set of parameters.
nagdmc_predict_reg	computes predictions given a fitted regression model.
poisson_reg_ex.c	the example calling program.
