NAG Library Routine Document

G13NAF

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

G13NAF detects change points in a univariate time series, that is, the time points at which some feature of the data, for example the mean, changes. Change points are detected using the PELT (Pruned Exact Linear Time) algorithm using one of a provided set of cost functions.

2 Specification

```
SUBROUTINE G13NAF (CTYPE, N, Y, BETA, MINSS, IPARAM, PARAM, NTAU, TAU,     &
                   SPARAM, IFAIL)

INTEGER            CTYPE, N, MINSS, IPARAM, NTAU, TAU(N), IFAIL
REAL (KIND=nag_wp) Y(N), BETA, PARAM(1), SPARAM(2*N+2)
```

3 Description

Let $y_{1:n} = \{y_j : j = 1, 2, \ldots, n\}$ denote a series of data and $\tau = \{\tau_i : i = 1, 2, \ldots, m\}$ denote a set of m ordered (strictly monotonic increasing) indices known as change points with $1 \leq \tau_i \leq n$ and $\tau_m = n$. For ease of notation we also define $\tau_0 = 0$. The m change points, τ, split the data into m segments, with the ith segment being of length n_i and containing $y_{\tau_{i-1}+1:\tau_i}$.

Given a cost function, $C(y_{\tau_{i-1}+1:\tau_i})$ G13NAF solves

$$\underset{m,\tau}{\text{minimize}} \sum_{i=1}^{m} (C(y_{\tau_{i-1}+1:\tau_i}) + \beta) \tag{1}$$

where β is a penalty term used to control the number of change points. This minimization is performed using the PELT algorithm of Killick *et al.* (2012). The PELT algorithm is guaranteed to return the optimal solution to (1) if there exists a constant K such that

$$C\big(y_{(u+1):v}\big) + C\big(y_{(v+1):w}\big) + K \leq C\big(y_{(u+1):w}\big) \tag{2}$$

for all $u < v < w$.

G13NAF supplies four families of cost function. Each cost function assumes that the series, y, comes from some distribution, $D(\Theta)$. The parameter space, $\Theta = \{\theta, \phi\}$ is subdivided into θ containing those parameters allowed to differ in each segment and ϕ those parameters treated as constant across all segments. All four cost functions can then be described in terms of the likelihood function, L and are given by:

$$C\big(y_{(\tau_{i-1}+1):\tau_i}\big) = -2\log L\Big(\hat{\theta}_i, \phi | y_{(\tau_{i-1}+1):\tau_i}\Big)$$

where $\hat{\theta}_i$ is the maximum likelihood estimate of θ within the ith segment. In all four cases setting $K = 0$ satisfies equation (2). Four distributions are available: Normal, Gamma, Exponential and Poisson. Letting

$$S_i = \sum_{j=\tau_{i-1}}^{\tau_i} y_j$$

the log likelihoods and cost functions for the four distributions, and the available subdivisions of the parameter space are:

Normal distribution: $\Theta = \{\mu, \sigma^2\}$

$$-2\log L = \sum_{i=1}^{m} \sum_{j=\tau_{i-1}}^{\tau_i} \log(2\pi) + \log\left(\sigma_i^2\right) + \frac{\left(y_j - \mu_i\right)^2}{\sigma_i^2}$$

Mean changes: $\theta = \{\mu\}$

$$C(y_{\tau_{i-1}+1:\tau_i}) = \sum_{j=\tau_{i-1}}^{\tau_i} \frac{\left(y_j - n_i^{-1} S_i\right)^2}{\sigma^2}$$

Variance changes: $\theta = \{\sigma^2\}$

$$C(y_{\tau_{i-1}+1:\tau_i}) = n_i \left(\log \left(\sum_{j=\tau_{i-1}}^{\tau_i} \left(y_j - \mu\right)^2 \right) - \log n_i \right)$$

Both mean and variance change: $\theta = \{\mu, \sigma^2\}$

$$C(y_{\tau_{i-1}+1:\tau_i}) = n_i \left(\log \left(\sum_{j=\tau_{i-1}}^{\tau_i} \left(y_j - n_i^{-1} S_i\right)^2 \right) - \log n_i \right)$$

Gamma distribution: $\Theta = \{a, b\}$

$$-2\log L = 2 \times \sum_{i=1}^{m} \sum_{j=\tau_{i-1}}^{\tau_i} \log \Gamma(a_i) + a_i \log b_i + (1 - a_i) \log y_j + \frac{y_j}{b_i}$$

Scale changes: $\theta = \{b\}$

$$C(y_{\tau_{i-1}+1:\tau_i}) = 2an_i(\log S_i - \log(an_i))$$

Exponential Distribution: $\Theta = \{\lambda\}$

$$-2\log L = 2 \times \sum_{i=1}^{m} \sum_{j=\tau_{i-1}}^{\tau_i} \log \lambda_i + \frac{y_j}{\lambda_i}$$

Mean changes: $\theta = \{\lambda\}$

$$C(y_{\tau_{i-1}+1:\tau_i}) = 2n_i(\log S_i - \log n_i)$$

Poisson distribution: $\Theta = \{\lambda\}$

$$-2\log L = 2 \times \sum_{i=1}^{m} \sum_{j=\tau_{i-1}}^{\tau_i} \lambda_i - \operatorname{floor} y_j + 0.5 \log \lambda_i + \log \Gamma\left(\operatorname{floor} y_j + 0.5 + 1\right)$$

Mean changes: $\theta = \{\lambda\}$

$$C(y_{\tau_{i-1}+1:\tau_i}) = 2S_i(\log n_i - \log S_i)$$

when calculating S_i for the Poisson distribution, the sum is calculated for $\operatorname{floor} y_i + 0.5$ rather than y_i.

4 References

Chen J and Gupta A K (2010) *Parameteric Statisical Change Point Analysis With Applications to Genetics Medicine and Finance* **Second Edition** Birkhäuser

Killick R, Fearnhead P and Eckely I A (2012) Optimal detection of changepoints with a linear computational cost *Journal of the American Statistical Association* **107:500** 1590–1598

5 Parameters

1: CTYPE – INTEGER *Input*

On entry: a flag indicating the assumed distribution of the data and the type of change point being looked for.

CTYPE $= 1$
: Data from a Normal distribution, looking for changes in the mean, μ.

CTYPE $= 2$
: Data from a Normal distribution, looking for changes in the standard deviation σ.

CTYPE $= 3$
: Data from a Normal distribution, looking for changes in the mean, μ and standard deviation σ.

CTYPE $= 4$
: Data from a Gamma distribution, looking for changes in the scale parameter b.

CTYPE $= 5$
: Data from an exponential distribution, looking for changes in λ.

CTYPE $= 6$
: Data from a Poisson distribution, looking for changes in λ.

Constraint: CTYPE $= 1$, 2, 3, 4, 5 or 6.

2: N – INTEGER *Input*

On entry: n, the length of the time series.

Constraint: N ≥ 2.

3: Y(N) – REAL (KIND=nag_wp) array *Input*

On entry: y, the time series.

if CTYPE $= 6$, that is the data is assumed to come from a Poisson distribution, floor $y + 0.5$ is used in all calculations.

Constraints:

if CTYPE $= 4$, 5 or 6, Y$(i) \geq 0$, for $i = 1, 2, \ldots,$ N;
if CTYPE $= 6$, each value of Y must be representable as an integer;
if CTYPE $\neq 6$, each value of Y must be small enough such that Y$(i)^2$, for $i = 1, 2, \ldots,$ N, can be calculated without incurring overflow.

4: BETA – REAL (KIND=nag_wp) *Input*

On entry: β, the penalty term.

There are a number of standard ways of setting β, including:

SIC or BIC
: $\beta = p \times \log(n)$

AIC
: $\beta = 2p$

Hannan-Quinn
$\beta = 2p \times \log(\log(n))$

where p is the number of parameters being treated as estimated in each segment. This is usually set to 2 when CTYPE $= 3$ and 1 otherwise.

If no penalty is required then set $\beta = 0$. Generally, the smaller the value of β the larger the number of suggested change points.

5: MINSS – INTEGER *Input*

On entry: the minimum distance between two change points, that is $\tau_i - \tau_{i-1} \geq$ MINSS.

Constraint: MINSS ≥ 2.

6: IPARAM – INTEGER *Input*

On entry: if IPARAM $= 1$ distributional parameters have been supplied in PARAM.

Constraints:

if CTYPE $= 4$, IPARAM $= 1$;
otherwise IPARAM $= 0$ or 1.

7: PARAM(1) – REAL (KIND=nag_wp) array *Input*

On entry: ϕ, values for the parameters that will be treated as fixed. If IPARAM $= 0$ then PARAM is not referenced.

If CTYPE $= 1$

if IPARAM $= 0$, σ, the standard deviation of the Normal distribution, is estimated from the full input data. Otherwise $\sigma =$ PARAM(1).

If CTYPE $= 2$

If IPARAM $= 0$, μ, the mean of the Normal distribution, is estimated from the full input data. Otherwise $\mu =$ PARAM(1).

If CTYPE $= 4$, PARAM(1) must hold the shape, a, for the Gamma distribution, otherwise PARAM is not referenced.

Constraint: if CTYPE $= 1$ or 4, PARAM(1) > 0.0.

8: NTAU – INTEGER *Output*

On exit: m, the number of change points detected.

9: TAU(N) – INTEGER array *Output*

On exit: the first m elements of TAU hold the location of the change points. The ith segment is defined by $y_{(\tau_{i-1}+1)}$ to y_{τ_i}, where $\tau_0 = 0$ and $\tau_i =$ TAU$(i), 1 \leq i \leq m$.

The remainder of TAU is used as workspace.

10: SPARAM($2 \times$ N $+ 2$) – REAL (KIND=nag_wp) array *Output*

On exit: the estimated values of the distribution parameters in each segment

CTYPE $= 1$, 2 or 3
SPARAM$(2i-1) = \mu_i$ and SPARAM$(2i) = \sigma_i$ for $i = 1, 2, \ldots, m$, where μ_i and σ_i is the mean and standard deviation, respectively, of the values of y in the ith segment.

It should be noted that $\sigma_i = \sigma_j$ when CTYPE $= 1$ and $\mu_i = \mu_j$ when CTYPE $= 2$, for all i and j.

CTYPE = 4
: SPARAM$(2i-1) = a_i$ and SPARAM$(2i) = b_i$ for $i = 1, 2, \ldots, m$, where a_i and b_i are the shape and scale parameters, respectively, for the values of y in the ith segment. It should be noted that $a_i =$ PARAM(1) for all i.

CTYPE = 5 or 6
: SPARAM$(i) = \lambda_i$ for $i = 1, 2, \ldots, m$, where λ_i is the mean of the values of y in the ith segment.

The remainder of SPARAM is used as workspace.

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.

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.**

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).

Errors or warnings detected by the routine:

IFAIL = 11

On entry, CTYPE = ⟨*value*⟩.
Constraint: CTYPE = 1, 2, 3, 4, 5 or 6.

IFAIL = 21

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

IFAIL = 31

On entry, CTYPE = ⟨*value*⟩ and Y(⟨*value*⟩) = ⟨*value*⟩.
Constraint: if CTYPE = 4, 5 or 6 then Y$(i) \geq 0.0$, for $i = 1, 2, \ldots,$ N.

IFAIL = 32

On entry, Y(⟨*value*⟩) = ⟨*value*⟩, is too large.

IFAIL = 51

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

IFAIL = 61

On entry, IPARAM = ⟨*value*⟩.
Constraint: if CTYPE ≠ 4 then IPARAM = 0 or 1.

IFAIL = 62

On entry, IPARAM = ⟨*value*⟩.
Constraint: if CTYPE = 4 then IPARAM = 1.

IFAIL $= 71$

On entry, CTYPE $= \langle value \rangle$ and PARAM(1) $= \langle value \rangle$.
Constraint: if CTYPE $= 1$ or 4 and IPARAM $= 1$, then PARAM(1) > 0.0.

IFAIL $= 200$

To avoid overflow some truncation occurred when calculating the cost function, C. All output is returned as normal.

IFAIL $= 201$

To avoid overflow some truncation occurred when calculating the parameter estimates returned in SPARAM. All output is returned as normal.

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

For efficiency reasons, when calculating the cost functions, C and the parameter estimates returned in SPARAM, this routine makes use of the mathematical identities:

$$\sum_{j=\mathrm{u}}^{v} {y_j}^2 = \sum_{j=1}^{v} {y_j}^2 - \sum_{j=1}^{u-1} {y_j}^2$$

and

$$\sum_{j=1}^{n} \left(y_j - \bar{y}\right)^2 = \left(\sum_{j=1}^{n} y_j^2\right) - n\bar{y}^2$$

where $\bar{y} = n^{-1}\sum_{j=1}^{n} y_j$.

The input data, y, is scaled in order to try and mitigate some of the known instabilities associated with using these formulations. The results returned by G13NAF should be sufficient for the majority of datasets. If a more stable method of calculating C is deemed necessary, G13NBF can be used and the method chosen implemented in the user-supplied cost function.

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

This example identifies changes in the mean, under the assumption that the data is normally distributed, for a simulated dataset with 100 observations. A BIC penalty is used, that is $\beta = \log n \approx 4.6$, the minimum segment size is set to 2 and the variance is fixed at 1 across the whole input series.

10.1 Program Text

```
    Program g13nafe
!     G13NAF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: g13naf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: beta
      Integer                          :: ctype, i, ifail, iparam, minss, n,   &
                                          ntau
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: param(1)
      Real (Kind=nag_wp), Allocatable  :: sparam(:), y(:)
      Integer, Allocatable             :: tau(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: repeat
!     .. Executable Statements ..
      Continue
      Write (nout,*) 'G13NAF Example Program Results'
      Write (nout,*)

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

!     Read in the problem size
      Read (nin,*) n

!     Allocate memory to hold the input series
      Allocate (y(n))

!     Read in the input series
      Read (nin,*) y(1:n)

!     Read in the type of change point, penalty and minimum segment size
      Read (nin,*) ctype, iparam, beta, minss

!     Read in the distribution parameter (if required)
      If (iparam==1) Then
        Read (nin,*) param(1)
      End If

!     Allocate output arrays
      Allocate (tau(n),sparam(2*n+2))

!     Call routine to detect change points
      ifail = -1
      Call g13naf(ctype,n,y,beta,minss,iparam,param,ntau,tau,sparam,ifail)

      If (ifail==0 .Or. ifail==200 .Or. ifail==201) Then
!       Display the results
        If (ctype==5 .Or. ctype==6) Then
!         Exponential or Poisson distribution
          Write (nout,99999) ' -- Change Points --      Distribution'
          Write (nout,99999) '  Number     Position      Parameter'
          Write (nout,99999) repeat('=',38)
          Do i = 1, ntau
```

```
            Write (nout,99998) i, tau(i), sparam(i)
          End Do
        Else
!         Normal or Gamma distribution
          Write (nout,99999) &
            ' -- Change Points --         --- Distribution ---'
          Write (nout,99999) ' Number     Position              Parameters'
          Write (nout,99999) repeat('=',50)
          Do i = 1, ntau
            Write (nout,99997) i, tau(i), sparam(2*i-1), sparam(2*i)
          End Do
        End If
        If (ifail==200 .Or. ifail==201) Then
          Write (nout,99999) &
            'Some truncation occurred internally to avoid overflow'
        End If
      End If

99999 Format (1X,A)
99998 Format (1X,I4,7X,I6,4X,F12.2)
99997 Format (1X,I4,7X,I6,2(4X,F12.2))
    End Program g13nafe
```

10.2 Program Data

```
G13NAF Example Program Data
100            :: N
 0.00  0.78 -0.02  0.17  0.04 -1.23  0.24  1.70  0.77  0.06
 0.67  0.94  1.99  2.64  2.26  3.72  3.14  2.28  3.78  0.83
 2.80  1.66  1.93  2.71  2.97  3.04  2.29  3.71  1.69  2.76
 1.96  3.17  1.04  1.50  1.12  1.11  1.00  1.84  1.78  2.39
 1.85  0.62  2.16  0.78  1.70  0.63  1.79  1.21  2.20 -1.34
 0.04 -0.14  2.78  1.83  0.98  0.19  0.57 -1.41  2.05  1.17
 0.44  2.32  0.67  0.73  1.17 -0.34  2.95  1.08  2.16  2.27
-0.14 -0.24  0.27  1.71 -0.04 -1.03 -0.12 -0.67  1.15 -1.10
-1.37  0.59  0.44  0.63 -0.06 -0.62  0.39 -2.63 -1.63 -0.42
-0.73  0.85  0.26  0.48 -0.26 -1.77 -1.53 -1.39  1.68  0.43 :: End of Y
1   1  4.6  2  :: CTYPE,IPARAM,BETA,MINSS
1.0            :: PARAM(1)
```

10.3 Program Results

```
 G13NAF Example Program Results

  -- Change Points --         --- Distribution ---
  Number     Position               Parameters
 ================================================
    1           12             0.34            1.00
    2           32             2.57            1.00
    3           49             1.45            1.00
    4           52            -0.48            1.00
    5           70             1.20            1.00
    6          100            -0.23            1.00
```

This example plot shows the original data series, the estimated change points and the estimated mean in each of the identified segments.

Example Program
Simulated time series and the corresponding changes in mean