NAG Library Routine Document

G10BBF

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

G10BBF performs kernel density estimation using a Gaussian kernel.

2 Specification

```
SUBROUTINE G10BBF (N, X, WTYPE, WINDOW, SLO, SHI, NS, SMOOTH, T, FCALL,    &
                   RCOMM, IFAIL)

INTEGER            N, WTYPE, NS, FCALL, IFAIL
REAL (KIND=nag_wp) X(N), WINDOW, SLO, SHI, SMOOTH(NS), T(NS),              &
                   RCOMM(NS+20)
```

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:

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

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, $b = n_s h$ is the upper bound and n_s is the total number of intervals. 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:

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

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. Kernel density estimation using the Fast Fourier Transform *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.

If FCALL $= 0$, N must be unchanged since the last call to G10BBF.

Constraint: N > 0.

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

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

If FCALL $= 0$, X must be unchanged since the last call to G10BBF.

3: WTYPE – INTEGER *Input*

On entry: how the window width, h, is to be calculated:

WTYPE $= 1$
h is supplied in WINDOW.

WTYPE $= 2$
h is to be calculated from the data, with

$$h = m \times \left(\frac{0.9 \times \min(q_{75} - q_{25}, \sigma)}{n^{0.2}} \right)$$

where $q_{75} - q_{25}$ is the inter-quartile range and σ the standard deviation of the sample, x, and m is a multipler supplied in WINDOW. The 25% and 75% quartiles, q_{25} and q_{75}, are calculated using G01AMF. This is the "rule-of-thumb" suggested by Silverman (1990).

Suggested value: WTYPE $= 2$ and WINDOW $= 1.0$

Constraint: WTYPE $= 1$ or 2.

4: WINDOW – REAL (KIND=nag_wp) *Input/Output*

On entry: if WTYPE $= 1$, then h, the window width. Otherwise, m, the multiplier used in the calculation of h.

Suggested value: WINDOW $= 1.0$ and WTYPE $= 2$

On exit: h, the window width actually used.

Constraint: WINDOW > 0.0.

5: SLO – REAL (KIND=nag_wp) *Input/Output*

On entry: if SLO $<$ SHI then a, the lower limit of the interval on which the estimate is calculated. Otherwise, a and b, the lower and upper limits of the interval, are calculated as follows:

$$\begin{aligned} a &= \min_i\{x_i\} - \text{SLO} \times h \\ b &= \max_i\{x_i\} + \text{SLO} \times h \end{aligned}$$

where h is the window width.

For most applications a should be at least three window widths below the lowest data point.

If FCALL $= 0$, SLO must be unchanged since the last call to G10BBF.

Suggested value: SLO $= 3.0$ and SHI $= 0.0$ which would cause a and b to be set 3 window widths below and above the lowest and highest data points respectively.

On exit: a, the lower limit actually used.

6: SHI – REAL (KIND=nag_wp) *Input/Output*

On entry: if SLO $<$ SHI then b, the upper limit of the interval on which the estimate is calculated. Otherwise a value for b is calculated from the data as stated in the description of SLO and the value supplied in SHI is not used.

For most applications b should be at least three window widths above the highest data point.

If FCALL $= 0$, SHI must be unchanged since the last call to G10BBF.

On exit: b, the upper limit actually used.

7: NS – INTEGER *Input*

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

If FCALL $= 0$, NS must be unchanged since the last call to G10BBF.

Suggested value: NS $= 512$

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.

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

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

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

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

10: FCALL – INTEGER *Input*

On entry: If FCALL $= 1$ then the values of Y_l are to be calculated by this call to G10BBF, otherwise it is assumed that the values of Y_l were calculated by a previous call to this routine and the relevant information is stored in RCOMM.

Constraint: FCALL $= 0$ or 1.

11: RCOMM(NS + 20) – REAL (KIND=nag_wp) array *Communication Array*

On entry: communication array, used to store information between calls to G10BBF.

If FCALL $= 0$, RCOMM must be unchanged since the last call to G10BBF.

On exit: the last NS elements of RCOMM contain the fast Fourier transform of the weights of the discretized data, that is RCOMM$(l + 20) = Y_l$, for $l = 1, 2, \ldots, n_s$.

12: 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.

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, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

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

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).

Note: G10BBF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL $= 11$

On entry, N $= \langle value \rangle$.
Constraint: N > 0.

IFAIL $= 12$

On entry, N $= \langle value \rangle$.
On entry at previous call, N $= \langle value \rangle$.
Constraint: if FCALL $= 0$, N must be unchanged since previous call.

IFAIL $= 31$

On entry, WTYPE $= \langle value \rangle$.
Constraint: WTYPE $= 1$ or 2.

IFAIL $= 41$

On entry, WINDOW $= \langle value \rangle$.
Constraint: WINDOW > 0.0.

IFAIL $= 51$

On entry, SLO $= \langle value \rangle$.
On exit from previous call, SLO $= \langle value \rangle$.
Constraint: if FCALL $= 0$, SLO must be unchanged since previous call.

IFAIL $= 61$

On entry, SLO $= \langle value \rangle$ and SHI $= \langle value \rangle$.
On entry, min(X) $= \langle value \rangle$ and max(X) $= \langle value \rangle$.
Expected values of at least $\langle value \rangle$ and $\langle value \rangle$ for SLO and SHI.
All output values have been returned.

IFAIL $= 62$

On entry, SHI $= \langle value \rangle$.
On exit from previous call, SHI $= \langle value \rangle$.
Constraint: if FCALL $= 0$, SHI must be unchanged since previous call.

IFAIL = 71

On entry, NS = ⟨*value*⟩.
Constraint: NS ≥ 2.

IFAIL = 72

On entry, NS = ⟨*value*⟩.
Constraint: Largest prime factor of NS must not exceed 19.

IFAIL = 73

On entry, NS = ⟨*value*⟩.
Constraint: Total number of prime factors of NS must not exceed 20.

IFAIL = 74

On entry, NS = ⟨*value*⟩.
On entry at previous call, NS = ⟨*value*⟩.
Constraint: if FCALL = 0, NS must be unchanged since previous call.

IFAIL = 101

On entry, FCALL = ⟨*value*⟩.
Constraint: FCALL = 0 or 1.

IFAIL = 111

RCOMM has been corrupted between calls.

IFAIL = −99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL = −399

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL = −999

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

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

8 Parallelism and Performance

Not applicable.

9 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.

10 Example

Data is read from a file and the density estimated. The first 20 values are then printed.

10.1 Program Text

```
    Program g10bbfe
!     G10BBF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: g10bbf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: shi, slo, window
      Integer                         :: fcall, i, ifail, n, ns, wtype
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: rcomm(:), smooth(:), t(:), x(:)
!     .. Intrinsic Procedures ..
      Intrinsic                       :: min
!     .. Executable Statements ..
      Write (nout,*) 'G10BBF Example Program Results'
      Write (nout,*)

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

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

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

      Allocate (smooth(ns),t(ns),rcomm(ns+20),x(n))

!     Only calling the routine once
      fcall = 1

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

!     Perform kernel density estimation
      ifail = 0
      Call g10bbf(n,x,wtype,window,slo,shi,ns,smooth,t,fcall,rcomm,ifail)

!     Display the results
      Write (nout,99998) 'Window Width Used = ', window
      Write (nout,99997) 'Interval = (', slo, ',', shi, ')'
      Write (nout,*)
      Write (nout,99999) 'First ', min(20,ns), ' output values:'
      Write (nout,*)
      Write (nout,*) '      Time        Density'
      Write (nout,*) '      Point       Estimate'
      Write (nout,*) ' --------------------------'
      Do i = 1, min(20,ns)
        Write (nout,99996) t(i), smooth(i)
      End Do

99999 Format (A,I0,A)
99998 Format (A,E11.4)
99997 Format (A,E11.4,A,E11.4,A)
99996 Format (1X,E13.4,1X,E13.4)
    End Program g10bbfe
```

10.2 Program Data

```
G10BBF Example Program Data
2  1.0  3.0  0.0  512              :: WTYPE,WINDOW,SLO,SHI,NS
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
```

10.3 Program Results

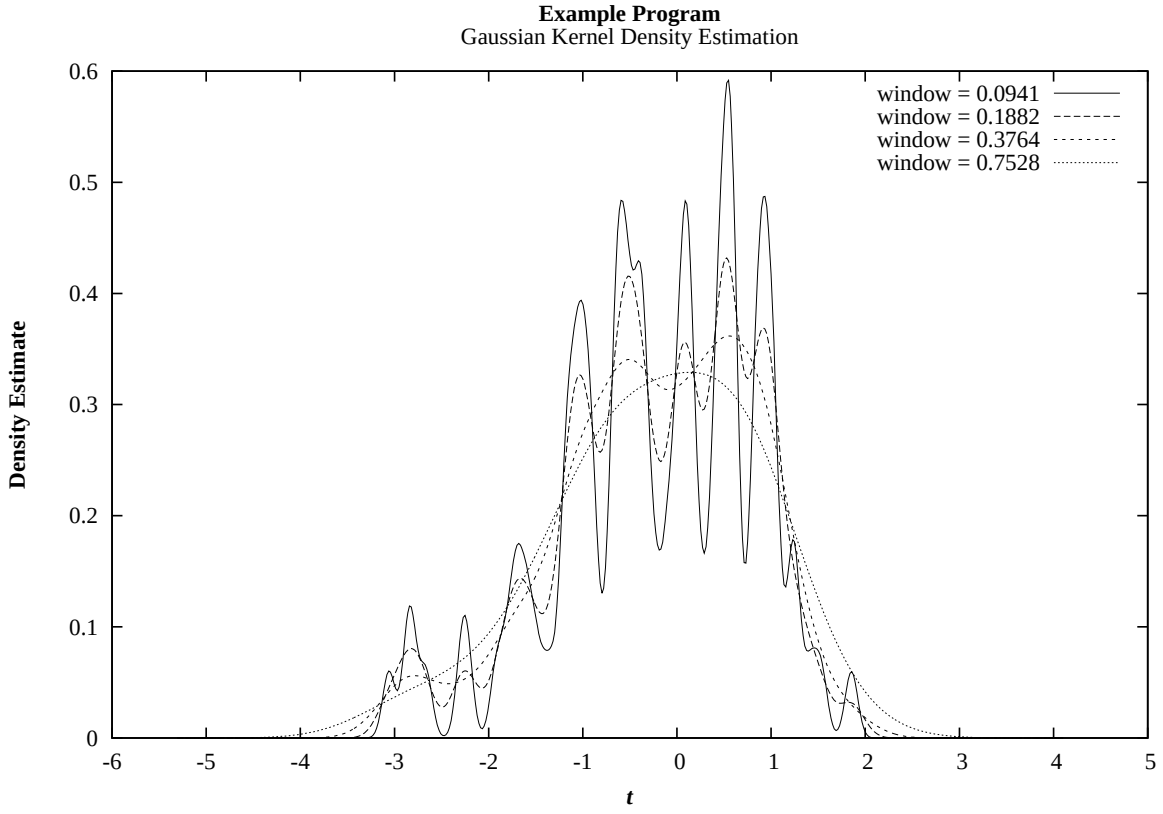
```
 G10BBF Example Program Results

Window Width Used =  0.3764E+00
Interval = (-0.4188E+01, 0.2982E+01)

First 20 output values:
```

Time Point	Density Estimate
-0.4181E+01	0.3828E-05
-0.4167E+01	0.4031E-05
-0.4153E+01	0.4423E-05
-0.4139E+01	0.5021E-05
-0.4125E+01	0.5846E-05
-0.4111E+01	0.6928E-05
-0.4097E+01	0.8305E-05
-0.4083E+01	0.1002E-04
-0.4069E+01	0.1215E-04
-0.4055E+01	0.1474E-04
-0.4041E+01	0.1788E-04
-0.4027E+01	0.2168E-04
-0.4013E+01	0.2624E-04
-0.3999E+01	0.3170E-04
-0.3985E+01	0.3821E-04
-0.3971E+01	0.4596E-04
-0.3957E+01	0.5514E-04
-0.3943E+01	0.6599E-04
-0.3929E+01	0.7877E-04
-0.3915E+01	0.9380E-04

This plot shows the estimated density function for the example data for several window widths.


Example Program
Gaussian Kernel Density Estimation
window = 0.0941
window = 0.1882
window = 0.3764
window = 0.7528
Density Estimate
t
0
0.1
0.2
0.3
0.4
0.5
0.6
-6
-5
-4
-3
-2
-1
0
1
2
3
4
5