NAG Library Routine Document

G10BAF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

G10BAF performs kernel density estimation using a Gaussian kernel.

2 Specification

```
SUBROUTINE G10BAF (N, X, WINDOW, SLO, SHI, NS, SMOOTH, T, USEFFT, FFT, IFAIL)

INTEGER             N, NS, IFAIL
REAL (KIND=nag_wp) X(N), WINDOW, SLO, SHI, SMOOTH(NS), T(NS), FFT(NS)
LOGICAL             USEFFT
```

3 Description

Given a sample of n observations, $x_1, x_2, \ldots, x_n$, from a distribution with unknown density function, $f(x)$, an estimate of the density function, $\hat{f}(x)$, may be required. The simplest form of density estimator is the histogram. This may be defined by:

$$\hat{f}(x) = \tfrac{1}{nh} n_j, \qquad a + (j-1)h < x < a + jh, \qquad j = 1, 2, \ldots, n_s,$$

where n_j is the number of observations falling in the interval $a + (j-1)h$ to $a + jh$, a is the lower bound to the histogram and $b = n_s h$ is the upper bound. The value h is known as the window width. To produce a smoother density estimate a kernel method can be used. A kernel function, $K(t)$, satisfies the conditions:

$$\int_{-\infty}^{\infty} K(t)\, dt = 1 \qquad \text{and} \qquad K(t) \geq 0.$$

The kernel density estimator is then defined as

$$\hat{f}(x) = \tfrac{1}{nh} \sum_{i=1}^{n} K\Big(\frac{x - x_i}{h}\Big).$$

The choice of K is usually not important but to ease the computational burden use can be made of the Gaussian kernel defined as

$$K(t) = \frac{1}{\sqrt{2\pi}} e^{-t^2/2}.$$

The smoothness of the estimator depends on the window width h. The larger the value of h the smoother the density estimate. The value of h can be chosen by examining plots of the smoothed density for different values of h or by using cross-validation methods (see Silverman (1990)).

Silverman (1982) and Silverman (1990) show how the Gaussian kernel density estimator can be computed using a fast Fourier transform (FFT). In order to compute the kernel density estimate over the range a to b the following steps are required.

(i) Discretize the data to give n_s equally spaced points t_l with weights ξ_l (see Jones and Lotwick (1984)).

(ii) Compute the FFT of the weights ξ_l to give Y_l.

(iii) Compute $\zeta_l = e^{-\frac{1}{2}h^2 s_l^2} Y_l$ where $s_l = 2\pi l/(b-a)$.

(iv) Find the inverse FFT of ζ_l to give $\hat{f}(x)$.

To compute the kernel density estimate for further values of h only steps (iii) and (iv) need be repeated.

4 References

Jones M C and Lotwick H W (1984) Remark AS R50. A remark on algorithm AS 176 *Appl. Statist.* **33** 120–122

Silverman B W (1982) Algorithm AS 176. Kernel density estimation using the fast Fourier transform *Appl. Statist.* **31** 93–99

Silverman B W (1990) *Density Estimation* Chapman and Hall

5 Parameters

1: N – INTEGER *Input*

On entry: n, the number of observations in the sample.

Constraint: N > 0.

2: X(N) – REAL (KIND=nag_wp) array *Input*

On entry: the n observations, x_i, for $i = 1, 2, \ldots, n$.

3: WINDOW – REAL (KIND=nag_wp) *Input*

On entry: h, the window width.

Constraint: WINDOW > 0.0.

4: SLO – REAL (KIND=nag_wp) *Input*

On entry: a, the lower limit of the interval on which the estimate is calculated. For most applications SLO should be at least three window widths below the lowest data point.

Constraint: SLO $<$ SHI.

5: SHI – REAL (KIND=nag_wp) *Input*

On entry: b, the upper limit of the interval on which the estimate is calculated. For most applications SHI should be at least three window widths above the highest data point.

6: NS – INTEGER *Input*

On entry: the number of points at which the estimate is calculated, n_s.

Constraints:

NS ≥ 2;
The largest prime factor of NS must not exceed 19, and the total number of prime factors of NS, counting repetitions, must not exceed 20.

7: SMOOTH(NS) – REAL (KIND=nag_wp) array *Output*

On exit: the n_s values of the density estimate, $\hat{f}(t_l)$, for $l = 1, 2, \ldots, n_s$.

8: T(NS) – REAL (KIND=nag_wp) array *Output*

On exit: the points at which the estimate is calculated, t_l, for $l = 1, 2, \ldots, n_s$.

9: USEFFT – LOGICAL *Input*

On entry: must be set to .FALSE. if the values of Y_l are to be calculated by G10BAF and to .TRUE. if they have been computed by a previous call to G10BAF and are provided in FFT. If USEFFT = .TRUE. then the arguments N, SLO, SHI, NS and FFT must remain unchanged from the previous call to G10BAF with USEFFT = .FALSE..

10: FFT(NS) – REAL (KIND=nag_wp) array *Input/Output*

On entry: if USEFFT = .TRUE., then FFT must contain the fast Fourier transform of the weights of the discretized data, ξ_l, for $l = 1, 2, \ldots, n_s$. Otherwise FFT need not be set.

On exit: the fast Fourier transform of the weights of the discretized data, ξ_l, for $l = 1, 2, \ldots, n_s$.

11: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, N $\leq$ 0,
or NS $<$ 2,
or SHI $\leq$ SLO,
or WINDOW $\leq$ 0.0.

IFAIL = 2

On entry, G10BAF has been called with USEFFT = .TRUE. but the routine has not been called previously with USEFFT = .FALSE.,
or G10BAF has been called with USEFFT = .TRUE. but some of the arguments N, SLO, SHI, NS have been changed since the previous call to G10BAF with USEFFT = .FALSE..

IFAIL = 3

On entry, at least one prime factor of NS is greater than 19 or NS has more than 20 prime factors (see C06EAF).

IFAIL = 4

On entry, the interval given by SLO to SHI does not extend beyond three window widths at either extreme of the dataset. This may distort the density estimate in some cases.

7 Accuracy

See Jones and Lotwick (1984) for a discussion of the accuracy of this method.

8 Further Comments

The time for computing the weights of the discretized data is of order n, while the time for computing the FFT is of order $n_s \log(n_s)$, as is the time for computing the inverse of the FFT.

9 Example

A sample of 1000 standard Normal $(0, 1)$ variates are generated using G05SKF and the density estimated on 100 points with a window width of 0.1. The resulting estimate of the density function is plotted using G01AGF.

9.1 Program Text

```
    PROGRAM g10bafe

!       G10BAF Example Program Text

!       Mark 23 Release. NAG Copyright 2011.

!       .. Use Statements ..
        USE nag_library, ONLY : g01agf, g10baf, nag_wp
!       .. Implicit None Statement ..
        IMPLICIT NONE
!       .. Parameters ..
        INTEGER, PARAMETER              :: nin = 5, nout = 6
!       .. Local Scalars ..
        REAL (KIND=nag_wp)              :: shi, slo, window
        INTEGER                         :: ifail, n, ns, nstepx, nstepy
        LOGICAL                         :: usefft
!       .. Local Arrays ..
        REAL (KIND=nag_wp), ALLOCATABLE :: fft(:), smooth(:), t(:), x(:)
        INTEGER, ALLOCATABLE            :: isort(:)
!       .. Executable Statements ..
        WRITE (nout,*) 'G10BAF Example Program Results'
        WRITE (nout,*)
        FLUSH (nout)

!       Skip heading in data file
        READ (nin,*)

!       Read in density estimation information
        READ (nin,*) window, slo, shi, ns

!       Read in plotting information
        READ (nin,*) nstepx, nstepy

!       Read in the size of the dataset
        READ (nin,*) n

        ALLOCATE (smooth(ns),t(ns),fft(ns),x(n),isort(ns))

!       Read in data
        READ (nin,*) x(1:n)

!       Perform kernel density estimation
        usefft = .FALSE.
        ifail = 0
        CALL g10baf(n,x,window,slo,shi,ns,smooth,t,usefft,fft,ifail)

!       Display smoothed data
        ifail = 0
        CALL g01agf(t,smooth,ns,isort,nstepx,nstepy,ifail)

    END PROGRAM g10bafe
```

9.2 Program Data

```
G10BAF Example Program Data
0.1  -4.0  4.0  100               :: WINDOW,SLO,SHI,NS
40  20                            :: NSTEPX,NSTEPY
100                               :: N
 0.114 -0.232 -0.570  1.853 -0.994
-0.374 -1.028  0.509  0.881 -0.453
 0.588 -0.625 -1.622 -0.567  0.421
```

```
-0.475  0.054  0.817  1.015  0.608
-1.353 -0.912 -1.136  1.067  0.121
-0.075 -0.745  1.217 -1.058 -0.894
 1.026 -0.967 -1.065  0.513  0.969
 0.582 -0.985  0.097  0.416 -0.514
 0.898 -0.154  0.617 -0.436 -1.212
-1.571  0.210 -1.101  1.018 -1.702
-2.230 -0.648 -0.350  0.446 -2.667
 0.094 -0.380 -2.852 -0.888 -1.481
-0.359 -0.554  1.531  0.052 -1.715
 1.255 -0.540  0.362 -0.654 -0.272
-1.810  0.269 -1.918  0.001  1.240
-0.368 -0.647 -2.282  0.498  0.001
-3.059 -1.171  0.566  0.948  0.925
 0.825  0.130  0.930  0.523  0.443
-0.649  0.554 -2.823  0.158 -1.180
 0.610  0.877  0.791 -0.078  1.412  :: End of X
```

9.3 Program Results

```
 G10BAF Example Program Results
```