NAG Toolbox for MATLAB

g10ac

1 Purpose

g10ac estimates the values of the smoothing parameter and fits a cubic smoothing spline to a set of data.

2 Syntax

```
[yhat, c, rss, df, res, h, crit, rho, ifail] = g10ac(method, weight, x, y, wt, crit, tol, 'n', n, 'u', u, 'maxcal', maxcal)
```

3 Description

For a set of n observations (x_i, y_i), for $i = 1, 2, \ldots, n$, the spline provides a flexible smooth function for situations in which a simple polynomial or non-linear regression model is not suitable.

Cubic smoothing splines arise as the unique real-valued solution function f, with absolutely continuous first derivative and squared-integrable second derivative, which minimizes

$$\sum_{i=1}^{n} w_i (y_i - f(x_i))^2 + \rho \int_{-\infty}^{\infty} \left(f''(x)\right)^2 dx,$$

where w_i is the (optional) weight for the ith observation and ρ is the smoothing parameter. This criterion consists of two parts: the first measures the fit of the curve and the second the smoothness of the curve. The value of the smoothing parameter ρ weights these two aspects; larger values of ρ give a smoother fitted curve but, in general, a poorer fit. For details of how the cubic spline can be fitted see Hutchinson and de Hoog 1985 and Reinsch 1967.

The fitted values, $\hat{y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)^{\mathrm{T}}$, and weighted residuals, r_i, can be written as:

$$\hat{y} = Hy \qquad \text{and} \qquad r_i = \sqrt{w_i}(y_i - \hat{y}_i)$$

for a matrix H. The residual degrees of freedom for the spline is $\mathrm{trace}(I - H)$ and the diagonal elements of H are the leverages.

The parameter ρ can be estimated in a number of ways.

(i) The degrees of freedom for the spline can be specified, i.e., find ρ such that $\mathrm{trace}(H) = \nu_0$ for given ν_0.

(ii) Minimize the cross-validation (CV), i.e., find ρ such that the CV is minimized, where

$$\mathrm{CV} = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} \left[\frac{r_i}{1 - h_{ii}}\right]^2.$$

(iii) Minimize the generalized cross-validation (GCV), i.e., find ρ such that the GCV is minimized, where

$$\mathrm{GCV} = \frac{n^2}{\sum_{i=1}^{n} w_i} \left[\frac{\sum_{i=1}^{n} r_i^2}{\left(\sum_{i=1}^{n} (1 - h_{ii})\right)^2}\right].$$

g10ac requires the x_i to be strictly increasing. If two or more observations have the same x_i value then they should be replaced by a single observation with y_i equal to the (weighted) mean of the y values and weight, w_i, equal to the sum of the weights. This operation can be performed by g10za.

The algorithm is based on Hutchinson 1986. c05az is used to solve for ρ given ν_0 and the method of e04ab is used to minimize the GCV or CV.

4 References

Hastie T J and Tibshirani R J 1990 *Generalized Additive Models* Chapman and Hall

Hutchinson M F 1986 Algorithm 642: A fast procedure for calculating minimum cross-validation cubic smoothing splines *ACM Trans. Math. Software* **12** 150–153

Hutchinson M F and de Hoog F R 1985 Smoothing noisy data with spline functions *Numer. Math.* **47** 99–106

Reinsch C H 1967 Smoothing by spline functions *Numer. Math.* **10** 177–183

5 Parameters

5.1 Compulsory Input Parameters

1: **method – string**

Indicates whether the smoothing parameter is to be found by minimization of the CV or GCV functions, or by finding the smoothing parameter corresponding to a specified degrees of freedom value.

method = 'C'

Cross-validation is used.

method = 'D'

The degrees of freedom are specified.

method = 'G'

Generalized cross-validation is used.

Constraint: **method** = 'C', 'D' or 'G'.

2: **weight – string**

Indicates whether user-defined weights are to be used.

weight = 'W'

User-defined weights should be supplied in **wt**.

weight = 'U'

The data is treated as unweighted.

Constraint: **weight** = 'W' or 'U'.

3: **x(n) – double array**

The distinct and ordered values x_i, for $i = 1, 2, \ldots, n$.

Constraint: $\mathbf{x}(i) < \mathbf{x}(i+1)$, for $i = 1, 2, \ldots, n-1$.

4: **y(n) – double array**

The values y_i, for $i = 1, 2, \ldots, n$.

5: **wt(∗) – double array**

Note: the dimension of the array **wt** must be at least 1 if **weight** = 'U' and at least **n** if **weight** = 'W'.

If **weight** = 'W', **wt** must contain the n weights.

If **weight** = 'U', **wt** is not referenced and unit weights are assumed.

Constraint: if **weight** = 'W', $\mathbf{wt}(i) > 0.0$, for $i = 1, 2, \ldots, n$.

6: **crit – double scalar**

If **method** = 'D', the required degrees of freedom for the spline.

If **method** = 'C' or 'G', **crit** need not be set.

Constraint: $2.0 < \mathbf{crit} \leq \mathbf{n}$.

7: **tol – double scalar**

The accuracy to which the smoothing parameter **rho** is required. **tol** should be preferably not much less than $\sqrt{\epsilon}$, where ϵ is the ***machine precision***.

Constraint: **tol** $\geq$ ***machine precision***.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the arrays **x**, **y**, **yhat**, **res**, **h**. (An error is raised if these dimensions are not equal.)

n, the number of observations.

Constraint: $\mathbf{n} \geq 3$.

2: **u – double scalar**

The upper bound on the smoothing parameter. See Section 8 for details on how this parameter is used.

Constraint: $\mathbf{u} > \mathbf{tol}$.

Suggested value: $\mathbf{u} = 1000.0$.

Default: 1000.0

3: **maxcal – int32 scalar**

The maximum number of spline evaluations to be used in finding the value of ρ.

Constraint: $\mathbf{maxcal} \geq 3$.

Suggested value: $\mathbf{maxcal} = 30$.

Default: 30

5.3 Input Parameters Omitted from the MATLAB Interface

ldc, wk

5.4 Output Parameters

1: **yhat(n) – double array**

The fitted values, $\hat{y}_i$, for $i = 1, 2, \ldots, n$.

2: **c(ldc,3) – double array**

The spline coefficients. More precisely, the value of the spline approximation at t is given by $((\mathbf{c}(i,3) \times d + \mathbf{c}(i,2)) \times d + \mathbf{c}(i,1)) \times d + \hat{y}_i$, where $x_i \leq t < x_{i+1}$ and $d = t - x_i$.

3: **rss – double scalar**

The (weighted) residual sum of squares.

4: **df – double scalar**

The residual degrees of freedom. If **method** = 'D' this will be $n -$ **crit** to the required accuracy.

5: **res(n) – double array**

The (weighted) residuals, r_i, for $i = 1, 2, \ldots, n$.

6: **h(n) – double array**

The leverages, h_{ii}, for $i = 1, 2, \ldots, n$.

7: **crit – double scalar**

If **method** = 'C', the value of the cross-validation, or if **method** = 'G', the value of the generalized cross-validation function, evaluated at the value of ρ returned in **rho**.

8: **rho – double scalar**

The smoothing parameter, ρ.

9: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail = 1

On entry, **n** < 3,
or **ldc** $<$ **n** $- 1$,
or **method** $\neq$ 'C', 'G' or 'D',
or **weight** $\neq$ 'W' or 'U',
or **method** = 'D' and **crit** ≤ 2.0,
or **method** = 'D' and **crit** $>$ **n**,
or **tol** $<$ ***machine precision***,
or **u** $\leq$ **tol**,
or **maxcal** < 3.

ifail = 2

On entry, **weight** = 'W' and at least one element of **wt** ≤ 0.0.

ifail = 3

On entry, $\mathbf{x}(i) \geq \mathbf{x}(i+1)$, for some i, $i = 1, 2, \ldots, n-1$.

ifail = 4

method = 'D' and the required value of ρ for specified degrees of freedom $> U$. Try a larger value of **u**; see Section 8.

ifail = 5

method = 'D' and the accuracy given by **tol** cannot be achieved. Try increasing the value of **tol**.

ifail = 6

A solution to the accuracy given by **tol** has not been achieved in **maxcal** iterations. Try increasing the value of **tol** and/or **maxcal**.

ifail = 7

method = 'C' or 'G' and the optimal value of $\rho >$ **u**. Try a larger value of **u**; see Section 8.

7 Accuracy

When minimizing the cross-validation or generalized cross-validation, the error in the estimate of ρ should be within $\pm 3(\textbf{tol} \times \textbf{rho} + \textbf{tol})$. When finding ρ for a fixed number of degrees of freedom the error in the estimate of ρ should be within $\pm 2 \times \textbf{tol} \times \max(1, \textbf{rho})$.

Given the value of ρ, the accuracy of the fitted spline depends on the value of ρ and the position of the x values. The values of $x_i - x_{i-1}$ and w_i are scaled and ρ is transformed to avoid underflow and overflow problems.

8 Further Comments

The time to fit the spline for a given value of ρ is of order n.

When finding the value of ρ that gives the required degrees of freedom, the algorithm examines the interval 0.0 to **u**. For small degrees of freedom the value of ρ can be large, as in the theoretical case of two degrees of freedom when the spline reduces to a straight line and ρ is infinite. If the CV or GCV is to be minimized then the algorithm searches for the minimum value in the interval 0.0 to **u**. If the function is decreasing in that range then the boundary-value of **u** will be returned. In either case, the larger the value of **u** the more likely is the interval to contain the required solution, but the process will be less efficient.

Regression splines with a small $(< n)$ number of knots can be fitted by e02ba and e02be.

9 Example

```
method = 'D';
weight = 'W';
x = [0.9;
     1;
     1.8;
     1.9;
     2.2;
     4.2;
     4.8;
     5.1;
     5.2;
     5.8;
     6.9;
     7.9;
     8.1;
     8.5;
     8.8;
     8.9;
     9.8;
     9.9;
     10.4;
     10.5;
     10.6;
     10.8;
     11;
     11.1;
     11.3;
     11.5;
     11.8;
     11.9;
     12.4;
     12.5;
     12.7;
     12.8;
     13.2;
```

```
      13.8;
      14.5;
      15.5;
      15.6];
y = [3;
      3.9;
      3.4;
      3.7;
      3.9;
      5.1;
      4.2;
      4.6;
      4.85;
      5.6;
      5.1;
      4.8;
      5.2;
      5.3;
      4.1;
      4.9;
      4.8;
      4.9;
      5;
      5.2;
      5;
      5.1;
      4.4;
      4.9;
      5.1;
      5.5;
      4.6;
      5.1;
      5.2;
      4.1;
      3.4;
      6.6;
      5.3;
      3.7;
      5.7;
      4.9;
      4.9];
wt = [1;
      1;
      1;
      1;
      1;
      1;
      2;
      1;
      2;
      1;
      1;
      2;
      1;
      1;
      1;
      1;
      1;
      1;
      1;
      1;
      2;
      1;
      1;
      2;
      1;
      1;
      1;
      1;
      1;
      1;
```

```
     1;
     1;
     1;
     2;
     1;
     1;
     1;
     1];
crit = 12;
tol = 0.001;
[yhat, c, rss, df, res, h, critOut, rho, ifail] = ...
    g10ac(method, weight, x, y, wt, crit, tol)
```

```
yhat =
    3.3732
    3.4061
    3.6424
    3.6856
    3.8395
    4.6142
    4.5762
    4.7146
    4.7830
    5.1926
    5.1836
    4.9578
    4.9307
    4.8452
    4.7629
    4.7475
    4.8500
    4.8745
    4.9704
    4.9774
    4.9788
    4.9702
    4.9614
    4.9636
    4.9753
    4.9747
    4.9297
    4.9112
    4.8517
    4.8570
    4.9002
    4.9316
    4.9547
    4.7972
    5.0757
    4.9794
    4.9462
c =
    0.3349         0   -0.6265
    0.3161   -0.1880    0.2026
    0.4044    0.2984   -0.2043
    0.4580    0.2371   -0.1801
    0.5516    0.0750   -0.0785
   -0.0910   -0.3962    0.7370
    0.2295    0.9304   -0.5260
    0.6457    0.4569   -0.7184
    0.7155    0.2414   -0.4933
    0.4724   -0.6466    0.1906
   -0.2583   -0.0177    0.0503
   -0.1430    0.1330   -0.4795
   -0.1474   -0.1547   -0.0273
   -0.2842   -0.1875    0.7362
   -0.1979    0.4751   -0.3768
   -0.1142    0.3621   -0.1208
    0.2440    0.0359   -0.2047
    0.2450   -0.0255   -0.1619
```

```
    0.0981   -0.2684   -0.1123
    0.0411   -0.3020    0.2614
   -0.0115   -0.2236    0.3326
   -0.0611   -0.0241    0.5505
   -0.0046    0.3062   -0.3920
    0.0449    0.1886   -0.6057
    0.0476   -0.1748   -0.3963
   -0.0699   -0.4126    0.4856
   -0.1863    0.0244   -0.0679
   -0.1834    0.0041    0.2490
    0.0074    0.3776    0.8338
    0.1079    0.6277   -0.4371
    0.3066    0.3655   -2.9556
    0.2910   -0.5212   -0.1547
   -0.2002   -0.7068    1.0048
    0.0368    1.1018   -0.8371
    0.3489   -0.6561    0.2109
   -0.3305   -0.0233    0.0776
rss =
   10.3514
df =
   24.9997
res =
   -0.3732
    0.4939
   -0.2424
    0.0144
    0.0605
    0.4858
   -0.5320
   -0.1146
    0.0948
    0.4074
   -0.0836
   -0.2231
    0.2693
    0.4548
   -0.6629
    0.1525
   -0.0500
    0.0255
    0.0296
    0.2226
    0.0300
    0.1298
   -0.5614
   -0.0900
    0.1247
    0.5253
   -0.3297
    0.1888
    0.3483
   -0.7570
   -1.5002
    1.6684
    0.4884
   -1.0972
    0.6243
   -0.0794
   -0.0462
h =
    0.5336
    0.4273
    0.3128
    0.3127
    0.4477
    0.5640
    0.4418
    0.1889
    0.4072
```

```
    0.4552
    0.5923
    0.5299
    0.2345
    0.2446
    0.2707
    0.2924
    0.3006
    0.2765
    0.1728
    0.1542
    0.2849
    0.1356
    0.1373
    0.2836
    0.1617
    0.1857
    0.2126
    0.2202
    0.2057
    0.1957
    0.1889
    0.1932
    0.4880
    0.4077
    0.5592
    0.4455
    0.5352
critOut =
    12
rho =
    2.6799
ifail =
           0
```