Operator CALC_FONCTION

1 Goal

To carry out mathematical operations on structures of data of type function.

The following operations are currently available :

- the derivation of a function,
- the integration of a function,
- the reverse of a function,
- the absolute value of a function,
- the research of the envelope of several functions,
- the calculation of the fractile of tablecloths or functions,
- calculation DE the average tablecloths or functions,
- real or complex linear combination several functions,
- the composition of two functions,
- the product of functions,
- concatenation (put end to end with management of the overlappings) several functions,
- the extraction of a real function starting from a complex function,
- the calculation of the power n ième of a function,
- polynomial regression of a function,
- the calculation of direct or opposite FFT of a function,
- correction of a accélérogramme measured for calculation of a seismic answer,
- the calculation of the function of coherence starting from a whole of signals, contained in two tablecloths
- smoothing wraps one or more rough spectra of oscillator,
- the calculation of the spectrum of oscillator of a accélérogramme (function of the frequency and damping) or starting from a spectral concentration of power (DSP), in the form of a tablecloth,

- the calculation of one DSP equivalent to the data of a spectrum of oscillator using the formula of Vanmarcke.
- the interpolation of a accélérogramme without addition of frequential contents by method “zero padding”.

Product a structure of data `function`, `fonction_c` or `tablecloth`, according to the keyword factor used.
At exit of the order, the function is reordered by increasing X-coordinates.

Contents

2 Syntax

```
Fr      = CALC_FONCTION

   (  ♦  /   DRIFT     =   _F (   ♦  FUNCTION  =  F,
   [function]
                                  ◊  METHOD =   'DIFF_CENTREE',  [DEFECT]
                             ),

         /   JUST    =   _F (   ♦  FUNCTION  =  F,              [function]
                                ◊  METHOD =  /   'TRAPEZOID',  [DEFECT]
                                             /   'SIMPSON',
                                 ◊  COEFF      =  /   0. ,
   [DEFECT]
                                             /   R,             [R]
                             ),

         /   OPPOSITE  =   _F (   ♦  FUNCTION  =  F,
   [function]
                             ),

         /   ABS       =   _F (   ♦  FUNCTION  =  F,
   [function]
                             ),

         /   ENVELOPE =    _F (   ♦  FUNCTION  =  F,
   [L_function]
                                ◊  CRITERION     =  / 'SUP',          [DEFECT]
                                                 / 'INF',
                             ),

         /   FRACTILE  =   _F (   ♦  FUNCTION  =  F,
   [L_function]
                                ◊  FRACT     =   / 1. ,            [DEFECT]
                                                 / fract           [R]
                             ),
         /   AVERAGE   =   _F (   ♦  FUNCTION  =  F,
   [L_function]
                             ),

         /   COHERENCE =   _F(♦  NAPPE_1      =  N,             [TABLECLOTH]
                              ♦  NAPPE_ 2      =  N,             [TABLECLOTH]
                              ◊  NB_FREQ_LISS =  / 12,          [DEFECT]
                                                 / nbl ,        [ I ]
                              ◊  FREQ_COUP     =   FC           [ R]
                              ◊  OPTION        =
                                           / 'ALL',             [DEFECT]
                                          / ' DUREE_PHASE_FORTE ',
                            IF OPTION    =  ' DUREE_PHASE_FORTE '
                              ◊  borne_inf =  / 0.05,           [DEFECT]
                                              / binf,           [R]
                              ◊  borne_sup =  / 0.95,           [DEFECT]
                                               / BSUP,          [ R ]
                             ),


         /   COMB      =   _F (   ♦  FUNCTION  =  F,
   [function]
```

```
                              ♦  COEFF     =  R,              [R]
                          ),

        /  COMB_C    =   _F (   ♦  FUNCTION  = f_c,
    [fonction_c]
                              ♦  /  COEF_R = R ,              [R]
                                 /  COEF_C = C ,              [C]
                          ),

        /  MULT   =   _F (   ♦  FUNCTION  =  F,
    [function]
                          ),

        /  REGR_POLYNOMIALE =   _F (   ♦  FUNCTION  =  F,
  [function]
                                    ♦  DEGREE =  N,           [I]
                          ),
                   # if COMB or COMB_C or REGR_POLYNOMIALE
                   ◊  LIST_PARA =  will lpara,
    [listr8]

        /  COMPOSE   =   _F (   ♦  FONC_RESU =  f_resu ,
    [function]
                              ♦  FONC_PARA =  will f_para  ,
    [function]
                          ),

        /  ADZE      =   _F (   ♦  FUNCTION  =  l_f,
    [l_fonction]
                              ◊  OVERLOAD  =  /  'RIGHT',     [DEFECT]
                                              /  'LEFT',
                          ),

        /  EXTRACTION=   _F (   ♦  FUNCTION  =  f_c,
    [fonction_c]
                              ♦  PART      =  /  'REAL',
                                              /  'IMAG',
                                              /  'MODULE',
                                              /  'PHASE',
                          ),

        /  POWER  =   _F (   ♦  FUNCTION  =  F,               [function]
                              ◊  EXHIBITOR =  / N,            [I]
                                              / 1,            [DEFECT]
                          ),

        /  FFT       =   _F (   ♦  FUNCTION  =  F,
    [function]
                              ◊  METHOD =  /  'PROL_ZERO', [DEFECT]
                                              /  'TRUNCATION',
                                              /  'COMPLETE',
                              ◊   SYME     =  /  'YES',       [DEFECT]
                                              /  'NOT',
                          ),FMIN

        /  CORR_ACCE = _F (  ♦ FUNCTION  =  F,                [function]
                              ♦  METHOD =   /  'FILTERING',
                                               /  'POLYNOMIAL',
        if METHOD == 'POLYNOMIAL'
```

```
                                ◊  CORR_DEPL =  /   'NOT',        [DEFECT]
                                                /   'YES',
                 if METHOD == 'FILTERING'
                                ◊  FREQ_FILTRE  =  / Fr
                                                   /0.05,         [DEFECT]
                             ),

                 /  LISS_ENVELOP = _F (
                                ♦   /  TABLECLOTH       =  N,
   [tablecloth]
                                  /  TABLE      =  T,             [table]
                                  /  FUNCTION   =  F,             [function]
                                ♦  OPTION     =  /   'DESIGN',
                                                /   'CHECKING',
                                 ◊  FREQ_MIN =  /   fmin,         [R]
                                ◊  FREQ_MAX  =  /   fmax,         [R]
                                ◊  ELARG     =  /   elar,         [l_R]
                                ◊  LIST_FREQ =  /   listfreq,     [l_R]
                                ◊  LIST_AMOR    =  /  listamor,     [l_R]
                                ◊  NB_FREQ_LISS =  /  Nb,         [R]
                                               =  /  10,          [DEFECT]
                                ◊  ZPA       =  /   zpa,          [R]
                             ),


               /   DSP = _F (
                             ♦  FUNCTION     =  sro,              [function]
                             ♦  AMOR_REDUIT  =  lam,              [l_R]
                            ◊ /  FREQ_PAS  =  freq_pas,             [R]
                              /  LIST_FREQ =  list_freq,          [listr8]
                             ♦  FREQ_COUP    =  frc               [R]
                             ♦  DURATION        =  tsm              [R]
                             ♦  NORMALIZES         =  R,               [R]
                             ◊  FRACT        =  /  0.5,           [DEFECT]
                                               /  fract           [R]


               /  SPEC_OSCI =   _F (  ♦  FUNCTION     =  F,             [function]
                                ◊  METHOD     =  /'NIGAM',     [DEFECT]
                                                   /'HARMO'
                                                   /'RICE'
                                ◊  AMOR_REDUIT  =  lam,           [L_R]
                                ◊  /  FREQ      =  lfre,          [L_R]
                                   /  LIST_FREQ =  lfreq,         [listr8]
                                ◊  NATURE       =  / 'ACCE',      [DEFECT]
                                                   / 'QUICKLY',
                                                   / 'DEPL',
                                ♦  NORMALIZES         =  R,             [R]

     if METHOD == 'RICE'
                                ◊  NATURE_FONC  =  'DSP',         [DEFECT]
                                ♦  DURATION        =  tsm            [R]
     if METHOD == 'NIGAM' or 'HARMO'
                                ◊  NATURE_FONC  =  'ACCE',        [DEFECT]

                             ),

                 ◊  NOM_PARA        =  para,                      [KN]
                 ◊  NOM_RESU        =  resu,                      [K N ]
```

```
    ◊   PROL_DROITE      =  /   'CONSTANT',
                            /   'LINEAR',
                            /   'EXCLUDED',
    ◊   PROL_GAUCHE      =  /   'CONSTANT',
                            /   'LINEAR',
                            /   'EXCLUDED'
    ◊   Interpol         =  |   'FLAX',                    [L_KN]
                            |   'LOG',
                            |   'NOT',
    ◊   INTERPOL_FONC    =  |   'FLAX',                    [L_KN]
                            |   'LOG',
                            |   'NOT',
    ◊   NOM_PARA_FONC    =  parf,                      [KN]
    ◊   PROL_DROITE_FONC =  /   'CONSTANT',
                            /   'LINEAR',
                            /   'EXCLUDED',
    ◊   PROL_GAUCHE_FONC =  /   'CONSTANT',
                            /   'LINEAR',
                            /   'EXCLUDED',
    ◊   INFORMATION   =  /   1,
[DEFECT]
                     /   2,
)
```

If keyword factor COMB_C then Fr = [FONCTION_C],
if keyword factor SPEC_OSCI then Fr = [TABLECLOTH],
if keyword factor COHERENCE then Fr = [FUNCTION],
if keyword factor ENVELOPE, FRACTILE, POWER then Fr is of the same type as the functions as starter,
for all the other keyword factors, Fr = [FUNCTION].

3 Operands

3.1 Keyword DRIFT

/ `DRIFT =`

The function is derived $f(t)$.

♦ `FUNCTION = F`

Name of the function which one wishes to derive.
Does not apply to the concepts of the type `tablecloth`.

◊ `METHOD =`

Name of `METHOD` that one wishes to use : the only method available is currently `DIFF_CENTREE` (by default).

Remarks :

See keyword `JUST`.

3.2 Keyword JUST

/ `INTEGRATE =`

The function is integrated $f(t)$.

◊ `COEFF =` r

Constant of integration, by default `0`.

♦ `FUNCTION = F`

Name of the function which one wishes to integrate.
Does not apply to the concepts of the type `tablecloth`.

◊ `METHOD =`

Name of `METHOD` that one wishes to use.

Two methods are available: method of `'TRAPEZOID'` (by default) and method of `'SIMPSON'`.

The integral is exact for the linear functions per pieces as provided as starter. The mistake made compared to the integral of the function which one discretized is in $o(1/n^2)$ for functions of classes C^2.

The method of Simpson is exact for the polynomials of degree lower or equal to 3. The error is in $o(1/n^4)$ for functions of classes C^4.

For the unspecified functions, very kicked up a rumpus, like the accélérogrammes, it is advised to use the method of the trapezoids.
On the other hand, when the function $f(t)$ (before discretization) is sufficiently regular, the method of Simpson is much more precise.

Remarks :

1) *For `JUST` as for `DRIFT`, it `NOM_PARA` produced function is unchanged. On the other hand, it `NOM_RESU` can be modified in the following cases: for derivation, `DEPL` becomes `QUICKLY`, `QUICKLY` becomes `ACCE` ; for integration, `ACCE` becomes `QUICKLY`, `QUICKLY` becomes `DEPL`. The user always has the possibility of modifying it by the keyword of the same name in `CALC_FONCTION`.*

2) *Concerning the prolongations, the produced function has, by default of the prolongations `EXCLUDED` on the left and on the right some are those of the starting function. Not to thus expect that a linear prolongation becomes constant in the derived function… There still, the user is Master of his prolongations for the function produced by the keywords `PROL_DROITE` and `PROL_GAUCHE`.*

3.3 Keyword OPPOSITE

/ `OPPOSITE =`

The function is reversed $f(t)$.

♦ `FUNCTION = F`

Name of the function which one wishes to reverse, it is necessary that this one is bijective (strictly increasing or strictly decreasing).

Does not apply to the concepts of the type `tablecloth`.

Notice :

1) *The labels of the parameters are not reversed! The care is left to the user affect the correct values by the keywords `NOM_PARA` and `NOM_RESU`. By default, it `NOM_PARA` is unchanged and `NOM_RESU` is affected with `'TOUTRESU'` .*
2) *The modes of interpolations are inverted: e.g. (`'FLAX'` , `'LOG'`) becomes (`'LOG'` , `'FLAX'`).*
3) *Prolongations `EXCLUDED` and `LINEAR` are unchanged. On the other hand, a prolongation `CONSTANT` is changed into `EXCLUDED`.*

3.4 Keyword ABS

/ `ABS =`

Provides the absolute value of a function or a tablecloth.

♦ `FUNCTION = F`

Name of the function which one wishes the absolute value.

Notice :

1) *The parameters (prolongations, interpolations, `NOM_PARA` and `NOM_RESU`) produced function are the same ones as those of the starting function.*
2) *Except for the prolongation `LINEAR`: systematically changed into `EXCLUDED` by precaution. Indeed, the linear prolongation on the right of a decreasing function leads for sufficiently large X-coordinates to negative values: responsibility is thus left to the user affect itself `PROL_DROITE=' LINEAIRE'` (and respectively on the left).*

3.5 Keyword ENVELOPE

/ `ENVELOPE =`

Calculation of the envelope of several functions.

This operation is available on operands of nature `function` or `tablecloth`.

3.5.1 Operand FUNCTION

♦ `FUNCTION = F`

List of the functions or tablecloths which one seeks the envelope.

3.5.2 Operand CRITERION

◊ CRITERION =

/ 'SUP'

The higher envelope is sought.

/ 'INF'

The lower envelope is sought.

Remarks for the research of the envelope:

- *the functions all must be of comparable nature (* `function` *or* `tablecloth` *),*
- *Case of the simple functions: for the prolongations, interpolations,* `NOM_PARA` *and* `NOM_RESU`*, they are the parameters of the first of the functions in the list which are retained. The support of X-coordinates of the function envelope will be the meeting of the lists of X-coordinates of all the functions.*
- *Case of the tablecloths: the parameters (prolongations, interpolations,* `NOM_PARA`*,* `NOM_RESU`*,* `NOM_PARA_FONC`*) must imperatively be identical between the provided tablecloths. The supports of X-coordinates (values of the parameters and X-coordinates of the functions of the tablecloths) are homogenized to be able to calculate the envelope. The produced tablecloth will have this discretization for X-coordinates.*

3.6 Keyword FRACTILE

/ FRACTILE =

Calculation of the fractile several functions.

This operation is available on operands of nature `function` or `tablecloth`.

3.6.1 Operand FUNCTION

♦ FUNCTION = F

List of the functions or tablecloths which one seeks to calculate the fractile.

3.6.2 Operand FRACT

♦ FRACT = fract

Value of the quantile to be calculated. By default `fract = 1`, the fractile is then the higher envelope.

3.7 Keyword AVERAGE

/ AVERAGE =

Calculation of the average of several functions.

This operation is available on operands of nature `function` or `tablecloth`.

3.8 Keyword COHERENCE

/ COHERENCE =

Calculation of function of coherence of temporal signals.
The function of coherence depends on the distance between two points of measurement and on the frequency. It expresses the correlation between the two signals, measured at a distance d , according to the frequency. The function of coherence complex is evaluated in frequential field, via the spectral concentrations (DSP). The operator returns the real part

complex coherence, which corresponds to " *unlagged coherency* " or it " *plane wave coherency* " in the case of a plane wave.
The temporal signals must have the same discretization (even not time dt and even lasted T). The user must provide two tablecloths which contain the signals respectively withX stations 1 and 2, outdistancedS of d . The signals in each tablecloth are parameterized by NUME_ORDRE (1,2,3.) and then paired to calculate coherences and to take the average of it.

3.8.1 OperandS NAPPE_1 and NAPPE_2

♦ NAPPE_1 = nappe1

Tablecloth containing NR functions measured at station 1. The functions of the tablecloth must be numbered of 1 to NR by NOM_PARA = 'NUMÉRIQUE_ORDRE'.

♦ NAPPE_2 = nappe2

Tablecloth containing NR functions measured at station 2. The functions of the tablecloth must be numbered of 1 to NR by NOM_PARA = 'NUMÉRIQUE_ORDRE'.

3.8.2 Operand FREQ_COUP

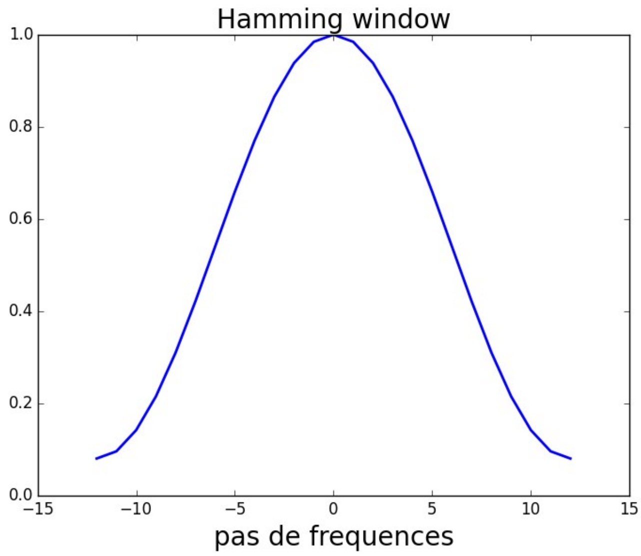
◊ FREQ_COUP = FC

Cut-off frequency for the tracing of the functions of coherence. cut-off frequency must to be lower than $\frac{1}{2 \times dt}$, where dt is the step of time of the temporal signals. For the evaluation of the functions of coherence, the cut-off frequency is always determined by the step of time of the temporal signals. If FREQ_COUP is informed, then the function of coherence is crossedE with fc before restitution.

3.8.3 Operand NB_FREQ_LISS

```
◊ NB_FREQ_LISS =/ 12      [defect]
                / nfl
```

Many steps of frequency used for the smoothing of the DSP by window the Hamming one. The window the Hamming one obtained with NB_FREQ_LISS = 12 is shown in the figure below.



3.8.4 OperandS OPTION, BORNE_INF and BORNE_SUP

◊ / OPTION =

Allows to determine whether the whole signal or only the strong phase is used for the calculation of the function of coherence :

/ 'ALL' [DEFECT]

One uses the temporal signals such as they are.

/ 'DUREE_PHASE_FORTE'

One only evaluates the function of coherence for the duration of strong phase. The moments of beginning and end of strong phase are calculated for the first signal only. One uses then the same time interval for the other signals. In the case of the earthquake, that supposes that all the recordings come from the same event. To evaluate the duration of strong phase, one uses the notion of the intensity of Arias. The limits should then be informed lower and higher (*cf.* [U4.32.05]).

◊ BORNE_INF = /0.05
/binf,
◊ BORNE_SUP = /0.95
/bsup,

Terminals limiting the share of intensity Arias defining them Instants Initial and final of the strong phase (enters $(b_{\inf})\%$ and $(b_{\sup})\%$ of intensity of Arias) earthquake - even significance that for INFO_FONCTION. By default, one takes 5% and 95%.

3.9 Keyword COMB and operand LIST_PARA

/ COMB =

Real linear combination several concepts of nature function or tablecloth.

♦ FUNCTION = F

Name of the function to be combined.

♦ COEFF = R

Value of the coefficient.

◊ LIST_PARA = will lpara

List of the values of the parameters for which the combination of the functions will be discretized. If this keyword is not indicated, a list by default is built by taking the union of the lists of the values of the parameters of each function.

Caution:

It is not a keyword of the keyword factor COMB .

Remarks for the combination:

See the remarks for the keyword ENVELOPE

3.10 Keyword COMB_C and operand LIST_PARA

/ COMB_C =

Linear combination complexes several concepts of nature fonction_c.

♦ FUNCTION = f_c

Name of the function to be combined. It can be with complex or real values.

/ COEF_R = R,
/ COEF_C = C,

Value of the multiplying coefficient, is in real form R, that is to say in complex form C.

◊ LIST_PARA = will lpara

List of the values of the parameters for which the combination of functions will be discretized. If this keyword is not indicated, a list by default is built by taking the union of the lists of the values of the parameters of each function.

Remarks for the combination:

See the remarks for the keyword ENVELOPE

3.11 Keyword MULT and operand LIST_PARA

/ MULT

Product of concepts of comparable nature function, fonction_c or tablecloth.

♦ FUNCTION = F

Name of the function to be multiplied with the others.

LIST_PARA allows to discretize the function result as for COMB.

3.12 Keyword COMPOSE

Keyword factor allowing to calculate the made up one of two functions $F(G(t))$.
Does not apply to the concepts of the type tablecloth.

/ COMPOSE =

♦ FONC_RESU = f_resu

Function f_resu (X)

♦ FONC_PARA = will f_para

Function will f_para (T)

It is checked that it NOM_PARA of f_resu corresponds to NOM_RESU of will f_para.

3.13 Keyword ADZE

/ ADZE =

Keyword factor allowing to create a real function by concaténant two tabulées real functions. Does not apply to the concepts of the type tablecloth.

3.13.1 Operand FUNCTION

♦ FUNCTION = l_f

Functions with concaténer. Two functions exactly are expected.

3.13.2 Operand OVERLOAD

◊ OVERLOAD = / 'RIGHT',
/ 'LEFT',

The points of discretization of the function created are those of the whole of the two functions, modulo the effects of overload.
If the fields of definition of the functions overlap, one of the functions imposes its points on the zone of covering and for the prolongations :

OVERLOAD =/ 'RIGHT' : it is the function which has largest xmax who is chosen,
OVERLOAD =/ 'LEFT' : it is the function which has smallest xmin who is selected.

3.13.3 Checks

It is checked that all the functions have the same one NOM_PARA, as well as the same interpolations.

3.14 Keyword EXTRACTION

/ EXTRACTION =

Keyword factor making it possible to build starting from a function complexes (standard fonct_c), a real function representative either the real part, or the imaginary part, or the module, or the phase of the complex function.

3.14.1 Operand FUNCTION

♦ FUNCTION = f_c

Complex function.

3.14.2 Operand PART

♦ PART =

/ 'REAL' : extraction of the real part of f_c,
/ 'IMAG' : extraction of the imaginary part of f_c,
/ 'MODULE' : extraction of the module of f_c,
/ 'PHASE' : extraction of the phase (in degree) of f_c.

3.15 Keyword POWER

This keyword makes it possible to build power N^ième of a function or a set of functions provided in the form of a tablecloth.

♦ FUNCTION = F

Name of the function F concerned (standard function or tablecloth).

♦ EXHIBITOR = N

The function result calculated will be $x \rightarrow f(x)^n$. By default, $n=1$.

3.16 Keyword REGR_POLYNOMIALE

This keyword calculates the polynomial regression of a function by the method of least squares (by using the function polyfit of numpy).

♦ FUNCTION = F

Name of the function F concerned (standard function).

♦ DEGREE = N

Degree of the required polynomial.

One can use the keyword LIST_PARA for tabuler the calculated polynomial. If not, it is tabulé on the list of the X-coordinates of the function F.

3.17 Keyword FFT

/ `FFT =`

One calculates the transform of Fourier direct or opposite of a function (of which by algorithm FFT).

♦ `FUNCTION = F`

Name of the function on which the operation is carried out.
If it `NOM_PARA` function is `INST`, then `FFT` direct is calculated.
If it `NOM_PARA` function is `FREQ`, then `FFT` opposite is calculated.
Does not apply to the concepts of the type `tablecloth`.

♦ `METHOD =`

Algorithm FFT is faster for the samples of which the length is a power of 2.
Method `'PROL_ZERO'` (by default) proposes to prolong the entry signal with zeros until having a full number of sample which is the first power of 2 whose value is higher than the initial number of samples.
Method `'TRUNCATION'` will consider only the first samples of which the full number is more the great power of two whose value is lower than the initial number of sample.
For example, on a signal of 601 values, the method `'PROL_ZERO'` will supplement the signal to have 1024 samples, whereas the method `'TRUNCATION'` will consider only the first 512 moments.
If the entry signal has a number of sample which is a power of two, the two methods are obviously equivalent: one takes into account the signal without modifying it.
Method `'COMPLETE'` allows to take into account the totality of the entry signal, some is the number of samples.
Nota bene: in the case of a sample length N, of which the step of time would be dt, the sampling rate of the FFT is $1/(N.dt)$. On the other hand, the last frequency for which the discrete transform is calculated is not $1/dt$, but $(N-1)/(N.dt)$.

♦ `SYME =`

Keyword which applies only for the opposite transform of Fourier.
If the complete spectrum would be provided, then the opposite transform is calculated directly while using `SYME = 'YES'`. Methods `'TRONCATURE'` and `'PROL_ZERO'` are then not active.
If the spectrum (complex) provided as starter of the opposite FFT does not contain the folded up part (partners at the negative frequencies of the spectrum), one can nevertheless consider a signal temporal having the same spectral contents on the part associated with the positive frequencies. If one notes X_k k ième sample of the transform of Fourier of a sample length N, then one has $X_k = X^*_{(N-k)}$, where $()^*$ corresponds to the combined complex. This information can be exploited to rebuild a temporal signal by knowing only half of the spectrum. This operation is carried out when one chooses `SYME = 'NOT'`. The temporal signal is then rebuilt to obtain a temporal sample even length. In theory, to rebuild a temporal signal length $2 \times M$, the spectrum must check certain conditions:

1. The spectrum must be length $M+1$,
2. The first point of the spectrum must be real,
3. The last point of the spectrum must be real.

If these conditions are not checked, then one builds an approximate spectrum odd length checking these conditions. If the initial spectrum is even length, the last point is then rebuilt by carrying out a prolongation by interpolation of the initial spectrum. This reconstruction can introduce a light skew when the spectral contents of the sample are very significant on the last points of the spectrum.
Methods `'TRONCATURE'` and `'PROL_ZERO'` are still available for the opposite FFT. Attention, however, with the use of the method `'TRONCATURE'`. If the number of truncated point is significant, then the results can be very appreciably different.

3.18 Keyword INTERPOL_FFT

/ INTERPOL_FFT =

Interpolation of a accélérogramme by method “zero padding”. This method consists in calculating the transform of Fourier of the initial signal, to supplement the frequential signal obtained by newsS frequencies of zero value then with calculationer the transform of Fourier reverses of this modified frequential signal. This functionality uses the same algorithms as the keyword FFT. The method is necessarily 'COMPLETE' and SYME = 'NOT'.
It makes it possible not to add of frequential contents the interpolated function.

♦ FUNCTION = F

Name of the function on which the operation is carried out.
NOM_PARA function is necessarily INST.

♦ PAS_INST =

Increment of time desired for the interpolated signal. Must be smaller than the increment of time provided function.

◊ PRECISION =
/ prec
/ 0.01 [DEFECT]

The step of time really obtained is often slightly different that indicated via PAS_INST. This operand defines the maximum change desired between these two values, if it is exceeded, a message of alarm is emitted to inform the user.

3.19 Keyword CORR_ACCE

/ CORR_ACCE =

Keyword factor allowing to correct a accélérogramme (seismic signal in acceleration) so that the signal in displacement does not have drift. One turns over at exit the corrected accélérogramme. The signal can then be integrated by the methods standards (keyword JUST of CALC_FONCTION). It is advisable to check the quality of the result by checking the absence of drift of the signal in displacement and comparing the spectra of answer before and after modification.

3.19.1 Operand FUNCTION

♦ FUNCTION = F

Measured real Accélérogramme.
Does not apply to the concepts of the type tablecloth.

3.19.2 Operand METHOD

♦ METHOD =

Name of METHOD that one wishes to use to correct the signals: correction by POLYNOMIALS or by FILTERING in the frequential field.

METHODE=' POLYNOME'
One removes the drift of the signal, calculated by linear smoothing within the meaning of least squares on the totality of the signal. The drift corresponding relative speed is also removed.

◊ CORR_DEPL =

/ 'NOT'

One does not correct the drift of relative displacement, it is the value by default.

/ 'YES'

One removes also the drift of relative displacement. This option is to be used with precaution, because one does not know a priori the value of final displacement after the earthquake.

METHODE=' FILTRAGE'
One removes the drift by filtering (“passe_haut”) signal in the field of the frequencies. This filter is described in R4.05.05 documentation (section 2.1).

◊ FREQ_FILTRE =
/ Fr
/ 0.05 [DEFECT]

It is necessary to choose the smallest frequency which makes it possible to obtain the discounted effect, namely to remove the drift, without (too much) deteriorating the other properties of the signal. By default the frequency of the filter is worth 0.05Hz. This value is generally well adapted for the seismic signals. This value can be decreased if the signal allows it or increased if a drift persists.

3.20 Keyword LISS_ENVELOP

♦ / TABLECLOTH = N

Name of the tablecloth or the tablecloths of formed entry(S) rough spectra associated with each level of damping. The tablecloth thus depends on the parameters frequency and damping.

/ TABLE =

Name of the table or the tables of entry formedE (S) rough spectra associated with each level of damping. In the expected format, the first column contains the values of the frequencies, each following column contains the value of acceleration associated with the frequency for a level with damping given. The values of depreciation are given by LIST_AMOR.

/ FUNCTION = N

Name of the formed function DU spectrum associated gross with a level of damping.

♦ OPTION =
/ ‘DESIGN’
The first stage consists in building an envelope on the provided spectra. In one second stage, one carries out the smoothing of the spectrum obtained according to the number of smoothed points asked NB_FREQ_LISS .
/ ‘ VERIFICATION ‘
The first stage consists in carrying out a smoothing of each provided spectrum according to the number of points asked . Then, one carries out a widening of each spectrum following the provided coefficients dan S ELARG . In a third stage, the envelope of the widened spectra is calculated . Finally the last stage realizes the smoothing of the spectrum obtained according to the number of smoothed points asked.

◊ NB_FREQ_LISS
Many frequencies desired for the smoothed spectrum. In the case of the option ‘DESIGN‘, only one value is provided. For the option ‘ CHECKING ‘, one can provide two values which will be applied at the first stage and to fourth stage.

◊ FREQ_MIN and FREQ_MAX

Beach of definition in frequency of the smoothed spectrum.
Frequencies mentioned under FREQ_MIN and FREQ_MAX must be selected among frequencies of discretization of the rough spectrum.
By default, one considers the complete spectrum.

◊ ELARG

Widening relates to the whole of a spectrum. One must provide values as many as of many tablecloths or tables.

For each frequency F_I rough spectrum, one defines two new values of frequencies such that:

- $F^- = F_i(1-\tau_g)$ with $0<\tau_g<1$;
- $F^+ = F_i(\ 1+\tau_d)$ with $0<\tau_d<1$.

Parameters τ_g and τ_d represent the amplitude of widening in frequency and are equal to the value providedE.

Values of the offset frequencies F^- and F^+ do not correspond to the values F_i list of definition of the rough spectrum. One defines thus F_j and F_k such as:

- F_j : value belonging to the list, immediately below or equalizes with F^- ,
- F_k : value belonging to the list, immediately below or equalizes with F^+ .

For each frequency F_i , two points of coordinates (F_j, γ_i) and (F_k, γ_i) are defined where γ_i represent acceleration at the frequency F_i . Two new spectra resulting from the shift rough spectrum on the axis of the frequencies are thus built. Then the envelope is calculated.

◊ LIST_FREQ

List of frequencies which one wishes to preserve for the smoothed spectrum. The elements of this list must be contained in the values of frequency of the spectra taken as starter.

◊ LIST_AMOR

List of depreciation which one wishes to associate with the various values of accelerations of provided table.

◊ ZPA

Value of the high frequency acceleration which one wishes to impose for the smoothed spectrum. By default, it acts of the value of the spectrum less deadened at the highest frequency.

An example of application is proposed in the case test ZZZZ100E.

3.21 Keyword SPEC_OSCI

/ SPEC_OSCI =

When METHOD = 'NIGAM' (defect) or 'HARMO' (*cf.* §3.21.1), NATURE_FONC = 'ACCE' (*cf.* §3.21.2), one Calcule the spectrum of oscillator of a accélérogramme [R4.05.03] given under the keyword FUNCTION (*cf.* §3.21.1).

The spectrum of oscillator is calculable only on the functions of NOM_RESU = 'ACCE' and of NOM_PARA = 'INST'.

For all i and all j one considers q_j^i the solution of the differential equation:

$$\ddot{q}_j^i + 2\xi_j \omega_i \dot{q}_j^i + \omega_i^2 q_j^i = f(t)$$

with $q_j^i(0) = \dot{q}_j^i(0) = f(0)$ and $\omega_i = 2\pi\varphi_i$

The produced concept fr is a tablecloth (function with two variables) made up by the functions $(fr_i, \ldots, fr_j, \ldots)$ with fr_j function defined in the points ω_i with:

$$fr_j(\omega_i) = \underset{t \in D}{Max} \left| q_i^j(t) \right| \text{ and } D = \{ t / f \text{ définie} \}$$

By default for the calculation of the spectrum of oscillator

- one considers for reduced depreciation the values:

  `0.02 0.05 0.10`

- one considers for the frequencies, the 150 values following in Hz,

  first is with $0.2\,Hz$ and one deduces the following ones by the rule;

2nd	with 57ème :	by step of 0.05 Hz
58	65	0,075 Hz
66	79	0.10 Hz
80	103	0,125 Hz
104	131	0.25 Hz
132	137	0.5 Hz
138	141	1. Hz
142	150	1.5 Hz

  The cut-off frequency by default is thus $35{,}5\,Hz$ (the user must check that this value is coherent with the frequential contents of the entry signal, if it is not the case, it is necessary to define a list of adapted frequency).

- the spectrum is normalized according to the value of `NORMALIZES`.

When `METHOD = 'RICE'` (*cf.* §3.21.1), `NATURE_FONC = 'DSP'` (*cf.* §3.21.2), L is calculatedE spectrum of oscillator are equivalent to a spectral concentration of power (DSP) [R4.05.03] under the keyword `FUNCTION` (*cf.* §3.21.1). The function must be of `NOM_PARA = 'FREQ'`.

3.21.1 Operand FUNCTION

♦ `FUNCTION = F`

Name of the function on which the operation is carried out.
Does not apply to the concepts of the type `tablecloth`.

3.21.2 Operands NATURE/NATURE_FONC

◊ `NATURE =`

Nature of the size of the tablecloth created by the order `CALC_FONCTION`.

ACCE	Spectrum of pseudo-acceleration	$\ddot{u}(t) = \omega_i^2 u(t)$
QUICKLY	Spectrum of pseudo-acceleration	$\dot{u}(t) = \omega_i u(t)$
DEPL	Spectrum of pseudo-acceleration	$u(t)$

◊ `NATURE_FONC =/ 'ACCE'`
`/ 'DSP'`

Nature of the function which is used to build the spectrum. The choice is imposed according to the selected method: `'ACCE'` (temporal signal) or `'DSP'` (spectral concentration of power in acceleration). This keyword makes it possible to overload it `NOM_RESU` function specified under the keyword `FUNCTION` when this one is created by `RECU_FONCTION` [U4.32.03].

3.21.3 Operand METHOD

◊ `METHOD =`

Name of METHOD that one wishes to use : the method by default, 'NIGAM', is detailed in the document [R5.05.01]. If the method is chosen 'HARMO', then the spectrum of answer is obtained by successive harmonic calculations (for various Eigen frequencies of oscillator) via a FFT/IFFT of the signal as starter. If one chooses 'RICE' then one determines the SRO are equivalent for the data of a DSP.

3.21.4 Operand AMOR_REDUIT

◊ AMOR_REDUIT = lam

lam = $\left(\xi_1, \ldots, \xi_i, \ldots\right)$

List of reduced depreciation : example 0.01,0.05,…

3.21.5 Operands FREQ/LIST_FREQ

~ ◊ FREQ = lfre

lfre = $\left(\varphi_1, \ldots, \varphi_i, \ldots\right)$. List of the frequencies.

~ ◊ LIST_FREQ = lfreq

List of the frequencies provided under a concept listr8.

The frequencies must be strictly positive.

3.21.6 Operand NORMALIZES

♦ = R NORMALIZES

The spectrum of oscillator will be normalized with the value R (value of pseudo-acceleration), this value is recalled in the file of message.

3.21.7 Operand DURATION

♦ DURATION = tsm

This keyword is to be informed only if METHODE=' RICE' and thus NATURE_FONC=' DSP'. It is the duration of the strong phase of the earthquake. The seismic signal is regarded as stationary over this duration. This value is necessary to evaluate the factor of peak which intervenes in the formula of Vanmarcke to find the SRO equivalent to the data of the DSP.

3.22 Keyword DSP

/ DSP =

Calculate the spectral concentration of power (DSP) equivalent to the data of a spectrum of answer of oscillator (SRO) with the formula of Vanmarcke, function of nature function [R4.05.03].

The spectrum of oscillator is calculable only on the functions of NOM_PARA = 'FREQ'.

For the calculation of the DSP starting from a spectrum of oscillator, one considers that

- the SRO expresses the median maximum (fractile 0.5), if not it is necessary to inform another value via the keyword FRACT.
- the DSP is worth zero for frequencies lower or equal to $1/2\pi\, Hz$.
- the spectrum is normalized according to the value of NORMALIZES.

The user must check that the frequential discretization (list of the frequencies) is sufficient compared to the frequential contents of the signals to be modelled. It is also advisable to check equivalence between the DSP and the SRO given by making pullings or by determining values of

maximum answer of an oscillator with POST_DYNA_ALEA. The operator calculates the spectral concentration of power (DSP) equivalent to the data of a spectrum of answer of oscillator (SRO) with the formula of Vanmarcke. One does not carry out an iteration to optimize did it.
An example of application is proposed in the case test ZZZZ100D.

3.22.1 Operand FUNCTION

♦ FUNCTION = sro

Name of the function defining the SRO.

3.22.2 Operands AMOR_REDUIT, LIST_FREQ, FREQ_PAS

The keyword AMOR_REDUIT and LIST_FREQ are identical to those described for SPEC_OSCI (cf. 3.21).
The keyword FREQ_PAS indicate the step of frequency if LIST_FREQ is not well informed.

3.22.3 Operands FREQ_COUP

◊ FREQ_COUP = frc

The cut-off frequency: one determines the DSP until this frequency. The SRO is prolonged (constant value corresponding to the ZPA) up to this value so necessary.

3.22.4 Operand DURATION

♦ DURATION = tsm

Duration of the strong phase of the earthquake. The seismic signal is regarded as stationary over this duration. This value is necessary to evaluate the factor of peak which intervenes in the formula of Vanmarcke.

3.22.5 Operand NORMALIZES

♦ = R NORMALIZES

One considers spectra of oscillator normalized with the value R (value of pseudo-acceleration). In general, the SRO are given in G, it is thus necessary to inform NORMALIZES = 9.81.

3.23 Attributes of the concept function at exit

3.23.1 Values by default

By default attributes of the concept function at exit of the order CALC_FONCTION are for the various options (cf orders DEFI_FONCTION [U4.31.02] and DEFI_NAPPE [U4.31.03]).

- Option DRIFT :

Interpolation : data by the function as starter

Left prolongation: EXCLUDED

Right prolongation: EXCLUDED

NOM_PARWITH = 'INST' (example) given by the function as starter

NOM_REKNOWN = 'QUICKLY' (example) given by the function as starter

- Option JUST :

Even rules that for DRIFT

- Options COMB / COMB_C :

Attributes of the first combined function.

- Option SPEC_OSCI : the result is a tablecloth

Attributes of the tablecloth:

NAME_PARA = 'AMOR'

NOM_RESU = 'DEPL' or 'QUICKLY' or 'ACCE'

Interpolation: 'LOG'

Left prolongation: 'EXCLUDED'

Right prolongation: 'EXCLUDED'

Attributes of each function:

NOM_PARA = 'FREQ'

Interpolation : 'LOG'

Left prolongation: 'EXCLUDED'

Right prolongation: 'CONSTANT'

- Option ENVELOPE :

Attributes of the first function given.

- Option FFT :

NOM_PARA = FREQ if NOM_PARA function is INST
If not it is the reverse

- Option COMPOSE :

NOM_PARA : that of the function FONC_PARA
NOM_RESU : that of the function FONC_RESU
Interpol : that of the function FONC_RESU
Prolongation: that of the function FONC_RESU

- Option EXTRACTION :

Attributes identical to those of the function given as starter

- Option ADZE :

NOM_PARA : that of the functions
NOM_RESU : that of the functions
Interpol : linear
Prolongation: 'EXCLUDED'

3.23.2 Overload of the attributes

The user can overload the attributes given by default by using the following keywords:

3.23.2.1 Operand **NOM_PARA**

◊ NOM_PARA = para

It indicates the name of the parameter (variable or X-coordinate) of the function or the tablecloth. Values currently authorized for para are:

```
/ 'TEMP'      / 'INST'          / 'EPSI'
/ 'X'         / 'Y'             / 'Z'
/ 'FREQ'      / 'SWEATERS'      / 'AMOR'
/ 'DX'        / 'DY'            / 'DZ'
/ 'DRX'       / 'DRY MARTINI'   / 'DRZ'
/ 'ABSC'
```

3.23.2.2 Operand **NOM_RESU**

◊ NOM_RESU = resu

It makes it possible to document, the function created by giving a name (8 characters) to the function. Except exception (cf [§3.1], [§3.2], [§3.5]), this name is not tested.

3.23.2.3 Operand Interpol

◊ Interpol

When the produced concept is a tablecloth, Interpol defines the type of interpolation between two consecutive values of the parameter of the tablecloth and between two functions (once those evaluated). Even direction that in DEFI_NAPPE.
When the produced concept is a function (real or complex), it defines the type of interpolation for the X-coordinates and the ordinates of the function. One waits up to two values. If it only one value is provided, it is used for the X-coordinates and the ordinates. Even direction that in DEFI_FONCTION.

3.23.2.4 Operands PROL_DROITE/ PROL_GAUCHE

◊ PROL_DROITE and PROL_GAUCHE

They define the type of prolongation on the right (respectively on the left) of the field of definition of the variable:

- 'CONSTANT' for a prolongation with the last (or the first) value of the function,
- 'LINEAR' for a prolongation along the first definite segment (PROL_GAUCHE) or of the last definite segment (PROL_DROITE),
- 'EXCLUDED' if the extrapolation of the values apart from the field of definition of the parameter is prohibited.

3.23.2.5 Operands NOM_PARA_FONC / INTERPOL_FONC / PROL_DROITE_FONC / PROL_GAUCHE_FONC

These keywords make it possible to modify the attributes of the tablecloth and apply to the parameters of the functions of this one. They have the same meaning as the keywords without the suffix FONC.

- NOM_PARA_FONC is the name of the parameter of the functions (as in DEFI_NAPPE).
- INTERPOL_FONC is the type of interpolation for the X-coordinates and ordinates of the functions of the tablecloth (identical to the keyword Interpol keyword factor DEFI_FONCTION of DEFI_NAPPE).
- PROL_GAUCHE_FONC/PROL_DROITE_FONC the prolongations of the functions (identical to the keywords define PROL_GAUCHE/PROL_DROITE keyword factor DEFI_FONCTION of DEFI_NAPPE).

3.24 Operand INFORMATION

◊ INFORMATION

If INFO=2, one prints the function (IMPR_FONCTION format TABLE) in the file MESSAGE.

4 Examples

4.1 Calculation of an envelope

The command file which follows:

```
DEPI=2. * pi
PAS0=DEPI/200.
LI1=DEFI_LISTE_REEL (   DEBUT=0.,
                     INTERVALLE=_F (  JUSQU_A = DEPI, NOT = PAS0))

COa = FORMULA (NOM_PARA=' INST', VALE=' cos (INST) ')
SIa = FORMULA (NOM_PARA=' INST', VALE=' sin (INST) ')

CO  = CALC_FONC_INTERP (     FONCTION=COa,    LIST_PARA=LI1,
                            NOM_PARA=' INST',
                            NOM_RESU=' DEPL',
                            PROL_GAUCHE=' EXCLU', PROL_DROITE=' LINEAIRE',
                            INTERPOL=' LIN',
                            TITRE=' FUNCTION COSINUS'   )

IF  = CALC_FONC_INTERP (     FONCTION=SIa,    LIST_PARA=LI1,
                            NOM_PARA=' INST',
                            NOM_RESU=' DEPLACEMENT',
                            PROL_GAUCHE=' EXCLU',
                            PROL_DROITE=' CONSTANT',
                            INTERPOL=' LIN',
                            TITRE=' FUNCTION SINE '    )

ENV1=CALC_FONCTION (ENVELOPPE=_F (  FUNCTION = (IF,  CO,),
                                   CRITERION = 'SUP'))
```

4.2 Calculation of the derivative of the function `if`

The orders which follow

```
der1 = CALC_FONCTION (DERIVE=_F (FONCTION= if),)

inst1 = 20. * not

TEST_FONCTION (  VALEUR=
                    _F (   FUNCTION = der1,  NOM_PARA = 'inst',
                       VALE_PARA= inst1, VALE_REFE= COa (inst1),)
              )
```

produce on the file `'RESULT'` :

```
---- FUNCTION : DER1
OK  INST         RELA    -0.016% VALE: 8.0888392298046D-01
    6.28319E-01 SHEET      0.100%   REFE: 8.0901699437495D-01
```

4.3 Concatenation of two functions

```
DFC1=DEFI_FONCTION (   NOM_PARA=' X',   NOM_RESU=' Y',
                    VALE= (   0. , 10. ,
```

```
                                  4. , 14. ,
                                  6. , 16. ,),
                         PROL_DROITE=' LINEAIRE',
                         PROL_GAUCHE=' LINEAIRE'
                        )

#

DFC2=DEFI_FONCTION (   NOM_PARA=' X',   NOM_RESU=' Y',
                         VALE= (   5. , 25. ,
                                  7. , 27. ,
                                  8. , 28. ,),
                         PROL_DROITE=' LINEAIRE',
                         PROL_GAUCHE=' LINEAIRE'
                        )

#

DFC3=CALC_FONCTION (   ASSE=_F (
                         FUNCTION = (DFC2,  DFC1,),
                         OVERLOAD = 'RIGHT')
                        )

DFC4=CALC_FONCTION (   ASSE=_F (
                         FUNCTION = (DFC1,  DFC2,),
                         OVERLOAD = 'LEFT')
                        )
```

Values of the function `dfc3` are:

```
X     =  0.     4.    5.    7.    8.
there =  10.   14.   25.   27.   28.
```

Values of the function `dfc4` are:

```
X     =  0.     4.    6.    7.    8.
there =  10.   14.   16.   27.   28.
```

4.4 Composition of two functions

```
fonc1 = DEFI_FONCTION  (  NOM_PARA  =  'X',
                          NOM_RESU  =  'F',
                          VALE      =  (  0. ,      0. ,
                                          2. ,      5. ,
                                          3. ,     10. ,
                                          5. ,     15. ,
                                          7. ,     13. ,
                                          8. ,     10. ,
                                          10. ,  9. ,
                                          12. ,  8. ,
                                          13. ,  5. ,
                                          15. ,  1. ,
                                          20. ,  0.    )  )

fonc2 = DEFI_FONCTION  (  NOM_PARA  =  'INST',
                          NOM_RESU  =  'X',
                          VALE      =  (  0. ,      0. ,
                                          0.1,   2. ,
                                          0.2,   4. ,
                                          0.3,   6. ,
                                          0.4,   8. ,
```

```
                                                0.5 ,   10. ,
                                                0.6 ,   12. ,
                                                0.7 ,   14. ,
                                                0.8 ,   16. ,
                                                0.9 ,   18. ,
                                                1.0 ,   20.    )  )

comp1 = CALC_FONCTION   (   COMPOSE = _F (   FONC_RESU =   fonc1,
                                            FONC_PARA =   fonc2  )
                        )
```

Values of the function `comp1` are:

```
inst   =  0.  0.1    0.2   0.3   0.4   0.5   0.6   0.7   0.8   0.9   1.0
F      =  0.  5.    12.5  14.   10.    9.    8.    3.    0.8   0.4   0.
```