

## Euredit binary tree function: waid\_tree

### 1 Purpose

**waid\_tree** computes a binary decision tree by using a robust ANOVA criterion.

### 2 Specification

```
#include <euredit_sys.h>
```

```
void waid_tree (long rec1, long nvar, long nrec, long dblk, double data[],
               void (*dfun) (long, long, double [], int *), long ninde,
               long inde[], long dep, long ws, double w[], long ncat[],
               double tol, long maxit, double c, long mns, long mnc,
               long opt_build, int *root, int *info)
```

### 3 Parameters

#### rec1

*Input:* the first data point in the data block.

*Constraint:* **rec1**  $\geq 0$ .

#### nvar

*Input:* the number of variables in the data.

*Constraint:* **nvar**  $\geq 1$ .

#### nrec

*Input:* the number of consecutive records, beginning at **rec1**, used in the calculations.

*Constraints:* **nrec**  $> 1$  and **(rec1 + nrec)**  $\leq$  **dblk**.

#### dblk

*Input:* the number of records in the data block.

*Constraint:* **dblk**  $> 1$ .

#### data[dblk\*nvar]

*Input:* the element **data**[ $i * \mathbf{nvar} + j$ ] contains the (clean) data value for the  $j$ th variable of the  $i$ th data point, for  $j = 0, 1, \dots, \mathbf{nvar} - 1$ ; for  $i = 0, 1, \dots, \mathbf{dblk} - 1$ .

*Constraint:* if **dfun** is a valid pointer, the value of **data** must be NULL.

#### dfun

*Input:* the function **dfun**, supplied by the user, must return the **dblk** data records starting at record **rec1**.

*Constraint:* if the value of **dfun** is NULL, **data** must be a valid pointer.

The specification of **dfun** is:

```
void dfun (long rec1, long dblk, double x[], int *error)
    rec1
        Input: the first record to be returned.
    dblk
        Input: the number of records to be returned.
    x
        Output: x[ $i * \mathbf{nvar} + j$ ] is the value for the  $j$ th variable, for  $j = 0, 1, \dots, \mathbf{nvar} - 1$ ;
        for  $i = \mathbf{rec1}, \mathbf{rec1} + 1, \dots, \mathbf{rec1} + \mathbf{nrec} - 1$ .
    error
        Output: if the value pointed to by error on return is greater than 100, the function
        will terminate immediately and info will point to this value.
```

**ninde**

*Input:* the number of independent variables in the data.

*Constraint:*  $1 \leq \mathbf{ninde} < \mathbf{nvar}$ .

**inde[ninde]**

*Input:* an array describing the independent variables in the data.

*Constraint:*  $0 \leq \mathbf{inde}[i] < \mathbf{nvar}$ .

**dep**

*Input:* the dependent variable in the data.

*Constraint:*  $0 \leq \mathbf{dep} < \mathbf{nvar}$ .

**ws**

*Input:* the value of **ws** describes the robust weighting scheme:

1 signifies ordinary least squares and sets each weight value equal to one.

2 signifies the use of Andrew's sine wave weight function.

3 signifies the use of Tukey's bi-weight function.

4 signifies the use of Huber's weight function.

*Constraint:*  $\mathbf{ws} \in \{1, 2, 3, 4\}$ .

**w[nrec]**

*Output:*  $\mathbf{w}[i]$  is the sum of weight values computed for the  $i$ th data point.

**ncat[nvar]**

*Input:*  $\mathbf{ncat}[i]$  describes the number of categories in the  $i$ th variable, for  $i = 0, 1, \dots, \mathbf{nvar}$ . If the  $i$ th variable is a continuous, then  $\mathbf{ncat}[i]$  should be set equal to zero.

*Constraints:*  $\mathbf{ncat}[i] \geq 0$ , for  $i = 0, 1, \dots, \mathbf{nvar}$ . If the  $i$ th variable is categorical, its data must be labelled by  $1, 2, \dots, \mathbf{ncat}[i]$ .

**tol**

*Input:* the tolerance used in the test for convergence in the robust estimation of weights.

*Constraint:*  $\mathbf{tol} > 0.0$ .

**maxit**

*Input:* the maximum number of iterations to use in the robust weights calculation.

*Constraint:*  $\mathbf{maxit} > 0$ .

**c**

*Input:* the value of the constant in the Andrew's sine, Tukey and Huber weight functions.

*Constraint:*  $\mathbf{c} > 0.0$ .

**mns**

*Input:* the minimum number of observations that can be used to split a node into its two children.

*Constraint:*  $0 \leq \mathbf{mns} \leq \mathbf{nrec}$ .

**mnc**

*Input:* the minimum number of observations at each node.

*Constraint:*  $\mathbf{mnc} \leq \mathbf{mns} * 0.5$ .

**opt\_build**

*Input:* by setting **opt\_build** equal to 1 each leaf node stores an array of values of the dependent variable for its data.

**root**

*Output:* the value pointed to by **root** is an integer cast of the root of the decision tree.

**info**

*Output:* the value pointed to by **info** gives information on the success of the function call:

0: the function successfully completed its task.

-39: a calculation of robust weights failed to converge.

$i$ ;  $i = 1, 2, \dots, 19$ : the specification of the  $i$ th formal parameter was incorrect.

99: the function failed to allocate enough memory.

100: an internal error occurred during the execution of the function.  
> 100: an error occurred in a function specified by the user.  
-999: the routine does not have a valid licence.

---