LIÈGE université
Urban & Environmental Engineering

LIEGE UNIVERSITY

Urban & Environmental Engineering

Structural Engineering

USER'S MANUAL FOR SAFIR (version 2025)

A COMPUTER PROGRAM FOR ANALYSIS OF STRUCTURES SUBJECTED TO FIRE

Part 3: Structural calculations

by

Jean-Marc Franssen* & Thomas Gernay**

22/10/2024

* jm.franssen@uliege.be – Liege University, Liege, Belgium
** tgernay@jhu.edu – Johns Hopkins University, Baltimore, MD, U.S.A.

Table of content

1. Introduction

1.1 Strain decomposition

The stress-strain relationships in the load bearing materials are non-linear and are temperature dependent. In structures exposed to fire, the materials are subjected to initial strains ε_i, thermal expansions ε_{th} and stress related effects ε_s. The main hypothesis of the constitutive models in all materials in SAFIR is called strain decomposition; the stresses are caused by stress related strains, being the difference between the total strain ε_{total}, obtained from the nodal displacements, and the initial and thermal strains.

$$\epsilon_\sigma = \varepsilon_{total} - \varepsilon_i - \epsilon_{th}$$

1.2 Convergence criteria

SAFIR uses an iterative procedure to converge on the correct solution for each time step with a certain tolerance. The PRECISION given in the data file (see series 21) is a small value that must be reached at different times in SAFIR calculations to have convergence. A good PRECISION value is dependent on the type of structure that is being analysed and on information from preliminary runs. However, if the user does not know which value to choose, a value of 0.002 can be used as a starting point. After the first run, an examination in the output of the out-of-balance forces (command PRINTFHE, see series 29.B) and increments of displacement (command PRINTDEPL, see series 29.B) during subsequent iterations can help the user to modify the corresponding PRECISION value to obtain an acceptable solution.

2. Detailed description of the input file for structural analyses

SERIES 1: COMMENTS

A. Multiple cards (can be 0)

As many lines as required may be introduced by the user at the beginning of the file to comment the content of the file.

B. 1 card

1 blank line to indicate that the comments are finished.

SERIES 2: QUANTITY OF NODES

1 card

`NNODE`, *nnode*

- `NNODE`

  Command.
- *nnode*

  Quantity of nodes of the structure.

SERIES 3: QUANTITY OF DIMENSIONS

1 card

`NDIM`, *ndim*

- `NDIM`

  Command.
- *ndim*

  Quantity of dimensions of the structure (quantity of global axis).
  - *ndim* = 2 for 2D structures.
  - *ndim* = 3 for 3D structures.

SERIES 4: DEGREES OF FREEDOM

1 card

`NDOFMAX`, *ndofmax*

- `NDOFMAX`

  Command.

- *ndofmax*

  Maximum quantity of degrees of freedom per node.

  - If *ndim* = 2
    if TRUSS elements are present, *ndofmax* ≥ 2
    if BEAM elements are present, *ndofmax* ≥ 3

  - If *ndim* = 3
    if TRUSS elements are present, *ndofmax* ≥ 3
    if SOLID elements are present, *ndofmax* ≥ 3
    if SHELL elements are present, *ndofmax* ≥ 6
    if BEAM elements are present, *ndofmax* ≥ 7

SERIES 5: STRATEGY

A. 1 card, optional

`NCORES`, *ncores*

- `NCORES`

  Command.

- *ncores*

  Quantity of cores of the CPU of the computer used by matrix solver. The default value is 1, in which case this card may be omitted.

This card can be used to force SAFIR to use more than 1 core, if present on the computer.
ncores is set to 2 if this card is not present.

B. 1 card

One line, first line of three possible line series

`COMMAND1, COMMAND2`

- `COMMAND1 = STATIC, STATICCOLD` or `DYNAMIC`

  `STATIC` is used if the structure or one part of it is submitted to the fire and a static analysis is required.

STATICCOLD is used for the determination of the ultimate load bearing capacity of a structure which is not submitted to the fire, i.e., at room temperature[1]. DYNAMIC is used if the structure or one part of it is submitted to the fire and a dynamic analysis is required.

- COMMAND2 = PURE_NR or APPR_NR

  Type of convergence procedure used during the iterations performed in each time step. The program can use a pure Newton-Raphson procedure (PURE_NR) or a approached Newton-Raphson procedure (APPR_NR)., and Experience tends to show that convergence may be faster with APPR_NR in structures that contains shell finite elements. PURE_NR may be preferred otherwise.

C. 1 card, optional

VBA_FACTOR, *rvba*

- VBA_FACTOR

  Command.

- *rvba*

  Scalar, from 0.0 to 1.0, which expresses the degree of linear extrapolation for the first step of each iteration in the Velocity Based Algorithm.

  rvba = 0.0 => Displacements at the first iteration at step t_{n+1} = displacements at the last converged step t_n.

  rvba = 1.0 => Displacements at the first iteration at step t_{n+1} are extrapolated linearly from the displacements at steps t_{n-1} and t_n. This option is highly recommended with imposed displacements. It may be less efficient during sudden variations in the behaviour (e.g., at 20 seconds if the loads are applied with then load function FLOAD)

  Any value between 0.0 and 1.0 is also possible.

SERIES 6: NUMBER OF LOADS

A. 1 card

NLOAD,*nload*

- NLOAD

  Command.

- *nload*

  Quantity of groups of loads. One group of loads is made of different loads of different loads, possibly of different types, that will vary with time according to the same function.

[1] Utilisation of STATIC would also work in this case, possibly at the price of a few more iterations. Utilisation of DYNAMIC is also possible.

B. <u>1 card</u>

`HYDROST`, *nhydrost*

- `HYDROST`

  Command.

- *nhydrost*

  Quantity of hydrostatic loads. Hydrostatic loads may vary as a function of time during the calculation, depending on the level of the water table. They form a separate group of loads, which is not counted in *nload* that appears in card B.

SERIES 7: INCLINED SUPPORTS

<u>1 card</u>

`OBLIQUE`, *noblique*

- `OBLIQUE`

  Command.

- *noblique*

  Quantity of inclined supports. Every node where a boundary condition is expressed in a local system of coordinates, instead of the global system of coordinates of the structure, is an oblique support, see Series 19.

  noblique = 0 if there is no oblique support.

SERIES 8: CONVERGENCE STRATEGY

<u>1 card, optional</u>

Choice of two possible settings

- `COMEBACK,` *timestepmin*
  - `COMEBACK`

    Command.
  - *timestepmin*

    Minimum value for the time step in case of comeback.
- `NOCOMEBACK`

  Command.

`NOCOMEBACK` will be used if this card is not present

Notes :

If `NOCOMEBACK` is chosen, the simulation terminates at the first time when convergence cannot be obtained, or if an error occurs in the computational process.

If `COMEBACK` is chosen, the time step is divided by 2 if convergence cannot be obtained within a small number of iterations; time is then reset at the last converged point and the simulation restarts from there with the smaller time step. On the other hand, the time step is multiplied by 2 if convergence is obtained easily for a certain number of consecutive time steps, up to a maximum value which is defined in series 27

SERIES 9: QUANTITY OF MATERIALS

1 card

`NMAT`, *nmat*

- `NMAT`

  Command.
- *nmat*

  Quantity of different materials.

Note : if two materials have the same material law but different characteristics, it makes two different materials, e.g. S235 and S355 steel

SERIES 10: ELEMENTS

1 card

`ELEMENTS`

Command.

BEAM elements sub-series, present if BEAM finite elements are used in the structure

a) 1 card

`BEAM`, *nbeam, ngeobeam*

- `BEAM`

  Command.
- *nbeam*

  Quantity of BEAM elements in the structure.
- *ngeobeam*

Quantity of different groups of geometrical properties.

> Note : one group of geometrical properties comprises elements that have the same materials, the same cross section and the same temperature history. One ".TEM" file will be necessary to describe each of the *ngeobeam* groups.

b) 1 card

`NG`, *ng*

- `NG`

  Command.

- *ng*

  Quantity of longitudinal points of integration in elements: 2 or 3.

c) 1 card

`NFIBER`, *nfiberbeam*

- `NFIBER`

  Command.

- *nfiberbeam*

  Quantity of longitudinal fibres in the BEAM elements (the maximum value for all the different groups of geometrical properties).

TRUSS elements sub-series, present if TRUSS finite elements are used in the structure

1 card

`TRUSS`, *ntruss, ngeotruss*

- `TRUSS`

  Command.

- *ntruss*

  Quantity of TRUSS elements in the structure.

- *ngeotruss*

  Quantity of different groups of geometrical properties.

> Note : one group of geometrical properties comprises elements that have the same material, the same cross sectional area and the same temperature history.

SHELL elements sub-series, present if SHELL finite elements are used in the structure

a) 1 card

`SHELL`, *nshell, ngeoshell*

- `SHELL`

  Command.

- *nshell*

  Quantity of SHELL elements in the structure.

- *ngeoshell*

Quantity of different groups of geometrical properties.

> Note : one group of geometrical properties comprises elements that have the same materials, the same thickness, the same reinforcing bars and the same temperature.

b) 1 card

`NGTHICK`, *ngshellthick*

- `NGTHICK`

  Command.

- *ngshellthick*

  Quantity of points of integration on the thickness of the elements. Cannot be less than 2 and cannot be more than 10.

c) 1 card

`NREBARS`, *nrebarsmax*

- `NREBARS`

  Command.

- *nrebarsmax*

  Quantity of REBAR layers in the SHELL elements. The maximum value found in all groups of geometrical properties must be given.

SOLID elements sub-series, present if SOLID finite elements are used in the structure

a) 1 card

`SOLID`, *nsolid*

- `SOLID`

  Command.

- *nsolid*

  Quantity of SOLID elements in the structure.

b) 1 card

`NG`, *ng*

- `NG`

  Command.

- *ng*

  Quantity of integration points in each direction. Valid entries are 1, 2 and 3.

SPRING elements sub-series, present if SPRING finite elements are used in the structure

1 card

`SPRING`, *nspring*

- `SPRING`

  Command.

- *nspring*

  Quantity of SPRING elements in the structure.

Last line of series

1 card

`END_ELEM`

Command.

SERIES 11: NODES

A. 1 card

`COMMAND`

- `COMMAND = NODES`

  The position of the nodes will be given in a Cartesian system of coordinates.

- `COMMAND = NODES_CYL`

  The position of the nodes will be given in a cylindrical system of coordinates. Cylindrical coordinates are transformed for the internal solution process by the following equations:

  if *ndim* = 2 (r,θ) => X = r cos(θ); Y = r sin(θ)

  if *ndim* = 3 (r,θ,Z) => X = r cos(θ); Y = r sin(θ); Z

  Note that θ is in degrees. The transformation is made after all nodes have been read and the automatic generations and repeats have been made, see card C and D.

Several cards are then written, following the format of B, C or D, until all *nnode* nodes (see series 2) have been given their position.

B. 1 card for 1 node

`NODE`, *nno*, *rcoordg(1,nno)*, …, *rcoordg*(*ndim*,*nno*)

- `NODE`

  Command.

- *nno*

  Number of the specific node.

- *rcoordg(1,nno)*

  First global coordinate (in meter) of node *nno*.

…

- *rcoordg*(*ndim*,*nno*)

  Last global coordinate of node *nno* in the direction of the *ndim* global axis.

C. 1 card (optional) for automatic generation

`GNODE`, *nno*, *rcoordg(1,nno)*, …, *rcoordg*(*ndim*,*nno*)

- `GNODE`

  This command is used for automatic equidistant generation of nodes between the previously defined node and the node *nno*.

- *nno*

  Number of the specific node.

- *rcoordg*(*1*,*nno*)

  First global coordinate (in meter) of node *nno*.

…

- *rcoordg*(*ndim*,*nno*)

  Last global coordinate (in meter) of node *nno*.

D. 1 card (optional) for repeating a series of nodes

`REPEAT`, *nno*, *deltac(1),* …, *deltac*(*ndim*), *kgene*

- `REPEAT`

  This command is used to repeat the definition of *nno* previously defined nodes with an increment in coordinates.

- *nno*

  Quantity of nodes to be repeated.

- *deltac(1)*

  Increment for the first coordinate (in meter).

…

- *deltac*(*ndim*)

  Increment on the coordinate *ndim*.

- *kgene*

  Number of times that the previous *nno* defined nodes must be repeated.

SERIES 12: SUPPORTS AND IMPOSED DISPLACEMENTS

A. 1 card

`FIXATIONS`

Command.

B. 1 card (optional) at each node where the solution follows a defined function[2] of time and the reaction must be calculated

`BLOCK`, *nno*, *cblock(1,nno),* …, *cblock*(*ndofmax*,*nno*)

- `BLOCK`

  Command.

- *nno*

  Number of the specific node where the solution must not be calculated.

- *cblock(1,nno*)

  Function describing the displacement for the first D.o.F at this node with respect to time. Type `NO` if the displacement is not prescribed for this D.o.F.

- *cblock*(*2,nno*)

[2] See User's manual for SAFIR, Part 1, section 5-5

Function describing the displacement for the second D.o.F at this node with respect to time. Type `NO` if the displacement is not prescribed for this D.o.F.

…

- *cblock*(*ndofmax*,*nno*)

  Function describing the displacement for the last D.o.F at this node with respect to time. Type `NO` if the displacement is not prescribed for this D.o.F.

> Notes :
>
> 1) This command is used to apply supports, as fixed displacement, with the function `F0` being used.
> 2) The next lines with the `SAME` commands can come only after all `BLOCK` commands have been entered

C. <u>1 card (optional) for each slave node</u>

`SAME`, *nno1*, *nno2*, *ctrav(1)*, …, *ctrav(ndofmax)*

- `SAME`

  Command.

- *nno1*

  Number of the specific slave node.

- *nno2*

  Number of the master node.

- *ctrav*(1)
  - `YES` if the displacement is the same at node *nno1* as at node *nno2* for the D.o.F 1.
  - `NO` otherwise.

…

- *ctrav(ndofmax*)
  - `YES` if the displacement is the same at node *nno1* as at node *nno2* for the D.o.F *ndofmax*.
  - `NO` otherwise.

D. <u>1 card (optional) to repeat the previous `SAME` command</u>

`REPEAT`, *n*, *incr*, *ctrav(1)*, …, *ctrav(ndofmax)*

- `REPEAT`

  Command.

- *n*

  Number of times that the previous `SAME` command must be repeated.

- *incr*

  Increment on the node numbers *nno1* and *nno2* of the previous `SAME` command.

- *ctrav(1*)
  - `YES` if the displacement is the same at node *nno1* as at node *nno2* for the D.o.F 1.
  - `NO` if there is no master-slave relation for this D.o.F.

…

- *ctrav(ndofmax)*
  - `YES` if the displacement is the same at node *nno1* as at node *nno2* for the D.o.F *ndofmax*.
  - `NO` if there is no master-slave relation for this D.o.F.

E. <u>1 card (optional) to create master-slave relationship between all nodes that have the same coordinates</u>

`SAMEALL`, *ctrav(1)*, …, *ctrav(ndofmax)*

- `SAMEALL`

  Command. All the nodes of the structure that have the same coordinates (with a precision of 10^{-8} m) will automatically be attributed a master-slave relationship, depending on the entries given for *ctrav(i)*.
- *ctrav(1)*
  - `YES` if the solution is the same for the D.o.F 1.
  - `NO` otherwise.
- *ctrav(ndofmax)*
  - `YES` if the solution is the same for the D.o.F *ndofmax*.
  - `NO` otherwise.

Note : A node cannot be defined as a slave in a relationship and as a master in another relationship. Therefore, if a given node is used in two or more relationships of slave-master, it must be defined either as the master in all these relationships, or as the slave in all these relationships.

For example, consider that nodes 1, 2 and 3, which have 7 D.o.F, coincide and that these three nodes must have the same displacement in the **z** direction (in global axis). The following relationships are valid:

```
SAME    2     1    NO    NO   YES    NO    NO    NO    NO
SAME    3     1    NO    NO   YES    NO    NO    NO    NO
```

The following relationships are not valid :

```
SAME    2     1    NO    NO   YES    NO    NO    NO    NO
SAME    1     3    NO    NO   YES    NO    NO    NO    NO
```

F. <u>1 card</u>

`END_FIX`

Command indicating the end of the series.

SERIES 12: BEAM ELEMENTS

Note : this series is skipped if no BEAM element is present in the structure

A. 1 card

`NODOFBEAM`

Command.

B. BEAM elements sub-series, one sub-series for each of the *ngeobeam* groups of geometrical properties

a) 1 card, file name

[`cfilename`]

Full name of the file where the information on this section type can be found. Usually the extension is ".TEM". File name is left justified.

b) 1 card, material translation, one line added for each different material used in the section

`TRANSLATE`, *matl, matg*

- `TRANSLATE`

  Command.
- *matl*

  Local number of this material in the ".TEM" file of this section type.
- *matg*

  Global number of this material in the structure.

Notes : *matl* starts from 1 for the first material in this section type. The second line is for the second local material, etc. Those lines are necessary because of the strategy used for the data files. One structure can be made of several BEAM section types, each of them being described in one ".TEM" file. In each of those ".TEM" files, the different materials are given numbers starting from 1. It is necessary to indicate at the level of the structure, which global material number corresponds to the numbers given in the ".TEM" files.

c) 1 card, last line

`END_TRANS`

Command.

C. 1 card, BEAM elements list, in increasing order, from 1 to *nbeam*

`ELEM`, *nelem, nodofbeam(1,ne), …, nodofbeam(4,ne), itypebeam(ne)*

- *nelem*

  Number of this element.
- *nodofbeam(1,nelem)*

  First end node of this element.

- *nodofbeam(3,nelem)*

  Third (i.e. central) node of this element.
- *nodofbeam(2,nelem)*

  Second end node of this element.
- *nodofbeam(4,nelem)*

  4th node of this element (present only if *ndim*=3).
- *itypebeam(nelem)*

  The section type of this element.

D. 1 card (optional) for automatic generation

`GELEM`, *nelem, nodofbeam(1,nelem), …, nodofbeam(4,nelem), itypebeam(nelem), kgene*

- *nelem*

  Number of this element.
- *nodofbeam(1,nelem)*

  First end node of this element.
- *nodofbeam(3,nelem)*

  Third (i.e. central) node of this element.
- *nodofbeam(2,nelem)*

  Second end node of this element.
- *nodofbeam(4,nelem)*

  4th node of this element (present only if *ndim*=3).
- *itypebeam(nelem)*

  The section type of this element.
- *kgene*

  Allows the generation from the previously defined element up to this one. *kgene* gives the increment on the first 3 nodes. If, *ndim* =3, The 4th node remains the same.

E. 1 card (optional) for repeating a series of elements

`REPEAT`, *nelem, nincr123, nincr4, nincrtype, ntimes*

- *nelem*

  The *nelem* previously defined elements will be repeated.
- *nincr123*

  Increment on the nodes 1, 2 and 3.
- *nincr4*

  Increment on the node 4 (present only if *ndim* = 3).
- *nincrtype*

  Increment on the type of the element.
- *ntimes*

  How many times these *nelem* elements will be repeated.

For example, the following sequence

```
       ELEM    1    1    2    3  108    1
      GELEM    8   15   16   17  108    1    2
       ELEM    9   18   19   20  108    1
      GELEM   16   32   33   34  108    1    2
       ELEM   17   35   36   37  108    1
      GELEM   24   49   50   51  108    1    2
       ELEM   25   52   53   54  108    1
      GELEM   32   66   67   68  108    1    2
       ELEM   33   69   70   71  108    1
      GELEM   40   83   84   85  108    1    2
```

can be replaced by the following one

```
       ELEM    1    1    2    3  108    1
      GELEM    8   15   16   17  108    1    2
     REPEAT    8   17    0    0    4
```

to generate

```
      ELEM     NODE1     NODE3     NODE2     NODE4       TYPE     LENGTH
         1         1         2         3       108          1    0.1288E+01
         2         3         4         5       108          1    0.1288E+01
         3         5         6         7       108          1    0.1288E+01
         4         7         8         9       108          1    0.1288E+01
         5         9        10        11       108          1    0.1200E+01
         6        11        12        13       108          1    0.1200E+01
         7        13        14        15       108          1    0.1200E+01
         8        15        16        17       108          1    0.1200E+01
         9        18        19        20       108          1    0.1288E+01
        10        20        21        22       108          1    0.1288E+01
        11        22        23        24       108          1    0.1288E+01
        12        24        25        26       108          1    0.1288E+01
        13        26        27        28       108          1    0.1200E+01
        14        28        29        30       108          1    0.1200E+01
        15        30        31        32       108          1    0.1200E+01
        16        32        33        34       108          1    0.1200E+01
        17        35        36        37       108          1    0.1288E+01
        18        37        38        39       108          1    0.1288E+01
        19        39        40        41       108          1    0.1288E+01
        20        41        42        43       108          1    0.1288E+01
        21        43        44        45       108          1    0.1200E+01
        22        45        46        47       108          1    0.1200E+01
        23        47        48        49       108          1    0.1200E+01
        24        49        50        51       108          1    0.1200E+01
        25        52        53        54       108          1    0.1288E+01
        23        47        48        49       108          1    0.1200E+01
        24        49        50        51       108          1    0.1200E+01
        25        52        53        54       108          1    0.1288E+01
        26        54        55        56       108          1    0.1288E+01
        27        56        57        58       108          1    0.1288E+01
        28        58        59        60       108          1    0.1288E+01
        29        60        61        62       108          1    0.1200E+01
        30        62        63        64       108          1    0.1200E+01
        31        64        65        66       108          1    0.1200E+01
        32        66        67        68       108          1    0.1200E+01
        33        69        70        71       108          1    0.1288E+01
        34        71        72        73       108          1    0.1288E+01
        35        73        74        75       108          1    0.1288E+01
        36        75        76        77       108          1    0.1288E+01
        37        77        78        79       108          1    0.1200E+01
        38        79        80        81       108          1    0.1200E+01
        39        81        82        83       108          1    0.1200E+01
        40        83        84        85       108          1    0.1200E+01
```

SERIES 15: SOLID ELEMENTS

Note : this series is skipped if no SOLID element is present in the structure

A. 1 card

`NODOFSOLID`

Command.

B. 1 card,

`[cfilename]`

Full name of the file where the information on the temperature distribution in the structure can be found. Usually the extension is ".OUT". This file is the result of a 3D thermal analysis.

C. 1 card, SOLID elements list, one line for each SOLID element

`ELEM`, *nsl, n1, n2, n3, n4, n5, n6, n7, n8, nmat, res1, res2, res3*

- `ELEM`

  Command.
- *nsl*

  Number of the element.
- *n1*

  Node 1.
- *n2*

  Node 2.
- *n3*

  Node 3.
- *n4*

  Node 4.
- *n5*

  Node 5.
- *n6*

  Node 6.
- *n7*

  Node 7.
- *n8*

  Node 8.
- *nmat*

  Number of the material.

- *res1*

  Residual stress in the element in the direction of the global axis 1.
- *res2*

  Residual stress in the element in the direction of the global axis 2.
- *res3*

  Residual stress in the element in the direction of the global axis 3.

D. 1 card (optional) for repeating a series of elements

`REPEAT`, *nsl, n1,…, number*

- `REPEAT`

  Command.
- *nsl*

  Quantity of elements to be repeated.
- *n1*

  Node increment.
- *number*

  Number of times that the *nsl* elements must be repeated.

Notes :

1) The last nodes are given the number 0 if the SOLID element has less than 8 nodes.

2) Only hexahedral and octahedral elements can be used in a mechanical analysis. Tetrahedral elements are not supported.

SERIES 16: SHELL ELEMENTS

Note : this series is skipped if no SHELL element is present in the structure

A. 1 card

`NODOFSHELL`

Command.

B. SHELL elements sub-series, one sub-series for each section type

a) 1 card, file

`[cfilename]`

Name of the file where the information concerning this section type is read.

Notes : the name of the ".TSH" files that describe the section heated by the HASEMI fire is, for each section type, the name of ONE of the relevant ".TSH" file. For example, [s0156_3.tsh].

The information about the re-bar layers has to be present only in this file, not in the other ".TSH" files of the same section type that describe the temperature at the other points of integration.

Consequently, all the SHELL elements of one section type have the same re-bars.

b) 1 card

`THICKNESS`, *thickshell*

- `THICKNESS`

  Command.

- *thickshell*

  Thickness of the structural part of this section type. All material above or below the structural part is considered as non-load bearing (insulating material, for example)

c) 1 card, optional

`Z0`, *Z0*

- `Z0`

  Command.

- *Z0*

  Position in the local z direction of the plane that contains the nodes of the shell.

Note: If this card is absent, *Z0* = 0 is assumed.

d) 1 card

`MATERIAL`, *mat*

- `MATERIAL`
  Command.
- *mat*
  GLOBAL material number of this section type. This is the material of the plain section, to which layers of re-bars can be added. This material must have a plane-stress 2D constitutive model.

e) <u>1 card</u>

`REBARS`, *nbars*

- `REBARS`
  Command.
- *nbars*
  Quantity of re-bars layers in this section type.

The following card f), g), h), i) and j) are repeated *nbars* times.

f) <u>1 card</u>

`REBARMAT`, *mat*

- `REBARMAT`
  Command.
- `mat`
  Global number of the material of this layer. This material must be a 1D load bearing material.

g) <u>1 card</u>

`SECTION`, *A*

- `SECTION`
  Command.
- *A*
  Cross sectional area of this layer (in m^2/m).

h) <u>1 card (optional)</u>

`SIGMA_RES`, *sigma*

- `SIGMA_RES`
  Command.
- *sigma*

Initial stress of the bars in this layer (N/m^2).

Notes :

1) Initial stresses can be introduced in the bars if the bars are made of steel. If not, the stress can be entered as 0; or this card can be omitted.
2) Positive values are given for tension in the bars.
3) This capability is used for pretensioning. The stress in the bars will be reduced during the first timestep because of the elastic shortening developing in concrete.
4) For introducing initial stresses in the material of the plain section, see Section 3.7 'Instructions for introducing residual stresses in steel SHELL elements'.

i) <u>1 card</u>

`LEVEL`, *z*

- `LEVEL`

  Command.

- *z*

  Position of this layer in the local **z** direction.

j) <u>1 card</u>

For each bar layer, there are two methods to give the orientation of the bars in the plane of the element.

<u>Method 1</u>: with respect to the local system of coordinates of each element.

`ANGLE`, *angle*

- `ANGLE`

  Command.

- *angle*

  Angle in degrees between the local **x** axis and the layer of rebars, see Figure 1.

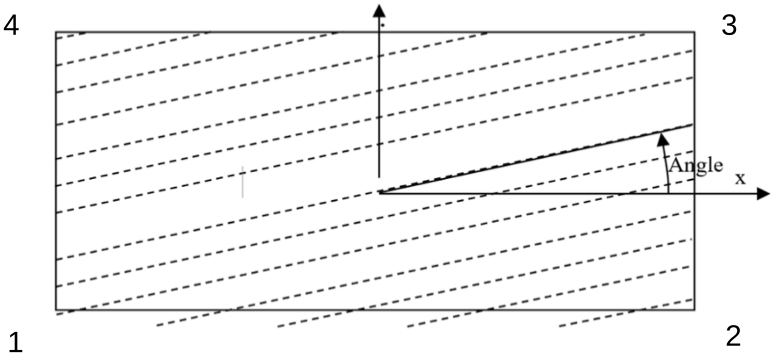


Figure 1: orientation of rebars in SHELL F.E. in local axes

<u>Method 2</u>**:** with respect to the global system of coordinates of the structure.

`NORMAL`, *n1, n2, n3*

- `NORMAL`

Command.

- *n1, n2, n3*

  < N_1 ; N_2 ; N_3 > is a vector in the global system of coordinates of the structure. The norm of the vector does not have to be 1.
  The technique used is illustrated in Figure 2.

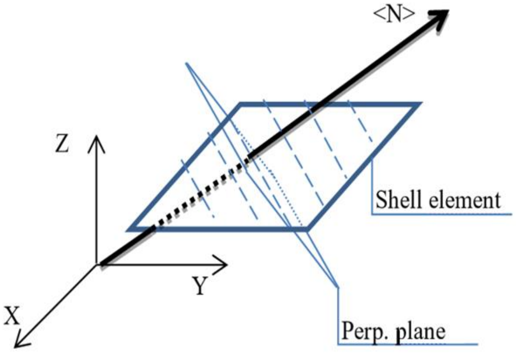


Figure 2: method of the normal

The bars have the orientation of the line which is the intersection between the SHELL element and a plane that is perpendicular to the normal.
If the norm of the vector is 0, then the orientation of this bar layer is perpendicular, in each element, to the previous bar layer (not possible for bar layer 1).

C. 1 card, one line for each SHELL element

`ELEM`, *nsh, n1, n2, n3, n4, itypeshell(nsh), kgene*

- `ELEM`

  Command.
- *nsh*

  Number of the element.
- *n1*

  Node 1.
- *n2*

  Node 2.
- *n3*

  Node 3.
- *n4*

  Node 4.
- *itypeshell(nsh)*

  Type of geometrical section.
- *kgene*

  Automatic generation on the element number.

D. <u>1 card (optional) for repeating a series of elements</u>

`REPEAT`, *nsh, n1, number*

- `REPEAT`

  Command.
- *nsh*

  Quantity of elements to be repeated.
- *n1*

  Node increment.
- *number*

  Number of times that the *nsh* elements must be repeated.

Example of series 16 if *ngeoshell* = 2

```
NODOFSHELL

walls.tsh
 THICKNESS     0.200
        Z0     0.000
  MATERIAL     1
    REBARS     2
  REBARMAT     2
   SECTION     600.E-6
     LEVEL     0.010
    NORMAL     0.   0.   1.
  REBARMAT     2
   SECTION     600.E-6
     LEVEL    -0.010
    NORMAL     0.   0.   0.

slabs.tsh
 THICKNESS     0.180
        Z0     0.000
  MATERIAL     1
    REBARS     2
  REBARMAT     2
   SECTION     1200.E-6
     LEVEL    -0.065
    NORMAL     1.   0.   0.
  REBARMAT     2
   SECTION     240.E-6
     LEVEL    -0.050
    NORMAL     0.   0.   0.

      ELEM    1      1601      1602      1802      1801    1
      ELEM    2      1602      1603      1803      1802    1
      ELEM    3      1603      1604      1804      1803    2
...
```

SERIES 17: TRUSS ELEMENTS

Note : this series is skipped if no TRUSS element is present in the structure

A. 1 card

```
NODOFTRUSS
```

Command.

B. *ngeotruss* cards, TRUSS elements file,

one card for each different TRUSS section type used (*ngt* varies from 1 to *ngeottruss*).
`[cfilename]`, *geotruss(1,ngt), geotruss(2,ngt), imattruss(ngt).*

- `[cfilename]`

  Name of the file where the temperatures concerning this section types are read.
- *geotruss(1,ngt)*

  Cross sectional area of this section type.
- *geotruss(2,ngt)*

  Initial stress of this section type.
- *imattruss(ngt)*

  Global number of the material in this section type.

Notes : if [`cfilename(ngt)`] is left blank, then :

- This must be the case for all the section types.
- There is only one element in each *ngt.*
- The elements must be linked to nodes which belong to SOLID elements.
- The temperature of each TRUSS element is the average of the temperature of its 2 nodes, calculated with SOLID elements.

C. <u>1 card, TRUSS elements list</u>

`ELEM`, *ntr, nodottruss(1,ntr), nodoftruss(2,ntr), igeotruss(ntr)*

- `ELEM`

  Command.
- *ntr*

  Number of this element.
- *nodoftruss(1,ntr)*

  First node of this element.
- *nodoftruss(2,ntr)*

  Second node of this element.
- *igeotruss(ntr)*

  Number of the section type for this element.

D. <u>1 card (optionalal) for automatic generation</u>

`GELEM`, *ntr, nodoftruss(1,ntr), nodoftruss(2,ntr), igeotruss(ntr), kgene*

- `GELEM`

  Command.
- *ntr*

  Number of this element.
- *nodoftruss(1,ntr)*

  First node of this element.
- *nodoftruss(2,ntr)*

  Second node of this element.
- *igeotruss(ntr)*

  Number of the section type for this element.
- *kgene*

  Allows for automatic generation from the previously defined element to the element *ntr*.

SERIES 18: SPRING ELEMENTS

Note : this series is skipped if no SPRING element is present in the structure

Each SPRING element is applied on one node of the structure.

The positive direction of the spring element is always defined from the foundation to the structure, which means that, in the output, a positive force in the spring, corresponding to tension, will be pulling on the structure, whereas a negative force will push on the structure.

In 2D models, the direction of the element is defined by the angle at the foundation between the global axis **x** and the element, in counterclockwise direction, as shown on Figure 3.

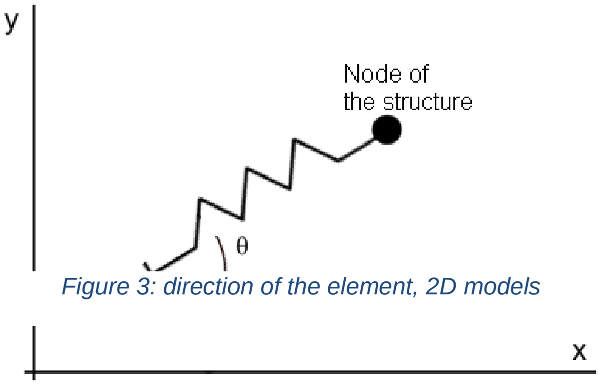


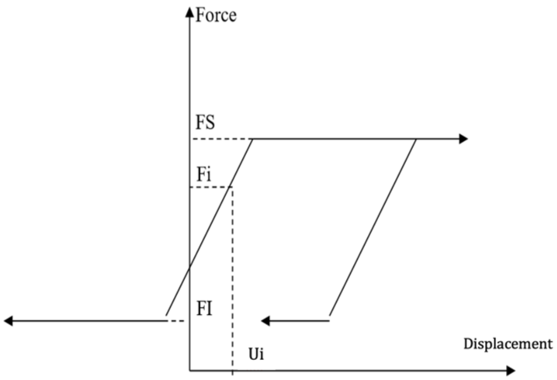
Figure 3: direction of the element, 2D models

In 3D models, the direction of the element is defined by a 3D vector that has the direction from the foundation to the node of the structure. This vector does not have to be normalised.

This element has no material. Its behaviour is given directly in term of a force-displacement relationship as shown on Figure 4.
The force is comprised between two limits: the upper limit FS and the lower limit FI (either positive or negative, with FS > FI).

Figure 4: force-displacement diagram of the SPRING

The slope of the elastic part is the stiffness K.



The state of the spring at time *t*=0 is defined by the initial displacement Ui and initial force Fi.
The behaviour in unloading is plastic.

Note : in contradiction with the sign convention that is used in the output, the values of FS, FI and Fi (as well as K) are positive when the spring pushes on the structure. This is by analogy with active and passive pressure when the spring is used to model a soil beside a wall or a foundation underneath a beam.

All forces from the F-u diagram will be multiplied by the area of influence *area*. Consequently, the stiffness is also multiplied by *area*.

The area of influence is there to help when the properties of the spring refer to properties of soil mechanics. The properties are then expressed in term of N/m³ and these properties are all multiplied by the area of influence.

For example, in a 2D model made of a beam on an elastic foundation with uniform soil characteristics, with all elements having a length of 0.800 meters, and representing a slice of a tunnel which is 0,100 m wide, the area of the springs applied at the first and the last node of the beam would be 0.100 m wide x (0.800 m long / 2) = 0.040 m². The area of all other springs would be 0.100 m wide x 2 x (0.800 m long / 2) = 0.080 m².

The load applied to the beam in N/m should then be the load for the 0.100 m slice.

In Diamond, you can plot the forces in the springs. If the beam of the example mentioned above is subjected to a uniformly distributed load, the force in the springs at the end nodes is twice smaller than for the other nodes.

You can also plot the pressure, and you will see that the pressure is uniform.

If the properties of the springs have nothing to do with soil, just put an area of 1 m².

A. 1 card

`NDFSPRING`

Command.

B. 1 card, one line for each spring element

For 2D structures

`ELEM`, *nspr, nnode, teta, FS, FI, k, area, ui, Fi*

- `ELEM`

  Command.
- *nspr*

  Number of the element.
- *nnode*

  Node where this element is attached.
- *teta*

  Angle between **x** axis and this element, in degree.
- *FS*

  Superior limit of the force, in N/m².
- *FI*

  Inferior limit of the force, in N/m²
- *K*

  Stiffness of the element, In N/m³.
- *area*

  Area of influence, in m²
- *ui*

Displacement in the element in the configuration of reference of the structure (at time t=0) in meters.

- *Fi*

  Force in the element in the configuration of reference of the structure, in N/m².

For 3D structures

`ELEM`, *nspr, nnode, cx, cy, cz, FS, FI, K, area, ui, Fi*

- `ELEM`

  Command.
- *nspr*

  Number of the element.
- *nnode*

  Node where this element is attached.
- *cx*

  Component x of the vector that defines the direction of the element, from the foundation to the structure.
- *cy*

  Component y of the vector that defines the direction of the element
- *cz*

  Component z of the vector that defines the direction of the element.
- *FS*

  Superior limit of the force.
- *FI*

  Inferior limit of the force.
- *K*

  Stiffness of the element.
- *area*

  Area of influence.
- *ui*

  Displacement in the element in the configuration of reference of the structure (at time t=0).
- *Fi*

  Force in the element in the configuration of reference of the structure.

SERIES 19: OBLIQUE SUPPORTS

A. 1 card, one line for each oblique support

For 2D structures

`INCLIN`, *ni, nj*

- `INCLIN`

  Command.

- *ni*

  Node where a boundary condition is expressed in a local system of coordinates.
- *nj*

  Another node of the structure.

ni and *nj* define the direction in which the node *ni* can move. It cannot move perpendicularly to this direction.

For 3D structures

Boundary condition applied to the displacements
`INCLIN`, *ni, nj, nk*

- `INCLIN`

  Command.
- *ni*

  Node where a boundary condition is expressed in a local system of coordinates.
- *nj, nk*

  2 other nodes of the structure.

ni, nj and *nk* define the plane in which the node *ni* can move. It cannot move out of this plane.

Boundary condition applied to the rotations

`INCLIN`, *-ni, nj, nk*

- `INCLIN`

  Command.
- *ni*

  Node where a boundary condition is expressed in a local system of coordinates. A negative value has been entered for *ni* to indicate that this line is about rotation and not displacement.
- *nj, nk*

  2 other nodes of the structure.

ni, nj and *nk* define the plane in which the node *ni* can have rotations. It cannot rotate along an axis perpendicular to this plane.

B. 1 card, one line to indicate that this is the end of the series

```
END_INCLIN
```

Command.

SERIES 20 (OPTIONAL): RELAXATION

This series is used to relax the internal forces at some DOF's at the end of some elements.
The internal force F at the designated degree of freedom is computed according to the following equation:

$$F = K\,(u_s - u_i) \leq F_u$$

where K is a stiffness.

u_s is the displacement of the node of the structure, in the local system of coordinates of the beam.

u_i is the displacement of the internal node at the end of the element, in the local system of coordinates of the beam. This internal node must not be introduced by the user; it is added by SAFIR at the same location as the node of the structure.

F_u is the maximum value that the force can reach. When this value is reached, the stiffness goes down to 0 and the value of the force remains fixed at F_u (plastic behaviour).

A. <u>1 card</u>

`RELAX_ELEM`

Command.

B. <u>1 card</u>

`BEAMS`

Command. This line indicates that at least one BEAM finite element has one or several degrees of freedom that are relaxed. It has been introduced here to allow, in future developments of SAFIR, to relax also DoF's at the ends of other element types.

C. <u>1 card, 1 line or 2 lines for each BEAM element that has one or several relaxed degrees of freedom.</u>

C.1 First line of the card. Stiffness of the relaxed degrees of freedom.

`ELEM`, *nbm, k(1), k(2), …, k(idimk-1)*

- `ELEM`

  Command.

- *nbm*

  Number of the BEAM element.

- *k(1)*

Stiffness for the relaxation at the first degree of freedom of the beam (translation along local axis **x** of the beam at node 1).
If the value is 0, there is no spring between the BEAM element and the node of the structure. The relaxation is total and the force at the end of the BEAM element is equal to 0.
If the value is negative, there is no relaxation. The displacement at the end of the beam is equal to the displacement of the structure.

- *k(2)*

  Stiffness at the second degree of freedom of the beam.

…

- *k(idimk-1)*

  Stiffness at the last rotational degree of freedom of node 2.

Notes :

1) *idimk* is the quantity of degrees of freedom of the beam, 7 for 2D beams and 15 for 3D beams.
2) The degree of freedom of the central node cannot be relaxed.
3) If the behaviour of the relaxation is elasto-plastic (Card C.2 will be present), the stiffnesses of this card C.1 cannot be equal to 0. All degrees of freedom *i* must either be without relaxation (k(*i*) < 0) or have a finite stiffness (k(*i*) > 0)

C.2 Optional second line of the card. Ultimate capacity of the relaxed degrees of freedom. If this line is omitted, all degrees of freedom that have been relaxed in the previous line are infinitely elastic. This card allows introducing elastic-perfectly plastic relationships.

`Fu`, *F(1), F(2), …, F(idimk-1)*

- `Fu`

  Command.

- *F(1)*

  Ultimate force that can be transmitted at the first degree of freedom of the beam (translation along local axis **x** of the beam at node 1).
  If this degree of freedom has not been relaxed or has been fully relaxed in the previous line, the value 0. is given here.
  If this degree of freedom has been relaxed with an elastic stiffness in the previous line, the value of the ultimate force is given here.

- *F(2)*

  Ultimate force at the second degree of freedom of the beam.

…

- *F(idimk-1)*

  Ultimate moment at the last rotational degree of freedom of node 2.

D. 1 card

`END_BEAMS`

Command.

The next commands, until command `END_RELAX`, are optional. They are introduced only if the mechanical properties of the springs are time dependent.

E. 1 card

`TIME_RELAX`, *ntime*

- `TIME_RELAX`

  Command.

- *ntime*

  Quantity of time steps for which the evolution of the properties of the springs will be given.

F. *ntime* cards

time, value

- *time*

  Time (in seconds) for which the value is given.

- *value*

  Coefficient that multiplies the mechanical characteristics (stiffness, strength) of the springs given in the `ELEM` commands. Typically, the value is 1. At the beginning of the fire and it decreases progressively with time.

Note :

The stiffness and the strength of the springs can be modified as a function of time only if they have finite values different from 0.

A stiffness which is infinite because $k < 0$, or a stiffness which is equal to 0 because $k = 0$ in series C.1, cannot have their value modified with time.

A strength which is infinite because the series C.2 "F_u" is absent, cannot have its value modified with time.

G. 1 card

`END_RELAX`

Command.

SERIES 21: PRECISION

1 card

`PRECISION`, *precision*

- `PRECISION`

  Command.

- *precision*

  Small value that must be reached for convergence.

SERIES 22 (OPTIONAL): LIMITING DISPLACEMENT

1 card

`MAX_DISPL`, *max_displ*

- `MAX_DISPL`

  Command.

- *max_displ*

  Maximum value admitted for the displacements (or rotations). If this value is exceeded for an incremental displacement during iteration or for a total displacement after convergence, a `COMEBACK` is performed, if it has been foreseen. If not, the run is terminated.

Note : if this series is not present, *max_displ* is given the value of 10 (meters or radians).

SERIES 23: LOADS

This series is repeated *nload* times, see Series 6

Note : if *nload* is equal to 0, then the 2 following lines must nevertheless be present:

```
    LOADS
END_LOAD
```

A. 1 card, loading

`LOADS`

Command.

B. 1 card, loading function

`FUNCTION`, *cforce(nlo)*

- `FUNCTION`

  Command.

- *cforce(nlo)*

  Function of time that will multiply the loads given in cards C. to K. This can be a SAFIR function or the name of a file (not longer than 10 characters), see User’s manual of SAFIR, Part 1, Section 5.5.

C. <u>several cards for loads applied on a node</u>

`NODELOAD`, *nno, load(1), load(2), …, load(iloc)*

- `NODELOAD`

  Command.

- *nno*

  Number of the node where loads are applied.

- *load(1)*

  Load for degree of freedom 1.

- *load(2)*

  Load for degree of freedom 2.

…

- *load(iloc)*

  Load for degree of freedom *iloc.*

> Note : iloc = minimum(*ndof*,6) where *ndof* is the quantity of degrees of freedom of this node (The 7th DoF of end nodes of 3D beams cannot be loaded.

D. <u>Several cards for uniformly distributed loads on BEAM elements, one line added for each element with a uniformly distributed load applied.</u>

The direction of these loads is according to the global system of coordinates, and they are measured per meter in the direction of the local axis **x** of the BEAM finite element. For example, the dead weight of a prismatic column that extends along the global axis **z** can be described by a `DISTRBEAM` load in the direction of the global axis **z**.

`DISTRBEAM`, *nbm, trav(1), trav(2), …, trav(ndim)*

- `DISTRBEAM`

  Command.

- *nbm*

  Number of the specific BEAM under a distributed load.

- *trav(1)*

  Uniformly distributed load in the direction of the global axis 1 (in Newton per meter of beam).

- *trav(2)*

  Uniformly distributed load in the direction of the global axis 2.

- *trav(3)*

  Uniformly distributed load in the direction of the global axis 3 (in 3D structures).

E. 1 card, loading on BEAM elements, uniformly distributed loads, (optional)

`GDISTRBEAM`, *nbm, trav(1), trav(2), …, trav(ndim), kgene*

- `GDISTRBEAM`

  Command.

- *nbm*

  Number of the specific BEAM under a distributed load.

- *trav(1)*

  Uniformly distributed load in the direction of the global axis 1.

- *trav(2)*

  Uniformly distributed load in the direction of the global axis 2.

- *trav(3)*

  Uniformly distributed load in the direction of the global axis 3 (in 3D structures).

- *kgene*

  Distributed loads are generated between the previously defined element and the present element with a step on the element numbers of *kgene.*

F. 1 card, loading on BEAM elements, trapezoidal distributed loads in local axis, one line added for each element with a distributed load applied.

The direction of these loads is according to the local system of coordinates of the beam, and they are measured per meter in the direction of the local axis **x** of the beam. Such load can be used, for example, to impose a wind pressure that varies in a trapezoidal manner along an inclined roof.

`TRAPLOCBM`, *nbm, trav(1), trav(2), …, trav(2*ndim)*

- `TRAPLOCBM`

  Command.

- *nbm*

  Number of the BEAM under a trapezoidal distributed load.

- *trav(1)*

  Distributed load in the direction of the local axis 1 at node 1 of the element.

- *trav(2)*

  Distributed load in the direction of the local axis 2 at node 1.

- *trav(3)*

  Distributed load in the direction of the local axis 3 at node 1 (not present if NDIM = 2).

- *trav(ndim+1)*

  Distributed load in the direction of the local axis 1 at node 2 of the element.

- *trav(ndim+2)*

  Distributed load in the direction of the local axis 2 at node 2.
- *trav(6)*
  - Distributed load in the direction of the local axis 3 at node 2 (not present if NDIM = 2).

G. 1 card, loading on BEAM elements, trapezoidal distributed loads in local axis, (optional)

`GTRAPLOCBM`, *nbm1, nbm2, trav(1), trav(2), …, trav(2*ndim), kgene*

- `GTRAPLOCBM`

  Command.
- *nbm1*

  Number of the first BEAM under a trapezoidal distributed load.
- *nbm2*

  Number of the last BEAM under a trapezoidal distributed load.
- *trav(1)*

  Distributed load in the direction of the local axis 1 at node 1 of the element *nbm1*.
- *trav(2)*

  Distributed load in the direction of the local axis 2 at node 1 of the element *nbm1*.
- *trav(3)*

  Distributed load in the direction of the local axis 3 at node 1 of the element *nbm1* (not present if *ndim* = 2).
- *trav(ndim+1)*

  Distributed load in the direction of the local axis 1 at node 2 of the element *nbm2*.
- *trav(ndim+2)*

  Distributed load in the direction of the local axis 2 at node 2 of the element *nbm2*.
- *trav(6)*
  - Distributed load in the direction of the local axis 3 at node 2 of the element *nbm2* (not present if *ndim* = 2).
- *kgene*

  Distributed loads are generated with a step on the element numbers of *kgene*. The value of the trapezoidal loads at the intermediate nodes are calculated by linear interpolation based on the length of the elements.

H. 1 card, loading on BEAM elements, trapezoidal distributed loads in global axis, one line added for each element with a distributed load applied.

The direction of these loads is according to the global system of coordinates and they are measured per meter in the direction of the local axis **x** of the beam. Such load can be used, for example, to impose the dead weight of a tapered member.

`TRAPGLOBM`, *nbm, trav(1), trav(2), …, trav(2*ndim)*

I. 1 card, loading on BEAM elements, trapezoidal distributed loads in local axis, (optional)

`GTRAPGLOBM`, *nbm1, nbm2, trav(1), trav(2), …, trav(2*ndim), kgene*

The format of these commands (cards H and I) is the same as the format of `TRAPLOCBM` and `GTRAPLOCBM`, except that the distributed loads are here described in the direction of the global system of coordinates.

J. 1 card, loading on SHELL elements, one line added for each element with a distributed load applied.

`DISTRSH`, *nsh, trav(1), trav(2), trav(3)*

- `DISTRSH`

  Command.
- *nsh*

  Number of the SHELL element under a distributed load.
- *trav(1)*

  Uniformly distributed load in the direction of the global axis 1.
- *trav(2)*

  Uniformly distributed load in the direction of the global axis 2.
- *trav(3)*

  Uniformly distributed load in the direction of the global axis 3.

K. 1 card, loading on SHELL elements (optional)

`GDISTRSH`, *nsh, trav(1), trav(2), trav(3), kgene*

- `DISTRSH`

  Command.
- *nsh*

Number of the SHELL element under a distributed load.

- *trav(1)*

  Uniformly distributed load in the direction of the global axis 1.
- *trav(2)*

  Uniformly distributed load in the direction of the global axis 2.
- *trav(3)*

  Uniformly distributed load in the direction of the global axis 3.
- *kgene*

  Increment on the element number.

L. 1 card, loading end of series

`END_LOAD`

Command.

SERIES 24: HYDROSTATIC LOADS

This series is repeated nhydrost time

A. 1 card

`WATERTABLE`, *cwatertable*

- `WATERTABLE`

  Command.
- *cwatertable*

  Chain of (maximum 10) characters, function describing the level (in meters) of the water table, see Users manual for SAFIR 2022, Part 1, Section 5.5.

B. 1 card

`SPECWEIGHT`, *gamma*

- `SPECWEIGHT`

  Command.
- *gamma*

Specific weight of the fluid, in m(N/m³).

> Note : The first meter in the dimension of this value represents the horizontal distance between adjacent BEAM elements. For example, if 2D frames are representing frames that are, in fact, 3 meters away in the direction perpendicular to the plane of the frame, the specific weight of water to be considered here is not 10.000 N/m³, but 30.000 m(N/m³).
> If the specific weight is positive, the load on each element is in the direction of the local axis **y** of the element. It is possible to enter a negative value of the specific weight to change the direction of the load.

C. 1 card, list of the BEAM elements loaded by this hydrostatic load

`HYDROBM`, *nbm*

- `HYDROBM`

  Command.

- *nbm*

  Number of the element loaded by this hydrostatic load.

D. 1 card, list of the BEAM elements loaded by this hydrostatic load, automatic generation (optional)

`GHYDROBM`, *nbm, kgene*

- `HYDROBM`

  Command.

- *nbm*

  Number of the element loaded by this hydrostatic load.

- *kgene*

  The elements from the previously defined element to the element *nbm*, with an increment of *kgene*, are loaded by this hydrostatic load.

E. 1 card

`END_HYDRO`

Command.

> Note: A BEAM element loaded by a hydrostatic load will be loaded by a `TRAPLOCBM` load. The distributed load at each end node of the element is given by P = *gamma* x *depth*
> where *depth* is the difference between the value of *cwatertable* and the coordinate *ndim* of the node in the deformed configuration. It is thus assumed that gravity is in the direction of –**y** for 2D structures and –**z** for 3D structures.
> If the value of *depth* at a given node is negative, it is replaced by 0.
> The pressure will be applied on the deformed shape of the structure.

SERIES 25: MASS CHARACTERISTICS

Notes:

1) This series is present only if `DYNAMIC` has been chosen in series 6. If this is the case, cards A and G must be present even if there is no mass (although it does not really make sense to make a dynamic analysis if there is no mass in the model).

```
      MASS
END_MASS
```

2) In SAFIR, masses and forces are totally independent. The masses introduced produce no force and the forces are not linked to any mass. Consequently, if a force of X Newton is produced by gravity, a mass of X/10 kg must normally be also introduced in the data. If a force is produced by wind, no mass has to be introduced.
3) As a consequence of the fact that mass don't create loads, mass scaling can be used to facilitate convergence.

A. <u>1 card</u>

`MASS`

Command.

B. <u>1 card, concentrated mass on nodes, one line added for each concentrated mass linked to a node</u>

`M_NODE`, *nno, mass(1), mass(2), …, mass(ndof)*

- `M_NODE`

  Command.

- *nno*

  Number of the node where the mass is applied.

- *mass(1)*

  Mass linked to degree of freedom 1.

- *mass(2)*

  Mass linked to degree of freedom 2.

…

- *mass(ndof)*

  Mass linked to degree of freedom *ndof*.

Notes:

1) A mass linked to a displacement is in kg. A mass linked to a rotation is in kgm.
2) Usually, a concentrated mass linked to a displacement is active in all directions. Only in some cases can a mass be inactive in a particular direction (for example, a ball laying on a horizontal surface might be active in the direction perpendicular to this surface, and not active in the directions parallel to this surface)

C. 1 card, distributed mass on BEAM elements, one line added for each BEAM element with a distributed mass applied

`M_BEAM`, *nbm, trav(1), trav(2)*

- `M_BEAM`

  Command.

- *nbm*

  Number of the BEAM under a distributed mass.

- *trav(1)*

  Uniformly distributed mass applied on the BEAM element (kg/m).

- *trav(2)*

  Rotational inertia around the longitudinal axis of the element (kgm/m). Only present if a 3D analysis is made. If the value is not known, an arbitrary value of 1 can be given.

D. 1 card, distributed mass on BEAM elements, automatic generation (optional)

`GM_BEAM`, *nbm, trav(1), trav(2), kgene*

- `GM_BEAM`

  Command.

- *nbm*

  Number of the BEAM under a distributed mass.

- *trav(1)*

  Uniformly distributed mass applied on the BEAM element (kg/m).

- *trav(2)*

  Rotational inertia around the longitudinal axis of the element (kgm/m). Only present if a 3D analysis is made.

- *kgene*

  Increment on the element number. (Distributed mass are generated between the previously defined element and the present element).

E. 1 card, distributed mass on SHELL elements, one line added for each SHELL element with a distributed mass applied

`M_SHELL`, *nsh, trav(1)*

- `M_SHELL`

  Command.

- *nsh*

  Number of the SHELL element under a distributed mass.

- *trav(1)*

Uniformly distributed mass applied on the SHELL element (kg/m^2).

F. 1 card, distributed mass on SHELL elements, automatic generation (optional)

`GM_SHELL`, *nsh, trav(1), trav(2), kgene*

- `GM_SHELL`

  Command.

- *nsh*

  Number of the SHELL element under a distributed mass.

- *trav(1)*

  Uniformly distributed mass applied on the SHELL element (kg/m^2).

- *kgene*

  Increment on the element number.

G. 1 card, mass end of series

`END_MASS`

Command.

SERIES 26: MATERIAL DESCRIPTION

A. 1 card

`MATERIALS`

Command.

Material description sub-series. One sub-series comprising card a) and b) must be entered for each of the *nmat* material types (*nm*=1,*nmat*).

a) 1 card

`CMAT`

Name of the material, `CMAT` is replaced by one of the material names given hereafter.

b) 1 card

paracold

The values of the parameters associated with this material are introduced in the *paracold* vector.

INSULATION MATERIAL TYPES

If `CMAT = INSULATION, USER1, USER2, USER3, USER4, USER5, X_GYPSUM, C_GYPSUM, SFRM_PROBA,` no parameter is necessary because these materials do not carry any stress. In this case, the second line is a blank line.

UNIAXIAL MATERIAL TYPES

If `CMAT` = `ELASTIC` (this material is valid only at 20°C.)

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio

If `CMAT` = `BILIN` (this material is valid only at 20°C.)

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Yield strength.
paracold(4,nm)	Slope of the hardening branch.

```
     BILIN
210000.E6       0.3    355.E6   21000.E6
```

If `CMAT` = `RAMBOSGOOD` (this material is valid only at 20°C.)

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio.
paracold(3,nm)	l_p, limit of proportionality.
paracold(4,nm)	n, exponent of the law.
paracold(5,nm)	K, factor of the law.

$$\varepsilon = \frac{\sigma}{E} \qquad \text{for} \quad \sigma \le l_p$$

$$\varepsilon = \frac{\sigma}{E} + \left(\frac{\sigma - l_p}{K}\right)^n \qquad \text{for} \quad \sigma > l_p$$

If `CMAT` = `CALCON_ETC, SILCON_ETC`

paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Compressive strength (≥0).
paracold(4,nm)	Tensile strength (≥0).

The `CALCON_ETC`, `SILCON_ETC` materials take into account transient creep strain explicitly. The models are described in: Gernay T., Franssen J.M. (2012). “A formulation of the Eurocode 2 concrete model at elevated temperature that includes an explicit term for transient creep”. *Fire Safety Journal*, *51*, 1-9.

If `CMAT` = `CALCONC_EN, SILCONC_EN, LWCONC_EN`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

If `CMAT` = `CALCONCEC2, SILCONCEC2`

This material is from ENV 1992-1-2 (obsolete)

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

paracold(5,nm) < 0 if peak stress strain e_{c1} = minimum value (= 0.015 above 600°C, stiffer).

= 0 if peak stress strain e_{c1} = recommended value.

> 0 if peak stress strain e_{c1} = maximum value (= 0.025 above 600°C, more ductile).

If `CMAT` = `SILHSC1ETC, SILHSC2ETC, SILHSC3ETC,CALHSC1ETC, CALHSC2ETC, CALHSC3ETC`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

If `CMAT` = `SILHSC1_EN, SILHSC2_EN, SILHSC3_EN, CALHSC1_EN, CALHSC2_EN, CALHSC3_EN`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

If `CMAT` = `CALCONC_PR, SILCONC_PR`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

paracold(5,nm) Time at which this concrete is cast. Before and until this time, the material does not carry any stress or have any stiffness. Only the displacements that will take place after this time will induce stresses.

If `CMAT` = `CALCONETCL, SILCONETCL`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

paracold(5,nm) % loss of compressive strength in cooling [0;1].

If `CMAT` = `CACOPRBWE, SICOPRBWE`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength (≥0).

paracold(4,nm) Tensile strength (≥0).

paracold(5,nm) CDF quantile of Weibull distribution for probabilistic compressive strength]0;1[.

These materials are similar to `CALCON_ETC` or `SILCON_ETC` but with a probabilistic temperature-dependent compressive strength according to the model in: Qureshi et al. (2020). “Probabilistic models for temperature dependent strength of steel and concrete”. *JSE ASCE*, *146*(6).

If `CMAT` = `CALCO_COLD, SILCO_COLD` (these materials are valid only at 20°C.)

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength, f_{cm}, in N/m^2.

Concrete according to Eq. (3.14) of EN 1992-1-1

If `CMAT` = `PARABCONC` (this material is valid only at 20°C.)

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength.

paracold(4,nm) Tensile strength.

paracold(5,nm) Strain at compressive strength.

paracold(6,nm) Ultimate strain.

If `CMAT` = `AL6061_T6, AL6063_T6, AL5083_O, AL5083_H12`

(aluminium)

paracold(2,nm) $f_{0.2}$

paracold(3,nm) f_p

paracold(4,nm) $\varepsilon_{rupture}$ in %.

If `CMAT` = `SS1.4301, SS1.4307, SS1.4318, SS1.4420` (austenitic stainless steels from FprEN1993-1-2: 2023-05)

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) 0.2 % proof strength fy

paracold(4,nm) Ultimate strength fu

If `CMAT` = `STEELEC2EN`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength.

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold*(4,NM) [N/m²K].

`PROCESS`

Chain of character that indicates the fabrication process of the reinforcing bar. It can be:

- `HOTROLLED` for hot rolled bars (columns 2, 4 and 6 in Table 3.2a of EN 1992-1-2).
- `COLDWORKED` for cold worked bars (columns 3, 5 and 7 in Table 3.2a of EN 1992-1-2).

`CLASS`

Chain of character that indicates the class of ductility of the reinforcing bars. It can be:

- `CLASS_A` for Class A (low) ductility.
- `CLASS_B` for Class B (high) ductility.
- `CLASS_C` for Class C (very high) ductility.

```
STEELEC2EN
 210000.E6    0.3    500.E6    600.    0.3E6    COLDWORKED    CLASS_C
```

If CMAT = STEELEC3, STEELEC3EN, STEELEC3DC, STEELEC2, PSTEELA16, STEEL_WPB

paracold(1,nm) Young's modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength.

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold*(4,nm) [N/m²K].

If CMAT = STEELSL

paracold(1,nm) Young's modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength.

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than paracold(4,nm) [N/m²K].

paracold(9,nm) Slenderness *b*/*t* of the plate where the material is present, with: *b* the width of the plate, for example H-2(t_f+r) for the web of an I profile, B-t_w-2r for the flanges. *t* the thickness of the plate.

paracold(10,nm) Quantity of supports of the plate where the material is present
= 3 for oustand plates (flanges)
= 4 for internal plates (webs)

The STEELSL material is described in the reference: Franssen, J.M., Cowez, B., Gernay, T. (2014), "Effective stress method to be used in BEAM finite elements to take local instabilities into account", Fire Safety Science 11, in Proc. of the 11th IAFSS Symposium, Christchurch, New Zealand, Feb 10-14, pp. 544-557.

If CMAT = STEC3PROBA

paracold(1,nm) Young's modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength.

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than paracold(4,nm) [N/m²K].

paracold(6,nm) Value of the standard normal parameter ε for the probabilistic reduction of yield strength with temperature.

The `STEC3PROBA` material has the same expression of stress-strain relationship as steel of Eurocodes but the reduction of yield strength with temperature follows the logistic EC3-based probabilistic model proposed in: Khorasani N.E., Gardoni P., Garlock M. (2015). “Probabilistic fire analysis: material models and evaluation of steel structural members”. *JSE*, *141*(12).

If `CMAT` = `WOODEC5` or `CMAT = WOOD2020`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength.

paracold(4,nm) Tensile strength.

paracold(7,nm) Limit strain in compression *(optional)*

The last parameter is optional. If no value is input, the stress-strain law in compression is elastic-brittle. If a value of limit strain is input, the stress-strain law in compression is elastic-perfectly-plastic up to the specified limit strain. The limit strain must be larger than the ratio of compressive strength over modulus.

The strength and modulus are fully irreversible during cooling.

WOODEC5 is the material of EN 1995-1-2:2004 and WOOD2020 is the material of the new generation of Eurocode, prEN 1995-1-2:2021

If `CMAT` = `WOODPRBWE`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength.

paracold(4,nm) Tensile strength.

paracold(5,nm) CDF quantile of Weibull distributions for probabilistic compressive and tensile strengths]0;1[.

paracold(7,nm) Limit strain in compression *(optional)*

The material is similar to `WOODEC5` but with probabilistic temperature-dependent reduction factors for compressive and tensile strengths according to the model in Garcia-Castillo E, Gernay T, Paya-Zaforteza I. (2023). “Probabilistic models for temperature-dependent compressive and tensile strengths of timber”. *Journal of Structural Engineering ASCE*, 149 (2): 04022239.

If `CMAT` = `WOODECPF`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength.

paracold(4,nm) Tensile strength.

paracold(5,nm) Air cooling (>= 0) or water cooling (<0).

paracold(7,nm) Limit strain in compression *(optional)*

The material is similar to `WOODEC5` but with post-fire strength properties based on test data. The strength and modulus recover partly during cooling from temperature <300°C. The post-fire data and model is described in: Zhao J, Liu H, Gernay T, Chen Z, Zhao S, Yang S. (2024). “Investigation on the residual mechanical properties of glulam after exposure to elevated temperature”. *Construction and Building Materials*, 422, 135851.

If `CMAT` = `USER_STEEL`

paracold(1,nm) Young’s modulus at 20°C.

paracold(2,nm) Poisson’s ratio at 20°C.

paracold(3,nm) Yield strength at 20°C.

paracold(4,nm) critical temperature (in °C) beyond which the yield strength is not fully recovered during cooling.

paracold(5,nm) the rate of decrease of the residual yield strength if the temperature has exceeded the critical temperature given in paracold(4,nm) [N/m²K].

This `USER_STEEL` material has the same expression of stress-strain relationship as steel of Eurocodes but it will behave at elevated temperatures according to the decreasing curves specified in the file “*USER_STEEL.TXT*” that the user has to create and locate in the same folder as the input file.

In the file "*USER_STEEL.TXT*", k_E, k_{fy}, k_{fp}, ε_{th}, ε_y, ε_t and ε_u are given at different temperatures. Between two temperatures, a linear interpolation is performed by SAFIR.

Note : to have the same thermal elongation as in the material `STEELEC3EN` for all temperatures, the first value written in the file must be equal to -1.

Structure of the file "*USER_STEEL.TXT*"

One line

`Number_of_T:`, *number_of_T*

- `Number_of_T:`
  Command.
- *number_of_T*
  Quantity of elevated temperatures at which the values of the reduction factors are given.

One line

```
T    KE   Kfy   Kfp      EPSth        EPSy       EPSt       EPSu
```

One line for each temperature added to series, *number_of_T* lines

T, $k_E(T)$, $k_{fy}(T)$, $k_{fp}(T)$, $\varepsilon_{th}(T)$, $\varepsilon_y(T)$, $\varepsilon_t(T)$, $\varepsilon_u(T)$

- T Temperature at which the reduction factors are given.
- $k_E(T)$ Reduction factor relative to the value of E (Young's modulus) at 20°C.
- $k_{fy}(T)$ Reduction factor relative to the value of f_y (effective yield strength) at 20°C.
- $k_{fp}(T)$ Reduction factor relative to the value of f_p (limit of proportionality) at 20°C.
- $\varepsilon_{th}(T)$ Thermal elongation at temperature T.
- $\varepsilon_y(T)$ Yield strain at temperature T.
- $\varepsilon_t(T)$ Limiting strain for yield strength at temperature T.
- $\varepsilon_u(T)$ Ultimate strain at temperature T.

Example : the following file describes a material that has user defined variations of E, f_y and fp, but the same thermal elongation as the steel of Eurocode 3, and the same yield strain, limiting strain and ultimate strain as Eurocode 3.

```
Number_of_T: 4
           T   KE        Kfy       Kfp     EPSth       EPSy      EPSt      EPSu
          0.   1.0       1.00      1.00      -1.       0.02      0.15      0.20
        200.   1.0       0.95      0.90      -1.       0.02      0.15      0.20
        800.   0.1       0.15      0.10      -1.       0.02      0.15      0.20
       1200.   0.0       0.00      0.00      -1.       0.02      0.15      0.20
```

If `CMAT` = `USER_ELAS`

paracold(1,nm) Young's modulus in compression at 20°C.

paracold(1,nm) Young’s modulus in tension at 20°C.
paracold(3,nm) Compressive strength at 20°C.
paracold(4,nm) Tensile strength at 20°C.

This `USER_ELAS` material is a linear elastic - brittle material, with no thermal strain. Its behaviour is linear elastic up to the defined strength, beyond which the stress drops down to zero. The stiffness and the strength can be different in tension and compression.

`USER_ELAS` allows the user to define his own reduction factors. `USER_ELAS` will behave at elevated temperatures according to the reduction factors specified in the file “*USER_ELAS.TXT*” that the user must create and locate in the same folder as the input file. In the file “*USER_ELAS.TXT*” four factors are given at different temperatures: $k_{E,c}$, $k_{E,t}$, k_{fc}, k_{ft}. Between two given temperatures, a linear interpolation is performed by SAFIR. $k_{E,c}$, $k_{E,t}$, k_{fc}, k_{ft} are the reduction factors at elevated temperatures relative to the values of the modulus E in compression, the modulus E in tension, the compressive strength, and the tensile strength at 20°C.

Structure of the file “*USER_ELAS.TXT*”

One line

`Number_of_T:`, *number_of_T*

- `Number_of_T:`
  Command.
- *number_of_T*
  Quantity of elevated temperatures at which the values of the reduction factors are given.

One line

```
T        KEc        KEt        Kfc        Kft
```

One line for each temperature added to series, *number_of_T* lines

T, $k_{Ec}(T)$, $k_{Et}(T)$, $k_{fc}(T)$, $k_{ft}(T)$

- T Temperature at which the reduction factors are given.
- $k_{Ec}(T)$ Reduction factor relative to the value of E (Young’s modulus) in compression at 20°C.
- $k_{Et}(T)$ Reduction factor relative to the value of E (Young’s modulus) in tension at 20°C.
- $k_{fc}(T)$ Reduction factor relative to the value of compressive strength at 20°C.
- $k_{ft}(T)$ Reduction factor relative to the value of tensile strength at 20°C.

If `CMAT` = `CALBFUP_EN, SILBFUP_EN`

paracold(2,nm) Poisson ratio.
paracold(3,nm) Compressive strength.

This material has the same expression of stress-strain relationship as concretes of Eurocodes but its behavior in tension at elevated temperatures will follow a multi-linear behavior according to the points specified in the file "*USER_BFUP.TXT*" that the user must create and locate in the same folder as the input file.
In the file "*USER_BFUP.TXT*", a quantity of couples (ε, σ) are given at different temperatures. The user can decide on the quantity of couples (ε, σ), as well as on the quantity of temperatures. Between two defined (ε, σ), the behavior is linear.

At a given temperature,
ε_1, σ_1 define the first point in the constitutive stress-strain law of concrete in tension.
ε_2, σ_2 defines the second point. Between the points (ε_1, σ_1) and (ε_2, σ_2), the behavior is linear.
The user defines as many points as desired.

Notes :

1) The law is assumed to start at (0,0). Therefore, the point (0,0) (origin of the stress-strain law) should not be defined in the "*USER_BFUP.TXT*" file.
2) The points must be defined in ascending order of strain ε (meaning that $\varepsilon_{1+1} > \varepsilon_i$).
3) For any strain larger than the last defined strain, the stress is set to zero. Between two temperatures, a linear interpolation is performed by SAFIR.
4) The SAFIR computation will stop if a temperature in any point in the material exceeds the highest defined temperature in "*USER_BFUP.TXT*".

Structure of the file “*USER_BFUP.TXT*”

One line

`Number_of_T:`, *number_of_T*

- `Number_of_T`

  Command.
- *number_of_T*

  Quantity of elevated temperatures at which the values of the points defining the stress-strain law in tension are given.

One line

`Number_of_EPS:`, *number_of_eps*

- `Number_of_EPS`

  Command.
- *number_of_eps*

  Quantity n of strains at which the values of the stress are given (i.e. quantity of points in the stress-strain law in tension which are specified at a given temperature).

One line

```
T       EPS1       SIG1       EPS2       SIG2     …   EPSn       SIGn
```

One line for each temperature added to series, *number_of_T* lines

$T, \varepsilon_1(T), \sigma_1(T), \varepsilon_2(T), \sigma_2(T), \varepsilon_3(T), \sigma_3(T), \ldots, \varepsilon_n(T), \sigma_n(T)$

- T Temperature at which the tensile stress-strain law is defined.
- $\varepsilon_1(T)$ First defined strain at the temperature T.
- $\sigma_1(T)$ First defined stress at the temperature T (i.e. the one corresponding to ε_1(T)).

…

- $\varepsilon_n(T)$ Last defined strain at the temperature T.
- $\sigma_n(T)$ Last defined stress at the temperature T (i.e. the one corresponding to ε_n(T)).

Example, the following file describes a material that has a behaviour in tension defined by 3 points in the stress-strain space (plus the origin 0,0). The behaviour at 200°C is plotted in Figure 5.

```
Number_of_T: 4
Number_of_EPS: 3
         T      EPS1       SIG1      EPS2       SIG2      EPS3       SIG3
        0.    0.0001       3.00    0.0003       0.50    0.0006       0.20
      200.    0.0001       2.40    0.0003       0.40    0.0006       0.16
      600.    0.0001       0.05    0.0003       0.02    0.0006       0.01
     1200.    0.0001       0.05    0.0003       0.02    0.0006       0.01
```

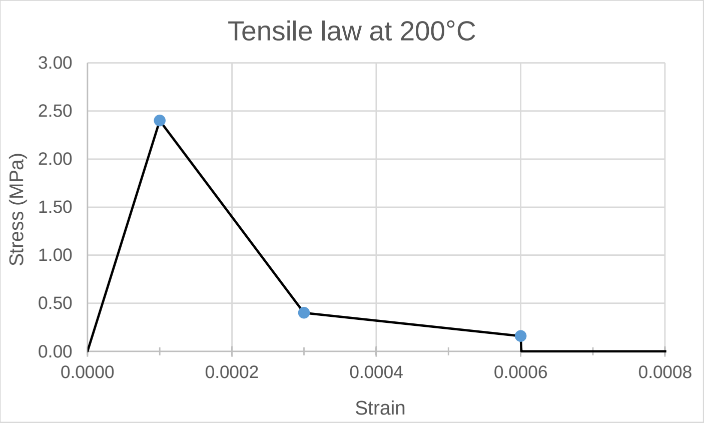


Figure 5: behaviour in tension at 200°C

If `CMAT` = `USER_CONC`

paracold(2,nm) Poisson ratio.
paracold(3,nm) Compressive strength.

This `USER_CONC` material is a uniaxial concrete material with a behavior that can be defined by the user. It requires a file "*USER_CONC.TXT*" that the user must create and locate in the same folder as the input file. In this file, k_{fc}, $\varepsilon_{c1,\theta}$, $\varepsilon_{cu1,\theta}$, n and $\varepsilon_{th,\theta}$ are given at different temperatures. Between two temperatures, a linear interpolation is performed on these values by SAFIR.

The behavior in tension follows a multi-linear behavior according to the points specified in the file "*USER_CONC.TXT*". In this file, a quantity of couples (ε, σ) are given at different temperatures for the behavior in tension. The user can decide on the quantity of couples (ε, σ), as well as on the quantity of temperatures. Between two defined (ε, σ), the behavior is linear.

At a given temperature,
ε_1, σ_1 define the first point in the constitutive stress-strain law of concrete in tension.
ε_2, σ_2 defines the second point. Between the points (ε_1, σ_1) and (ε_2, σ_2), the behavior is linear.

The user defines as many points as desired.

Notes :

1) the law is assumed to start at (0,0). Therefore, the point (0,0) (origin of the stress-strain law) should not be defined in the "*USER_CONC.TXT*" file.
2) The points must be defined in ascending order of strain ε (meaning that $\varepsilon_{1+1} > \varepsilon_i$).
3) For any strain larger than the last defined strain, the stress is set to zero.
4) Between two temperatures, a linear interpolation is performed by SAFIR.
5) the SAFIR computation will stop if a temperature in any point in the material exceeds the highest defined temperature in "*USER_CONC.TXT*".
6) Units follow the convention of the software (material strength in Pascal).

In cooling, both in compression and in tension, the behavior is irreversible when the temperature cools down. In other words, the parameters of the stress-strain laws are evaluated based on the maximum temperature reached in the integration point over the history of the fire, and not on the current temperature.

Structure of the file “*USER_CONC.TXT*”

One line

`Number_of_T:`, *number_of_T*

- `Number_of_T:`

  Command.
- *number_of_T*

  Quantity of elevated temperatures at which the values of the points defining the stress-strain law in compression are given.

One line

```
T        kfc      EPSC1     EPSCU1          n      EPSth
```

One line for each temperature added to series, *number_of_T* lines

T, $k_{fc}(T)$, $\varepsilon_{c1}(T)$, $\varepsilon_{cu1}(T)$, $n(T)$

- T Temperature at which the parameters are given.
- $k_{fc}(T)$ Reduction factor relative to the value of f_c (compressive strength) at 20°C.
- $\varepsilon_{c1}(T)$ Strain at peak stress at temperature T.
- $\varepsilon_{cu1}(T)$ Ultimate compression strain at temperature T.
- $N(T)$ Exponent of compression stress-strain law at temperature T.
- $\varepsilon_{th}(T)$ Thermal elongation at temperature T.

One line

`Number_of_T:`, *number_of_T*

- `Number_of_T:`

  Command.
- *number_of_T*

  Quantity of elevated temperatures at which the values of the points defining the stress-strain law in tension are given.

One line

`Number_of_EPS:`, *number_of_eps*

- `Number_of_eps:`

  Command.
- *number_of_eps*

  Quantity m of strains at which the values of the stress are given (i.e. quantity of points in the stress-strain law in tension which are specified at a given temperature).

One line

```
T       EPS1       SIG1       EPS2       SIG2     …    EPSm       SIGm
```

One line for each temperature added to series, *number_of_T* lines

T, $\varepsilon_1(T)$, $\sigma_1(T)$, $\varepsilon_2(T)$, $\sigma_2(T)$, $\varepsilon_3(T)$, $\sigma_3(T)$, …, $\varepsilon_m(T)$, $\sigma_m(T)$

- *T* Temperature at which the tensile stress-strain law is defined.
- $\varepsilon_1(T)$ First defined strain at the temperature T.
- $\sigma_1(T)$ First defined stress at the temperature T (i.e., the one corresponding to ε_1(T)).

…

- $\varepsilon_m(T)$ Last defined strain at the temperature T.
- $\sigma_m(T)$ Last defined stress at the temperature T (i.e. the one corresponding to ε_m(T)).

The following file describes a material that has a behaviour in compression according to the Eurocode 2 part 1-2 for siliceous aggregates concrete. The material has a behaviour in tension defined by 3 points in the stress-strain space (plus the origin 0,0).

```
Number_of_T: 14
         T       kfc     EPSC1    EPSCU1         n     EPSth
        0.     1.000    0.0025    0.0200         3    0.0000
       20.     1.000    0.0025    0.0200         3    0.0000
      100.     1.000    0.0040    0.0225         3    0.0007
      200.     0.950    0.0055    0.0250         3    0.0018
      300.     0.850    0.0070    0.0275         3    0.0031
      400.     0.750    0.0100    0.0300         3    0.0049
      500.     0.600    0.0150    0.0325         3    0.0072
      600.     0.450    0.0250    0.0350         3    0.0102
      700.     0.300    0.0250    0.0375         3    0.0140
      800.     0.150    0.0250    0.0400         3    0.0140
      900.     0.080    0.0250    0.0425         3    0.0140
     1000.     0.040    0.0250    0.0450         3    0.0140
     1100.     0.010    0.0250    0.0475         3    0.0140
     1200.     0.000    0.0250    0.0475         3    0.0140
Number_of_T: 4
Number_of_EPS: 3
         T      EPS1       SIG1     EPS2       SIG2     EPS3     SIG3
        0.    0.0001     3.00e6   0.0003     0.50e6   0.0006   0.20e6
      200.    0.0001     2.40e6   0.0003     0.40e6   0.0006   0.16e6
      600.    0.0001     0.05e6   0.0003     0.02e6   0.0006   0.01e6
     1200.    0.0001     0.05e6   0.0003     0.02e6   0.0006   0.01e6
```

BIAXIAL PLANE STRESS MATERIAL TYPES

If `CMAT` = `ELPLANESTR`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Coefficient of thermal expansion.

Elastic plane stress material law. The material is valid for steel at elevated temperature and the Young's modulus and thermal strain vary according to the Eurocode 3 part 1.2.

If `CMAT` = `PLSTRVML`

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Yield strength.
paracold(3,nm)	Strain hardening modulus.

This model is a simplified model for steel at elevated temperature, with a bilinear equivalent stress-strain relationship. The model STEELEC32D is to be preferred if no problem of convergence is encountered. The parameters vary according to the Eurocode 3 part 1.2. (variation of the strain hardening modulus is the same as that of the Young’s modulus).

If `CMAT` = `STEELEC32D`

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Yield strength.
paracold(4,nm)	Maximum temperature for a reversible behaviour during cooling.
paracold(5,nm)	Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold(4,nm)* [N/m^{2}K].

If `CMAT` = `USER_STL2D`

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Yield strength.
paracold(4,nm)	Maximum temperature for a reversible behaviour during cooling.
paracold(5,nm)	Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold*(4,nm) [N/m^{2}K].

This `USER_STL2D` material has the same expression of stress-strain relationship as `STEELEC32D` but it will behave at elevated temperatures according to the decreasing curves specified in the file “*USER_STEEL.TXT*” that the user must create and locate in the same folder as the input file. The structure of “*USER_STEEL.TXT*” is the same as described for the material `USER_STEEL`. The reader is referred to `USER_STEEL` for more information.

If `CMAT` = `STEELEC3PS`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength.

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold*(4,nm) [N/m^{2}K].

If `CMAT` = `STEELCFS2D`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength (0.2% proof stress).

paracold(6,nm) Tensile strength (ultimate).

paracold(9,nm) Limiting strain for tensile strength (plastic plateau).

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold*(4,nm) [N/m^{2}K].

If `CMAT` = `CALCOETC2D, SILCOETC2D`

paracold(2,nm) Poisson ratio, recommended value: 0.2.

paracold(3,nm) Compressive strength [N/m^2].

paracold(4,nm) Tensile strength (≥0) [N/m^2].

paracold(5,nm) Strain at peak stress, recommended value: 0.0025.

paracold(18,nm) Dilatancy parameter, recommended value: 0.25 (0.20 – 0.30).

paracold(19,nm) Compressive ductility parameter, recommended value: 0.19 (0.15 – 0.25).

paracold(20,nm) Compressive damage at peak stress, recommended value: 0.30 (0.18 – 0.32) Condition: < 0.50.

paracold(21,nm) Tensile ductility parameter [N/m^2].

This parameter can be estimated by dividing the crack energy by a characteristic length. Recommended value for plain concrete: 100 [N.m/m^2] / $\sqrt{A}$ [m] where A is the area of the SHELL element.

The `CALCOETC2D, SILCOETC2D` materials are plastic-damage constitutive models for concrete. The models are described in: Gernay T., Millard A., Franssen J.M. (2013). “A multiaxial constitutive model for concrete in the fire situation: Theoretical formulation”. *Int J Solids Structures*, *50*(22-23), 3659-3673.

If `CMAT` = `CALCONC2D, SILCONC2D`

paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Compressive strength.
paracold(4,nm)	Tensile strength (≥0).
paracold(5,nm)	< 0 if peak stress strain ε_{c1} = minimum value (stiffer). = 0 if peak stress strain ε_{c1} = recommended value. > 0 if peak stress strain ε_{c1} = maximum value (more ductile).

If `CMAT` = `LWCONC2D`

paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Compressive strength.
paracold(4,nm)	Tensile strength.

If `CMAT` = `VMRANK2D`

paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Compressive strength.
paracold(4,nm)	Tensile strength.

If `CMAT` = `BLPLSTRVM`

paracold(1,nm)	Young’s modulus.
paracold(2,nm)	Poisson ratio.
paracold(3,nm)	Yield strength.
paracold(4,nm)	Slope of the hardening branch.

If `CMAT` = `BLPLSTRDP`

paracold(1,nm)	Young’s modulus.
paracold*(2,nm)*	Poisson ratio.
paracold(3,nm)	Yield strength.
paracold(4,nm)	Slope of the hardening branch.
paracold(5,nm)	Material constant α (see W.J.Chen "Plasticity in reinforced concrete.", §8.2., p. 350)

If `CMAT`(*nm*) = `BLPLSTRVM`

paracold(1,nm) E, the Young’s modulus.

paracold(2,nm) Poisson’s ratio.

paracold(3,nm) f_p, the limit of proportionality.

paracold(4,nm) Slope of the hardening branch.

Bi-linear plane stress Von Mises material law. The material is valid at room temperature.

TRIAXIAL MATERIAL TYPES

If `CMAT` = `STEELEC23D, STEELEC33D`

paracold(1,nm) Young’s modulus.

paracold(2,nm) Poisson ratio.

paracold(3,nm) Yield strength.

paracold(4,nm) Maximum temperature for a reversible behaviour during cooling.

paracold(5,nm) Rate of decrease of the residual yield strength when the maximum temperature has been greater than *paracold*(4,nm) [N/m²K].

If `CMAT` = `CALCOETC3D, SILCOETC3D`

paracold(2,nm) Poisson ratio.

paracold(3,nm) Compressive strength [N/m²].

paracold(4,nm) Tensile strength (≥0) [N/m²].

paracold(5,nm) Strain at peak stress, recommended value: 0.0025.

paracold(18,nm) Dilatancy parameter, recommended value: 0.25 (0.20 – 0.30).

paracold(19,nm) Compressive ductility parameter, recommended value: 0.19 (0.15 – 0.25).

paracold(20,nm) Compressive damage at peak stress, recommended value: 0.30 (0.18 – 0.32).
Condition: < 0.50.

paracold(21,nm) Tensile ductility parameter [N/m²], recommended value: 400 N/m².
N.B. This parameter can be estimated by dividing the crack energy by a characteristic length. Recommended value:

100 [N.m/m^2] / / $\sqrt[3]{V}$ [m] where V is the volume of the SOLID element.

The `CALCOETC3D, SILCOETC3D` materials are plastic-damage constitutive models for concrete. The models are described in: Gernay T., Millard A., Franssen J.M. (2013). “A multiaxial constitutive model for concrete in the fire situation: Theoretical formulation”. *Int J Solids Structures*, *50*(22-23), 3659-3673.

SERIES 27: TIME_DISCRETIZATION

A. 1 card

`TIME`

Command.

B. 1 card

Two cases are possible:

1) In a **dynamic analysis with** `COMEBACK`, a single card is used because the program adjusts itself the time steps during calculation.

timestep, *uptime*, *timestepmax*

- *timestep*

  Initial time step in seconds.
- *uptime*

  Time for end of the calculation.
- *timestepmax*

  Maximum value of the time step.

```
     TIME
        2.      3600.        16.
 ENDTIME
```

2) In **other cases** several lines can be given (maximum of *idimtimestep* lines, = 100 in SAFIR2022).

One line added for each time frame added.

timestep, *uptime*

- *timestep*

  Time step in seconds.
- *uptime*

Limit of validity of this time step.

```
    TIME
      2.        20.
     16.      1620.
      8.      3600.
ENDTIME
```

C. <u>1 card, time last line</u>

One line, end of time discretization series.

`ENDTIME`

Command.

SERIES 28: THERMAL ELONGATION

<u>1 card</u>

`COMMAND`

- `COMMAND = NOEPSTH`

  Thermal elongation is 0 for all materials of the structure.

- `COMMAND = EPSTH`

  Thermal elongation is considered in all materials of the structure.

SERIES 29: OUTPUT RESULTS

A. <u>1 card</u>

`OUTPUT`

Command.

B. <u>1 card</u>

`TIMEPRINT`

Command.

<u>1 card</u>

One line added for each timeprint frame added (maximum of *idimtimeprint* lines)

timeprint, uptimeprint

- *timeprint*

  Time step for the output of the results.

- *uptimeprint*

  Limit of validity of this timeprint.

1 card, timeprint last line

`END_TIMEPR`

Command.

C. Multiple cards (can be 0), output optional results

Add one line for each option chosen.

`PRINTREACT`

The reactions are written for at every node where at least one degree of freedom is restrained (by a `BLOCK` or a `SAME` command). The sum of the reactions of all nodes is also written for each degree of freedom. It allows verifying the total applied load (except when master-slave relationships are used for the supports, in which case the results may be confusing because some reactions are counted several times).

`PRINTMN`

Print the internal forces of

1) The BEAM elements.
2) The SOIL elements
3) The TRUSS elements

`PRINTDEPL`, *tstart*

- `PRINTDEPL`

  The increments of displacements are written at every iteration.
- *tstart*

  Time from which the increments will be written.

`PRINTFHE`, *tstart*

- `PRINTFHE`

  The out of balance forces are written at every iteration.
- *tstart*

  Time from which the out of balance forces will be written.

`PRINTTMPRT`

The temperatures are written:

- ✓ in the fibres of the BEAM elements,
- ✓ in the TRUSS finite elements,
- ✓ in the integration points on the thickness and in the bars of the SHELL elements and

✓ at the nodes of the SOLID elements.

`PRINTVELAC`

The velocity and acceleration are written at every time step (In a dynamic analysis).

`PRNSIGMASL`, *nsol*

- `PRNSIGMASL`

  Print the stresses in the SOLID elements.

- *nsol*

  Number of the SOLID element where the mechanical strains, the stresses, the damage in tension and in compression and the accumulated plastic strain will be printed. Valid only for a 3D structural analysis.

  Note: If *nsol* = 0, then the stresses are printed for all SOLID elements. This produces a large amount of output.

`PRNEIBEAM`

Print the stiffness EA, ES and EI in the BEAM elements

`PRNSIGMABM`, *nbm*, *ng*

- `PRNSIGMABM`

  Print the stresses in a BEAM element (positive in tension).

- *nbm*

  Number of the BEAM element where stresses are printed.

- *ng*

  Integration point of the BEAM element where stresses are printed.

`PRINTET`, *nbm*, *ng*

- `PRINTET`

  Print the tangent moduli in a BEAM element.

- *nbm*

  Number of BEAM element where moduli are printed.

- *ng*

  Integration point of the BEAM element where moduli are printed.

`PRNEPSMBM`, *nbm*, *ng*

- `PRNEPSMBM`

Print the mechanical strains in a BEAM element (positive in tension).

- *nbm*

  Number of BEAM element where mechanical strains are printed.

- *ng*

  Integration point of the BEAM element where mechanical strains are printed.

`PRNSIGMASH`, *nsh*

- `PRNSIGMASH`

  Print the stresses in a SHELL element.

- *nsh*

  Number of the SHELL element where the stresses are printed.

`PRINTSHELL`

Equivalent to `PRNSIGMASH` for **all** the SHELL elements (large amount of results).

`PRNNXSHELL`

Print the membrane forces N_x, N_y and N_{xy}, N_1, N_2 and α in the SHELL elements.

`PRNMXSHELL`

Print the bending moments M_x, M_y and M_{xy}, M_1, M_2 and α in the SHELL elements

`PRNEASHELL`

Print the membrane stiffness EA_x, EA_y at the 4 integration points on the surface of the SHELL elements (in an elastic element, this stiffness would be $E\frac{t}{1-\nu^2}$ with *E* the Young's modulus, *t* the thickness of the plate and ν the Poisson ratio). If reinforcing bars are present in the element, the stiffness combines the effect of concrete with this of the bars.

`PRNEISHELL`

Print the bending stiffness EI_x, EI_y at the 4 integration points on the surface of the SHELL elements (in an elastic element, this stiffness would be $E\frac{t^3}{12(1-\nu^2)}$).

`PRNSTRAIN`, *eps_lim*

- `PRNSTRAIN`

  Print a message (with the total strain and the stress related strain) when the absolute value of the stress related strain in a bar of a SHELL element exceeds a certain limit. This message can be retrieved in the ".OUT" file by a search on the targets:

  `Total strain :` or `Stress related strain:`

- *eps_lim*

  Limit of the strain that triggers the message.

D. <u>1 card, output results last line</u>

One blank line as last line of series.

3. Structure of the ".TEM" files used with the BEAM F. E.

SERIES 1: COMMENTS

A. Multiple cards (can be 0)
As many lines as required may be introduced by the user at the beginning of the file to comment the content of the file.

B. 1 card
1 blank line to indicate that the comments are finished.

SERIES 2:

`NFIBERBEAM`, *nfiberbeam(ngb)*

- `NFIBERBEAM`
  Command.
- *nfiberbeam*
  Quantity of fibres in this section.

SERIES 3:

`FIBERS`
Command.

SERIES 4:

A. 1 card for the position of the node of the BEAM element in the (y,z) system of coordinates.
`NODELINE`, *Yo, Zo*

- `NODELINE`
  Command.
- *Yo*
  y coordinate (vertical, upward) of the node in the (y,z) system.
- *Zo*
  z coordinate (horizontal, to the right) of the node in the (y,z) system.

B. 1 card for the position of the centre of rotation of the section in the (y,z) system of coordinates.
`YC_ZC`, *Yc, Zc*

- `YC_ZC`
  Command.
- *Yc*

y coordinate of the centre of rotation in the (y,z) system.

- *Zc*

  z coordinate of the centre of rotation in the (y,z) system.

SERIES 5:

A. 1 card, *nfiberbeam* lines

rcoordyzinbeam(1,nfb), *rcoordyzinbeam(2,nfb), fibersection(nfb), Mat(nfb), Sr(nfb)*

- *rcoordyzinbeam(1,nfb)*

  y coordinate of this fibre.

- *rcoordyzinbeam(2,nfb)*

  z coordinate of this fibre.

- *fibersection(nfb)*

  cross sectional area of this fibre.

- *Mat(nfb)*

  Number of the material in this fibre (the number of the materials is given by the order in which they appear in Series 26 of the .IN file, see Section 2).

- *Sr(nfb)*

  Residual stress present in this fibre.

SERIES 6:

1 card

`COMMAND`

- `COMMAND = COLD`

  This section is not heated. The temperature in all fibres remains at 20°C for all the BEAM finite elements that belong to this type of section. ===> The ".TEM" file can be ended here. No need to write the next groups (`TIME`, `NFIBERBEAM`).

- `COMMAND = HOT`

  This section is heated and all the BEAM finite elements that belong to this type of section will have the same temperature distribution. The next groups are necessary.

- `COMMAND = HASEMI, LOCAFI` or `CFD`

  This section is heated by one or several localised fire and all longitudinal points of integration of all BEAM finite elements that belong to this type of section will have a different temperature distribution, depending on their position in the structure relative of the position of the fire(s).

  The series 1 to 6 are only present in the ".TEM" file of this section type that has the lowest number of all elements and for the point of integration 1.

  For example, if this section type comprises the elements from 11 to 25, Series 1 to 6 are requested only in file "B00011_1.TEM". The files "B00011_2.TEM" until "B00025_2.TEM" contain only Series 7.

SERIES 7: TEMPERATURES

Repeat this Series for each time step when the temperatures are given

A. 1 card

One blank line.

B. 1 card

`TIME=,` *time*

- `TIME=`

  Command.
- *Time*

  Time for which the temperatures are given.

C. 1 card

```
======
```

Command.

D. 1 card, nfiberbeam(ngb) lines

nfb, Tempbeam(nfb)

- *nfb*

  number of the fibre, given in increase order.
- *Tempbeam(nfb)*

  Temperature of the fibre *nfb*.

4. Structure of the “.TOR” files used with the BEAM F. E.

This file is requested for 3D beam finite elements. It gives the torsion stiffness and the warping function of the section. The calculation of these properties must be made by SAFIR on the same section and with the same discretisation as the determination of the temperatures. Therefore:

1) The name of both files must be consistent: if the file for the temperatures is named “*name.tem*”, the file for the torsion properties must be names “*name-t.tor*”.
2) The series 2 to 5 must be the same in both files. SAFIR will make a check on this correspondence and issue an error message if the correspondence is not perfect.

SERIES 1

This Series with the comments is not present in this file

SERIES 2 to 5

See section 3 of this document

SERIES 6

A. 1 card

One blank line.

B. 1 card

```
TIME=
```

Command.

C. 1 card

```
======
```

Command.

D. 1 card

```
W
```

Command.

E. 1 card, *nfiberbeam* lines

w, dw/dy, dw/dz

- *w*
  Value of the warping function in this fibre.
- *dw.dy*
  Spatial derivative of the warping function in this fibre.
- *Dw/dz*
  Spatial derivative of the warping function in this fibre.

SERIES 7

A. 1 card

`According`

Command.

B. 1 card

`GJ=`, *gj*

- `GJ=`
  Command.
- *gj(ngb)*
  Torsional stiffness of the cross section.

5. Structure of the ".TSH" files used with the SHELL F. E.

SERIES 1: COLD OR HOT SECTION

A. 1 card

`COMMAND`

- `COMMAND = COLD`

  This section is not heated. The temperature in the shell remains at 20°C. => The ".TSH" file can be ended here. No need to write the next series of cards.

- `COMMAND = HOT`

  This section is heated by a time-temperature curve (the same curve for all the elements of this type). The next groups are necessary.

- `COMMAND = HASEMI` or `CFD`

  This section is heated by one or several localised fire(s) and all points of integration in the plane of all SHELL finite elements that belong to this type of section will have a different temperature distribution, depending on their position in the structure relative of the position of the fire(s).

  The series 1 to 6 are only present in the ".TSH" file of this section type that has the lowest number of all elements and for the point of integration 1.

  For example, if this section type comprises the elements from 18 to 35, Series 1 to 6 are requested only in file "S00018_1.TSH". The files from "S00018_2.TSH" tol "S00035_4.TSH" contain only Series 7.

  .

SERIES 2: POSITION OF THE NODES

A. 1 card

`POSITION OF THE NODES.`

Command.

B. 1 card

`=======================`

C. 1 card

`NUMBER OF POSITIONS:`, *number_of_positions*

- `NUMBER OF POSITIONS:`

  Command.

- *number_of_positions*

  Gives the quantity of nodes which give the temperature of the slab across its thickness. The positions of these nodes only depend on the discretisation which was chosen when the temperature distribution was calculated. It is independent of the location of the integration points across the thickness which will be used in the structural analysis.

D. 1 card

- position of the first node (the one with the smallest z coordinate)
- position of the second node

…

- position of the last node (the one with the highest z coordinate)

  All positions are given, from node 1 to node *number_of_positions*, 8 values on each line.

SERIES 4: TEMPERATURE

Repeat this group of lines for each time step.

A. 1 card

1 blank line

B. 1 card

`TIME=`, *time*

- `TIME=`

  Command.

- *time*

  Value of the time when the temperatures are given.

C. 1 card

`=====`

D. *number_of_position* lines

- position of the node (same as in series 6)
- temperature at this node

6. Structure of the temperature files used with the TRUSS F.E.

As many lines as necessary, each line being, in a free format, a pair of values in the form:
time   *temperature*
Example :

```
         0.       20.
       300.      600.
       600.      800.
      1200.     1000.
      1500.      900.
      1800.       20.
      3600.       20.
```

7. SHELL elements

This section explains how to consider residual stresses in the first material of the SHELL element. This procedure has been foreseen to introduce residual stresses in steel plates, not in the rebars of reinforced concrete slabs.

To introduce residual stresses, it is necessary to create a file with the name "*resi_str_shell.txt*", which must be placed in the same folder as the structural input file.

In this file, the residual stresses should be placed in Pascal, for each point of integration of the elements in which there are residual stresses.

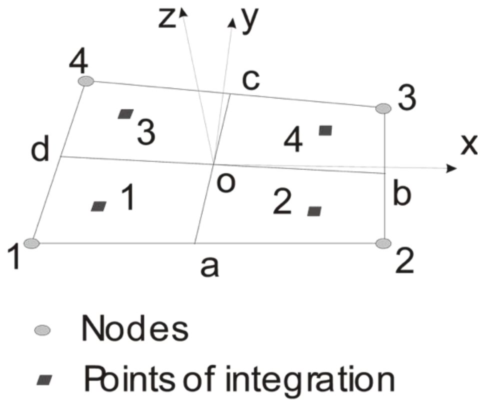


Figure 6: Positions of the integration points in the plane

There are 4 points of integration in the plane of the element located as indicated by Figure 6. The integration in the plane is by the method of Gauss.
The quantity of integration points on the thickness `NGTHICK` is chosen by the user, from 2 to 9. The integration on the thickness is also by the method of Gauss.

Structure of the file “*resi_str_shell.txt*”

One line for SHELL element

`ELEM`, *ne*

- `ELEM`

  Command.
- *ne*

  Number of this element.

One line for each gauss point of the SHELL element. There are (4 x `NGTHICK`) Gauss points.

`PG`, *npg, Sx, Sy, Sxy*

- `PG`

  Command.
- *npg*

  Number of this gauss point.
- *Sx*

  Residual strength in **x** axis.
- *Sy*

  Residual strength in **y** axis.
- *Sxy*

  Residual tangential strength.

One line (optional) for automatic generation of the gauss points.

`Gpg`, *nlpg, kgene*

- `Gpg`

  Command.
- *nlpg*

  Number of the last gauss point with automatic generation.
- *kgene*

  Step for the automatic generation of the gauss points.

One line (optional) for automatic generation of the elements

`GELEM`, *nle, kgene*

- `GELEM`

  Command.
- *nle*

  Number of the last element with automatic generation.
- *kgene*

  Step for the automatic generation of the elements.

Example

```
       ELEM    1
         PG    1    100000    0    0
         PG    2    100000    0    0
         PG    3    100000    0    0
         PG    4    100000    0    0
         PG    5    100000    0    0
         PG    6    100000    0    0
         PG    7    100000    0    0
         PG    8    100000    0    0
       ELEM    2
         PG    1   -100000    0    0
         PG    2   -100000    0    0
         PG    3   -100000    0    0
         PG    4   -100000    0    0
         PG    5   -100000    0    0
         PG    6   -100000    0    0
         PG    7   -100000    0    0
         PG    8   -100000    0    0
       ELEM    3
         PG    1   -100000    0    0
         PG    2   -100000    0    0
         PG    3   -100000    0    0
         PG    4   -100000    0    0
         PG    5   -100000    0    0
         PG    6   -100000    0    0
         PG    7   -100000    0    0
         PG    8   -100000    0    0
       ELEM    4
         PG    1    100000    0    0
         PG    2    100000    0    0
         PG    3    100000    0    0
         PG    4    100000    0    0
         PG    5    100000    0    0
         PG    6    100000    0    0
         PG    7    100000    0    0
         PG    8    100000    0    0
```

Or with automatic generation:

```
       ELEM    1
         PG    1    100000    0    0
        Gpg    8         1
      GELEM    4         3
       ELEM    2
         PG    1   -100000    0    0
        Gpg    8         1
      GELEM    3         1
```