

CADFEM Consulting

User Programmable Feature Programming in ANSYS®

Implementation of New Elements in ANSYS®

Your Contact:

M.Eng. Thomas Iberer
Phone +49 (0) 8092-7005-50
E-Mail tiberer@cadfem.de

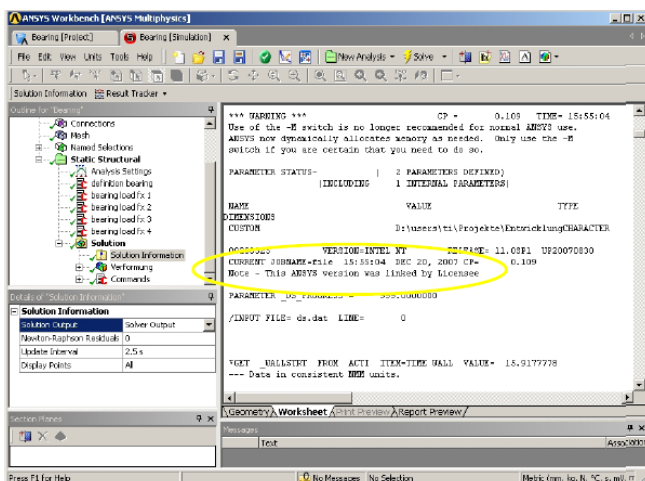
Topic

Sometimes engineers need to have their own element within the ANSYS® Mechanical™ environment. Examples can be found in the bearing industry where long term experience is available in the stiffness description of roller bearings. Sometimes this information is available based on empirical formulas, sometimes the information about the nonlinear stiffness is available just in simple tables. The question is now how to implement this information in ANSYS using the stiffness behavior as an element.

Solution

The User-Programmable-Feature-Interface (UPF) allows us to implement own commands, own material laws and also complete new elements. Within the users element description it is possible to define an individual stiffness, mass and damping matrix that can be also depending on the underlying deformation of the structure. Since this element is fully integrated in the Newton-Raphson scheme for the solution of nonlinear problems it is also possible to implement nonlinear elements in this way.

The code itself has to be provided in the Fortran77 format. The integration of this code is done by compilation of the code fragments and linking it to the remaining ANSYS® libraries. The element can also be used within the ANSYS® Workbench™ environment by means of command snippets.



User defined elements in the Workbench environment

```

nodes (int,ar(mod),in) - array of element node numbers
                        (nmod = number of nodes; 1 in this case)
locsvrl (int,sc,in)    - location of the saved variables
                        on file .esav for this element
kelreq (int,ar(10),in) - matrix and load vector form requests
                        (indices for kelreq are given with output
                        arguments below)
kelfil (int,ar(10),in) - keys indicating incoming matrices and
                        load vectors (indices for kelfil are the
                        same as given for kelreq with output
                        arguments below)
nr (int,sc,in)         - matrix and load vector size
xyz (dp,ar(6,mod),in) - nodal coordinates (orig) and rotation angle
u (dp,ar(nr,5),in)    - element nodal solution values

output arguments:
kelout (int,ar(10),out) - keys indicating created matrices and
                        load vectors (indices for kelout
                        are the same as for kelreq below,
                        as well as kelin and kelout later)
zs (dp,ar(nr,nr),inout) - stiffness/conductivity matrix (kelreq(1))
zass (dp,ar(nr,nr),inout) - mass matrix (kelreq(2))
damp (dp,ar(nr,nr),inout) - damping/specific heat matrix (kelreq(3))
gstif (dp,ar(nr,nr),inout) - stress stiffness matrix (kelreq(4))
zsc (dp,ar(nr),out) - applied f vector (kelreq(5))
zscnr (dp,ar(nr),out) - n-r restoring f vector (kelreq(6))
                        or imaginary f vector (kelreq(7))

```

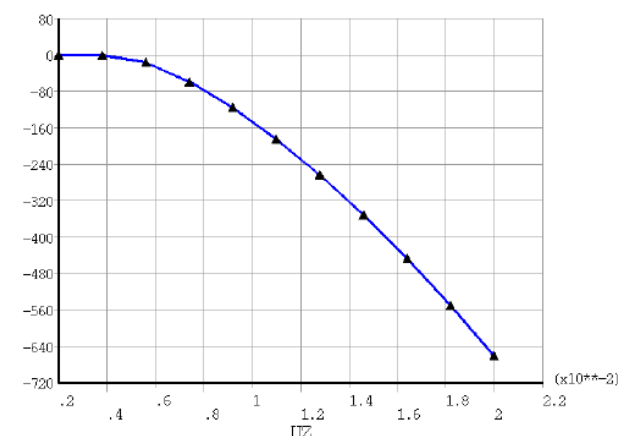
ANSYS®-UPF Documentation for element programming

```

uell01.F
. . . . . 5 . 10 . 15 . 20 . 25 . 30 . 35 . 40 . 45 . 50
412 c --- stiffness matrix
413   if (kelin(1).eq.1) then
414     call vzero (zs(1,1),36)
415     zs(1,1) = et*area/alenv
416     call mctac (zs(1,1),nr,tr(1,1),3, 3,3)
417     do 452 i = 1,3
418       ii = i + 3
419       do 451 j = 1,3
420         jj = j + 3
421         zs(ii,j) = -zs(i,j)
422         zs(i,jj) = -zs(i,j)
423         zs(ii,jj) = zs(i,j)
424       451 continue
425     452 continue
426     kelout(1) = 1
427   endif
428
429 c --- mass matrix
430   if (kelin(2).eq.1) then

```

Fortran code for stiffness matrix of new element



Validation of the nonlinear stiffness behavior