オープンCAE勉強会@富山
第35回　報告

～ユーザーサブルーチンumat～

秋山善克

ABAQUSのユーザーサブルーチン（umat）

```
SUBROUTINE UMAT(STRESS,STATEV,DDSDDE,SSE,SPD,SCD,
    1 RPL,DDSDDT,DRPLDE,DRPLDT,
    2 STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PREDEF,DPRED,CMNAME,
    3 NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
    4 CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
```

決まった変数を受け取り、サブルーチン内を自由に記述可能
子ルーチン、孫ルーチンも作成可能

各サブルーチンの比較

ABAQUS

```
SUBROUTINE UMAT(STRESS,STATEV,DDSDDE,SSE,SPD,SCD,
    1 RPL,DDSDDT,DRPLDE,DRPLDT,
    2 STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PREDEF,DPRED,CMNAME,
    3 NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
    4 CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
```

Code-Aster

```
SUBROUTINE UMAT(STRESS,STATEV,DDSDDE,SSE,SPD,SCD,
   1 RPL,DDSDDT,DRPLDE,DRPLDT,
   2 STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PREDEF,DPRED,CMNAME,
   3 NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
   4 CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
```

CalculiX

umat.f

```
 subroutine umat(stress,statev,ddsdde,sse,spd,scd,
     &  rpl,ddsddt,drplde,drpldt,
     &  stran,dstran,time,dtime,temp,dtemp,predef,dpred,cmname,
     &  ndi,nshr,ntens,nstatv,props,nprops,coords,drot,pnewdt,
     &  celent,dfgrd0,dfgrd1,noel,npt,layer,kspt,kstep,kinc)
```

CalculiXのユーザーサブルーチン

umat.f

```
subroutine umat(stress,statev,ddsdde,sse,spd,scd,
    &  rpl,ddsddt,drplde,drpldt,
    &  stran,dstran,time,dtime,temp,dtemp,predef,dpred,cmname,
    &  ndi,nshr,ntens,nstatv,props,nprops,coords,drot,pnewdt,
    &  celent,dfgrd0,dfgrd1,noel,npt,layer,kspt,kstep,kinc)
```

Umat_user.f

```
subroutine umat_user(amat,iel,iint,kode,elconloc,emec,emec0,
    &        beta,xokl,voj,xkl,vj,ithermal,t1l,dtime,time,ttime,
    &        icmd,ielas,mi,nstate_,xstateini,xstate,stre,stiff,
    &        iorien,pgauss,orab,pnewdt,ipkon)
```

ABAQUSのumatとの違い

```
      !     stran: in CalculiX: Lagrangian strain tensor
      !            in ABAQUS: logarithmic strain tensor
      !     dstran: in CalculiX: Lagrangian strain increment tensor
      !            in ABAQUS: logarithmic strain increment tensor
      !     temp:  in CalculiX: temperature at the end of the increment
      !            in ABAQUS: temperature at the start of the increment
      !     dtemp: in CalculiX: zero
      !            in ABAQUS: temperature increment
```

CalculiXのユーザーサブルーチン（umat_user.f）

input

amat	material name
iel	element number
iint	integration point number
kode	material type (-100-#of constants entered under *USER MATERIAL): can be used for materials with varying number of constants
elconloc(21)	user defined constants defined by the keyword card *USER MATERIAL (max. 21, actual # = -kode-100), interpolated for the actual temperature t1l
emec(6)	Lagrange mechanical strain tensor (component order:11,22,33,12,13,23) at the end of the increment (thermal strains are subtracted)
emec0(6)	Lagrange mechanical strain tensor at the start of the increment (thermal strains are subtracted)
beta(6)	residual stress tensor (the stress entered under the keyword *INITIAL CONDITIONS,TYPE=STRESS)
xokl(3,3)	deformation gradient at the start of the increment
voj	Jacobian at the start of the increment
xkl(3,3)	deformation gradient at the end of the increment
vj	Jacobian at the end of the increment
ithermal	0: no thermal effects are taken into account >0: thermal effects are taken into account (triggered by the keyword *INITIAL CONDITIONS,TYPE=TEMPERATURE)
t1l	temperature at the end of the increment
dtime	time length of the increment
time	step time at the end of the current increment
ttime	total time at the start of the current step
icmd	not equal to 3: calculate stress and stiffness 3: calculate only stress
ielas	0: no elastic iteration: irreversible effects are allowed 1: elastic iteration, i.e. no irreversible deformation allowed
mi(1)	max. # of integration points per element in the model
nstate_	max. # of state variables in the model
xstateini(nstate_,mi(1),# of elements)	state variables at the start of the increment
xstate(nstate_,mi(1),# of elements)	state variables at the end of the increment
stre(6)	Piola-Kirchhoff stress of the second kind at the start of the increment
iorien	number of the local coordinate axis system in the integration point at stake (takes the value 0 if no local system applies)
pgauss(3)	global coordinates of the integration point
orab(7,*)	description of all local coordinate systems. If a local coordinate system applies the global tensors can be obtained by premultiplying the local tensors with skl(3,3). skl is determined by calling the subroutine transformatrix: call transformatrix(orab(1,iorien),pgauss,skl)

```
subroutine umat_user(amat,iel,iint,kode,elconloc,emec,emec0,
     &        beta,xokl,voj,xkl,vj,ithermal,t1l,dtime,time,ttime,
     &        icmd,ielas,mi,nstate_,xstateini,xstate,stre,stiff,
     &        iorien,pgauss,orab,pnewdt,ipkon)
```

output

xstate(nstate_,mi(1),# of elements)	updated state variables at the end of the increment
stre(6)	Piola-Kirchhoff stress of the second kind at the end of the increment
stiff(21)	consistent tangent stiffness matrix in the material frame of reference at the end of the increment. In other words: the derivative of the PK2 stress with respect to the Lagrangian strain tensor. The matrix is supposed to be symmetric, only the upper half is to be given in the same order as for a fully anisotropic elastic material (*ELASTIC,TYPE=ANISO). Notice that the matrix is an integral part of the fourth order material tensor, i.e. the Voigt notation is not used.
pnewdt	to be specified by the user if the material routine is unable to return the stiffness matrix and/or the stress due to divergence within the routine. pnewdt is the factor by which the time increment is to be multiplied in the next trial and should exceed zero but be less than 1. Default is -1 indicating that the user routine has converged.
ipkon(*)	ipkon(iel) points towards the position in field kon prior to the first node of the element's topology. If ipkon(iel) is smaller than 0, the element is not used.

Code-Asterのユーザーサブルーチン

ユーザーサブルーチンの実行

```
$ ln -s umat001a.22 src.f
$ as_run --make_shared -o libumat.so src.f
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:`pwd`
$ as_run umat001a.export
```

ユーザーサブルーチンについてはマニュアルを参照
[U2.10.01] Instructions for the coupling between Code_Aster and ZMAT/UMAT modules

Validation
[V1.03.124] UMAT001 - Test of the Code_Aster Umat interface for linear thermo-elasticity
[V1.03.125] UMAT002 - Test of the Code_Aster Umat interface for linear elasticity under multiaxial loading

CalcliXのインストール

spooles

SourceCode圧縮ファイルを下記からDLする。
http://www.dhondt.de/ の下のthe source code
ccx_2.8p2.src.tar.bz2 をDL
$ bunzip2 ccx_2.8p2.src.tar.bz2 (圧縮解凍)
$ tar -xvf ccx_2.8p2.src.tar (tar ファイル展開)

SPOOLES: http://netlib.sandia.gov/linalg/spooles/ spooles.2.2.tgz
をダウンロード
圧縮ファイルを展開
spooles2.2
Make.inc
15行目
CC = /usr/lang-4.0/bin/cc
CC = /usr/bin/cc

/Tree/src/makeGlobalLib
9行目
drawTree.c
draw.c

spooles2.2ディレクトリで
$make lib
spooles.aができる

$ cd MT
$ make lib
以上でMT/srcにspoolesMT.aができる

参考資料　オープンCAE勉強会@岐阜40回　資料
http://opencae.gifu-nct.ac.jp/pukiwiki/index.php?plugin=attach&pcmd=open&file=OpenCAE2015-06-13-SH_pptx.pdf&refer=%C2%E8%A3%B4%A3%B0%B2%F3%CA%D9%B6%AF%B2%F1%A1%A7H270613

CalcliXのインストール

ARPACK
http://www.caam.rice.edu/software/ARPACK/
からarpack96.tar.gz とpatch.tar.gzをダウンロード
arpack96.tar.gzを展開
patch.tar.gzをその次に展開しarpackに上書き
ARmake.inc
28行目
home pathを書き換える
35行目
PLAT = SUN4
PLAT = INTEL
104行目
FC　　= gfortran
105行目
FFLAGS = -O -cg89
FFLAGS　　= -O2
115行目をコメントアウト
#MAKE　= /bin/make

ARPACK/UTIL/second.f
24行目
EXTERNAL ETIME
を削除

ARPACKで
make lib
libarpack_INTEL.aができる

CalculiX/ccx_2.8p2/src
$ cp Makefile_MT Makefile

Makefileの書き換え
2行目
-I ../../../SPOOLES.2.2
-I /home/akiyama/CaliculiX/spooles.2.2

21行目
-I /home/akiyama/CaliculiX/spooles.2.2
/home/akiyama/CaliculiX/spooles.2.2

26行目
../../../ARPACK/libarpack_INTEL.a ¥
/home/akiyama/CaliculiX/ARPACK/libarpack_INTEL.a ¥