USDFLD/VUSDFLD subroutines

User subroutine to redefine field variables at a material point

(V)USDFLD subroutine

USDFLD: **User subroutine to redefine field variables at a material point.**

VUSDFLD: **User subroutine to redefine field variables at a material point.**

(V)USDFLD subroutine

From the ABAQUS documentation:

ABAQUS/standard

Overview

User subroutine **`USDFLD`**:

- allows you to define field variables at a material point as functions of time or of any of the available material point quantities listed in the Output Variable Identifiers table ("Abaqus/Standard output variable identifiers," Section 4.2.1 of the Abaqus Analysis User's Guide) except the user-defined output variables UVARM and UVARM*n*;
- can be used to introduce solution-dependent material properties since such properties can easily be defined as functions of field variables;
- will be called at all material points of elements for which the material definition includes user-defined field variables;
- must call utility routine **`GETVRM`** to access material point data;
- can use and update state variables; and
- can be used in conjunction with user subroutine **`UFIELD`** to prescribe predefined field variables.

Field variables

ABAQUS/explicit

Overview

User subroutine **`VUSDFLD`**:

- allows the redefinition of field variables at a material point as functions of time or of any of the available material point quantities listed in "Available output variable keys" in "Obtaining material point information in an Abaqus/Explicit analysis," Section 2.1.7;
- can be used to introduce solution-dependent material properties since such properties can be easily defined as functions of field variables;
- will be called at all material points of elements for which the material definition includes user-defined field variables;
- can call utility routine **`VGETVRM`** to access material point data; and
- can use and update solution-dependent state variables.

(V)USDFLD subroutine

From the ABAQUS documentation:

ABAQUS/standard

Overview

User subroutine **`USDFLD`**:

- allows you to define field variables at a material point as functions of time or of any of the available material point quantities listed in the Output Variable Identifiers table ("Abaqus/Standard output variable identifiers," Section 4.2.1 of the Abaqus Analysis User's Guide) except the user-defined output variables UVARM and UVARM*n*;
- can be used to introduce solution-dependent material properties since such properties can easily be defined as functions of field variables;
- will be called at all material points of elements for which the material definition includes user-defined field variables;
- must call utility routine **`GETVRM`** to access material point data;
- can use and update state variables; and
- can be used in conjunction with user subroutine **`UFIELD`** to prescribe predefined field variables.

State dependent variables

ABAQUS/explicit

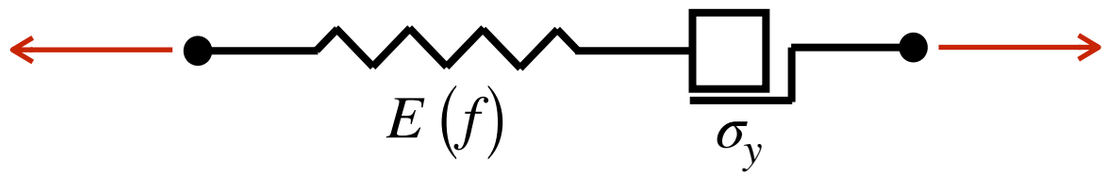
Overview

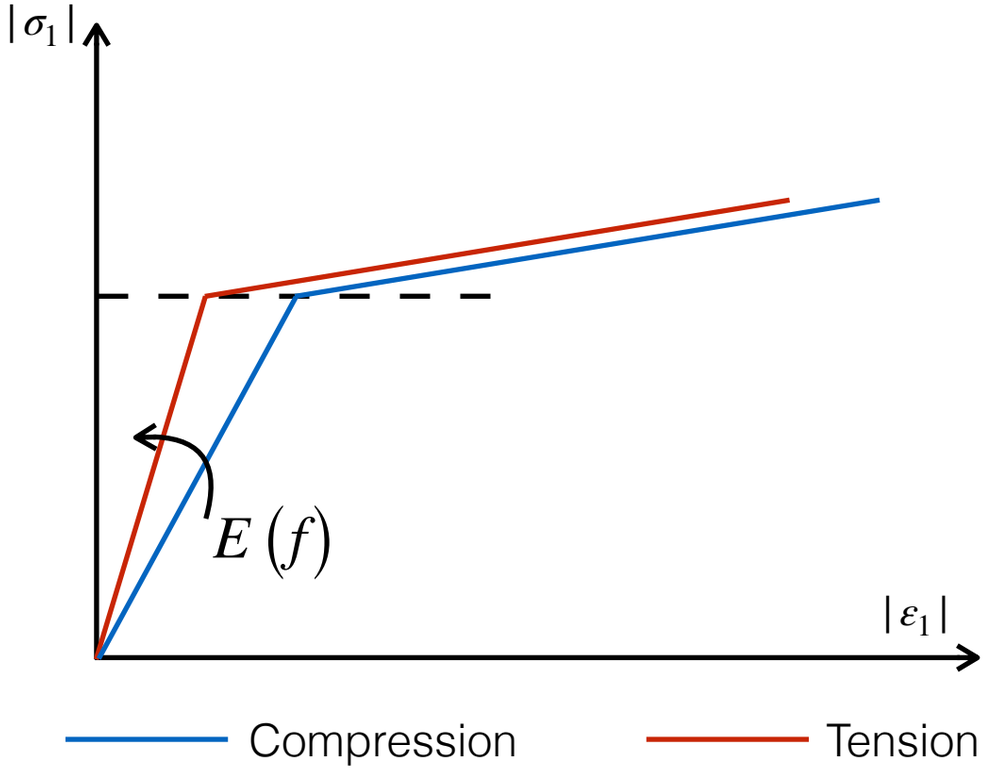
User subroutine **`VUSDFLD`**:

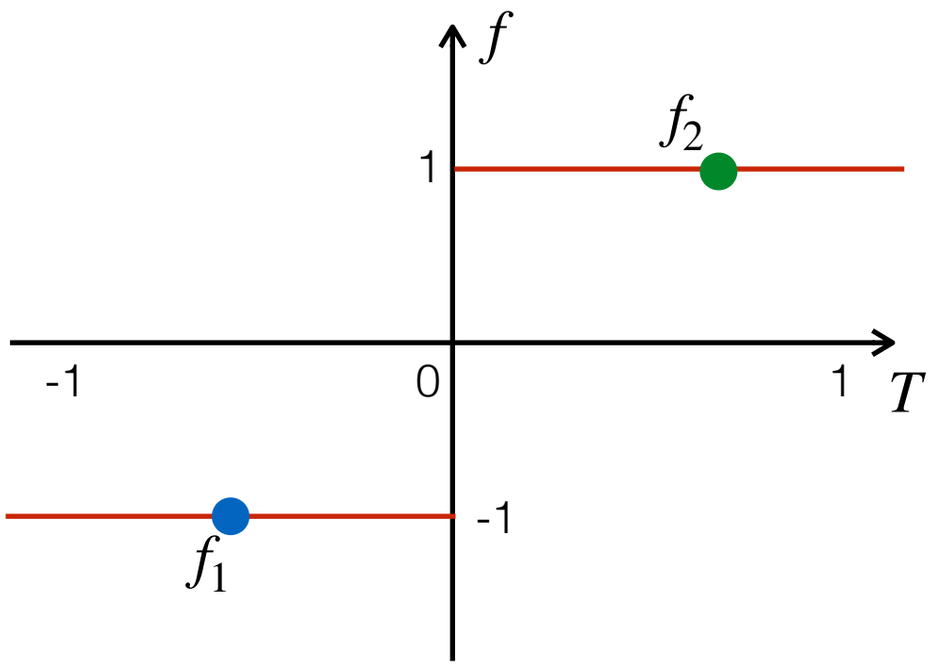
- allows the redefinition of field variables at a material point as functions of time or of any of the available material point quantities listed in "Available output variable keys" in "Obtaining material point information in an Abaqus/Explicit analysis," Section 2.1.7;
- can be used to introduce solution-dependent material properties since such properties can be easily defined as functions of field variables;
- will be called at all material points of elements for which the material definition includes user-defined field variables;
- can call utility routine **`VGETVRM`** to access material point data; and
- can use and update solution-dependent state variables.

(V)USDFLD subroutine

In this example, we want a material model where the Young's modulus E is different between tension and compression.







$$E\left(f_1\right) < E\left(f_2\right)$$

(V)USDFLD subroutine

```
77  **----------------------------------------------------------------
78  ** MATERIALS
79  **----------------------------------------------------------------
80  *material,name=EXAMPLE_VUSDLFD_V1
81  *density
82  7.8e-9
83  *elastic,dependencies=1
84  **     E,  NU,  TEMP,     F
85   21000.0, 0.3,   0.0, -1.0
86  210000.0, 0.3,   0.0,  1.0
87  *plastic
88  250.0, 0.0
89  350.0, 1.0
90  *User defined field
91  **----------------------------------------------------------------
```

Tabulated data with the field variable f
Temperature must always be present

Call to the VUSDFLD subroutine

(V)USDFLD subroutine

Same input between ABAQUS/standard and ABAQUS/explicit:

ABAQUS/explicit

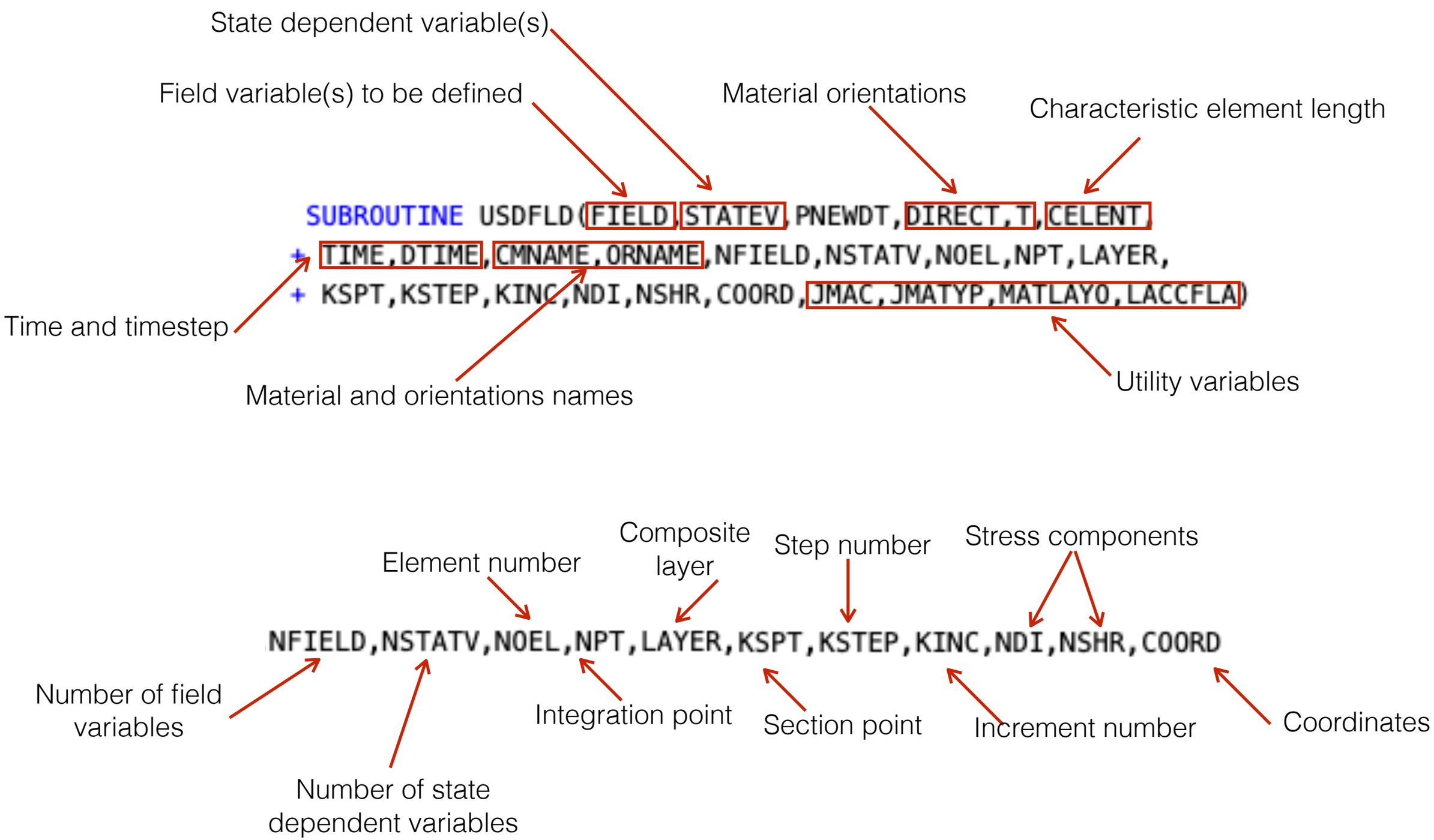
```
77  **-----------------------------------------------------------------
78  ** MATERIALS
79  **-----------------------------------------------------------------
80  *material,name=EXAMPLE_VUSDLFD_V1
81  *density
82  7.8e-9
83  *elastic,dependencies=1
84  **     E,  NU, TEMP,    F
85   21000.0, 0.3,  0.0, -1.0
86  210000.0, 0.3,  0.0,  1.0
87  *plastic
88  250.0, 0.0
89  350.0, 1.0
90  *User defined field
91  **-----------------------------------------------------------------
```

ABAQUS/standard

```
77  **-----------------------------------------------------------------
78  ** MATERIALS
79  **-----------------------------------------------------------------
80  *material,name=EXAMPLE_USDLFD_V1
81  *elastic,dependencies=1
82  **     E,  NU, TEMP,    F
83   21000.0, 0.3,  0.0, -1.0
84  210000.0, 0.3,  0.0,  1.0
85  *plastic
86  250.0, 0.0
87  350.0, 1.0
88  *User defined field
89  **-----------------------------------------------------------------
```

USDFLD subroutine

Description of the arguments for USDFLD:



USDFLD subroutine

Very similar coding to the UVARM subroutine:

```
      SUBROUTINE USDFLD(FIELD,STATEV,PNEWDT,DIRECT,T,CELENT,
     + TIME,DTIME,CMNAME,ORNAME,NFIELD,NSTATV,NOEL,NPT,LAYER,
     + KSPT,KSTEP,KINC,NDI,NSHR,COORD,JMAC,JMATYP,MATLAYO,LACCFLA)
      INCLUDE 'ABA_PARAM.INC'
!-----------------------------------------------------------------
!-----Declaration ABAQUS variables
!-----------------------------------------------------------------
      CHARACTER*80 CMNAME,ORNAME
      DIMENSION FIELD(NFIELD),STATEV(NSTATV),DIRECT(3,3)
      DIMENSION T(3,3),TIME(2),COORD(*),JMAC(*),JMATYP(*)
!-----Data from ABAQUS
      DIMENSION ARRAY(15),JARRAY(15)
      CHARACTER*3 FLGRAY(15)
!-----------------------------------------------------------------
!-----Declaration internal variables
!-----------------------------------------------------------------
      real*8 SIGH,SMISES,TRIAX
!-----------------------------------------------------------------
!     Access stress invariants
!-----------------------------------------------------------------
      CALL GETVRM('SINV',ARRAY,JARRAY,FLGRAY,JRCD,
     +            JMAC,JMATYP,MATLAYO,LACCFLA)
c
      SIGH   = ARRAY(3)
      SMISES = ARRAY(1)
```

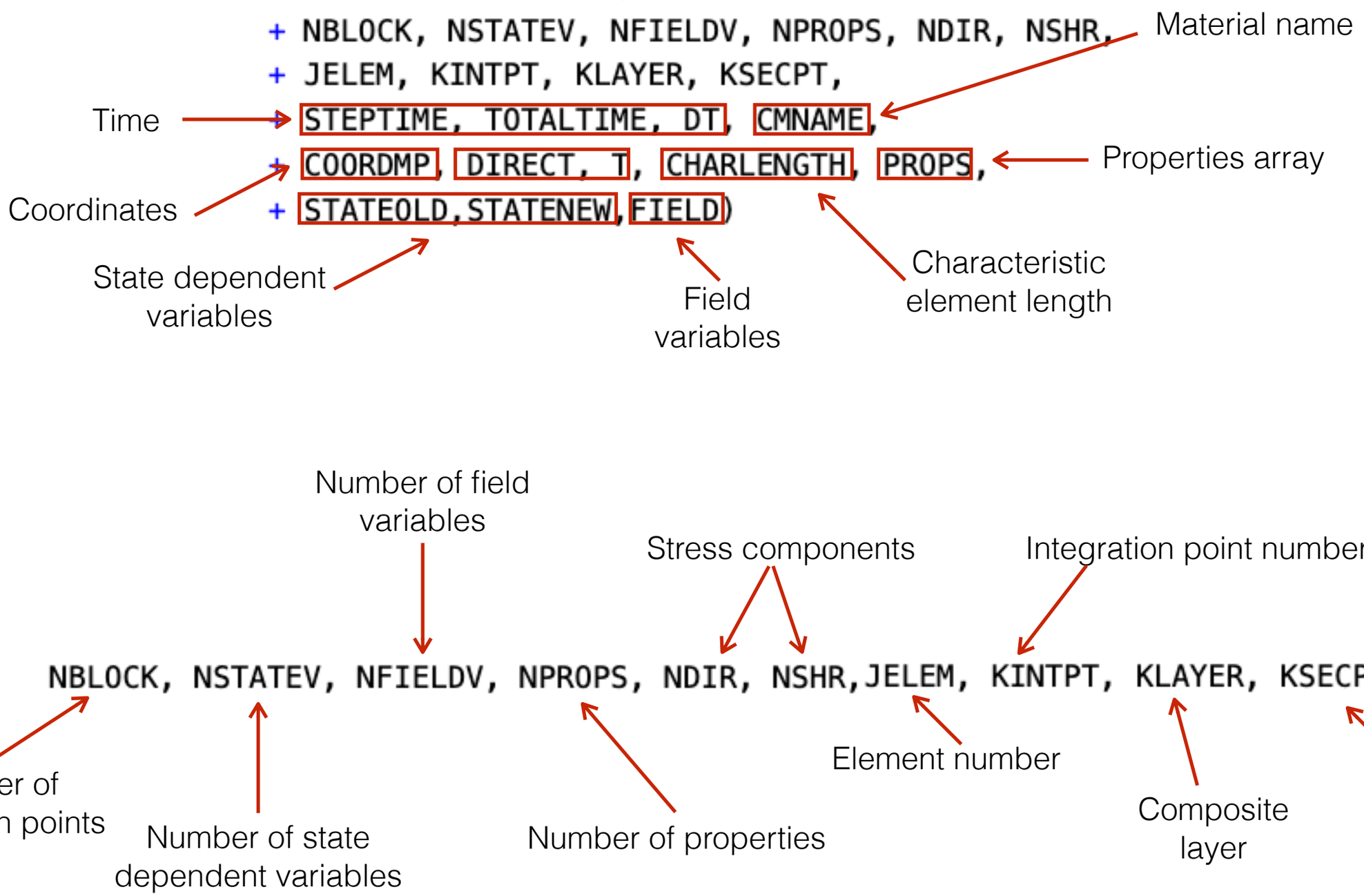
Access to ABAQUS variables

```
!-----------------------------------------------------------------
!     Compute the stress triaxiality
!-----------------------------------------------------------------
      if(SMISES.gt.0.0)then
         TRIAX = -SIGH/SMISES
      else
         TRIAX = 0.0
      endif
!-----------------------------------------------------------------
!     Update field variable
!-----------------------------------------------------------------
      if(TRIAX.ge.0.0)then
         FIELD(1) = 1.05
      else
         FIELD(1) =-1.05
      endif
!-----------------------------------------------------------------
!     End of subroutine
!-----------------------------------------------------------------
      RETURN
      END
```

Field variable f

VUSDFLD subroutine

Description of the arguments for VUSDFLD:



VUSDFLD subroutine

```
      subroutine VUSDFLD(
     + NBLOCK, NSTATEV, NFIELDV, NPROPS, NDIR, NSHR,
     + JELEM, KINTPT, KLAYER, KSECPT,
     + STEPTIME, TOTALTIME, DT, CMNAME,
     + COORDMP, DIRECT, T, CHARLENGTH, PROPS,
     + STATEOLD,STATENEW,FIELD)
      include 'vaba_param.inc'
!----------------------------------------------------------------------
!-----Declaration ABAQUS variables
!----------------------------------------------------------------------
      dimension JELEM(nblock),COORDMP(nblock,*),DIRECT(nblock,3,3),
     .          T(nblock,3,3),CHARLENGTH(nblock),PROPS(nprops),
     .          STATEOLD(nblock,nstatev),STATENEW(nblock,nstatev),
     .          FIELD(nblock,nfieldv)
      character*80 cmname
!-----Data from ABAQUS
      dimension stressdata(maxblk*(ndir+nshr))
      integer jSData(maxblk*(ndir+nshr))
      character*3 cSData(maxblk*(ndir+nshr))
      integer jStatus
!----------------------------------------------------------------------
!-----Declaration internal variables
!----------------------------------------------------------------------
      integer i
      real*8 s(nblock,6)
      real*8 SMISES(nblock),SIGH(nblock),TRIAX(nblock)
!----------------------------------------------------------------------
!     Access stress tensor
!----------------------------------------------------------------------
      call vgetvrm(  'S' , stressdata,jSData,cSData,jStatus)
!----------------------------------------------------------------------
```

Access to ABAQUS variables

```
!----------------------------------------------------------------------
!     Extract data from stressdata
!----------------------------------------------------------------------
      if(nshr.gt.1)then
         do i=1,nblock
            s(i,1) = stressdata(i)
            s(i,2) = stressdata(i+nblock)
            s(i,3) = stressdata(i+nblock*2)
            s(i,4) = stressdata(i+nblock*3)
            s(i,5) = stressdata(i+nblock*4)
            s(i,6) = stressdata(i+nblock*5)
         enddo
      else
         do i=1,nblock
            s(i,1) = stressdata(i)
            s(i,2) = stressdata(i+nblock)
            s(i,3) = stressdata(i+nblock*2)
            s(i,4) = stressdata(i+nblock*3)
         enddo
      endif
```

Accessing ABAQUS variables

Utility routine VGETVRM:

Maximum block size (given)

```
dimension rdata(maxblk*(nvar))
integer jData(maxblk*(nvar))
character*3 cData(maxblk*(nvar))
integer jStatus
```

To be defined

```
call vgetvrm('VAR', rdata,jData,cData, jStatus)
```

Output variable key

Array with values

Integer array with components

Array with flags

Error code

- In ABAQUS explicit, there is a limited number of variables which can be accessed but the vgetvrm utility subroutine
- The list is given in the documentation of the subroutine

Accessing ABAQUS variables

Utility routine VGETVRM:

Tensor are not stored as matrix but as a long vector:

```
!-----Data from ABAQUS
      dimension stressdata(maxblk*(ndir+nshr))
      integer jSData(maxblk*(ndir+nshr))
      character*3 cSData(maxblk*(ndir+nshr))
      integer jStatus
```

```
!-------------------------------------------------------------
!     Access stress tensor
!-------------------------------------------------------------
      call vgetvrm(  'S' , stressdata,jSData,cSData,jStatus)
!-------------------------------------------------------------
!     Extract data from stressdata
!-------------------------------------------------------------
      if(nshr.gt.1)then
         do i=1,nblock
            s(i,1) = stressdata(i)
            s(i,2) = stressdata(i+nblock)
            s(i,3) = stressdata(i+nblock*2)
            s(i,4) = stressdata(i+nblock*3)
            s(i,5) = stressdata(i+nblock*4)
            s(i,6) = stressdata(i+nblock*5)
         enddo
      else
         do i=1,nblock
            s(i,1) = stressdata(i)
            s(i,2) = stressdata(i+nblock)
            s(i,3) = stressdata(i+nblock*2)
            s(i,4) = stressdata(i+nblock*3)
         enddo
      endif
```

VUSDFLD subroutine

Access to ABAQUS variables

```
26  !-----------------------------------------------------------------
27  !-----Declaration internal variables
28  !-----------------------------------------------------------------
29        integer i
30        real*8 s(nblock,6)
31        real*8 SMISES(nblock),SIGH(nblock),TRIAX(nblock)
32  !-----------------------------------------------------------------
33  !     Access stress tensor
34  !-----------------------------------------------------------------
35        call vgetvrm(  'S' , stressdata,jSData,cSData,jStatus)
36  !-----------------------------------------------------------------
37  !     Extract data from stressdata
38  !-----------------------------------------------------------------
39        if(nshr.gt.1)then
40           do i=1,nblock
41              s(i,1) = stressdata(i)
42              s(i,2) = stressdata(i+nblock)
43              s(i,3) = stressdata(i+nblock*2)
44              s(i,4) = stressdata(i+nblock*3)
45              s(i,5) = stressdata(i+nblock*4)
46              s(i,6) = stressdata(i+nblock*5)
47           enddo
48        else
49           do i=1,nblock
50              s(i,1) = stressdata(i)
51              s(i,2) = stressdata(i+nblock)
52              s(i,3) = stressdata(i+nblock*2)
53              s(i,4) = stressdata(i+nblock*3)
54           enddo
55        endif
```

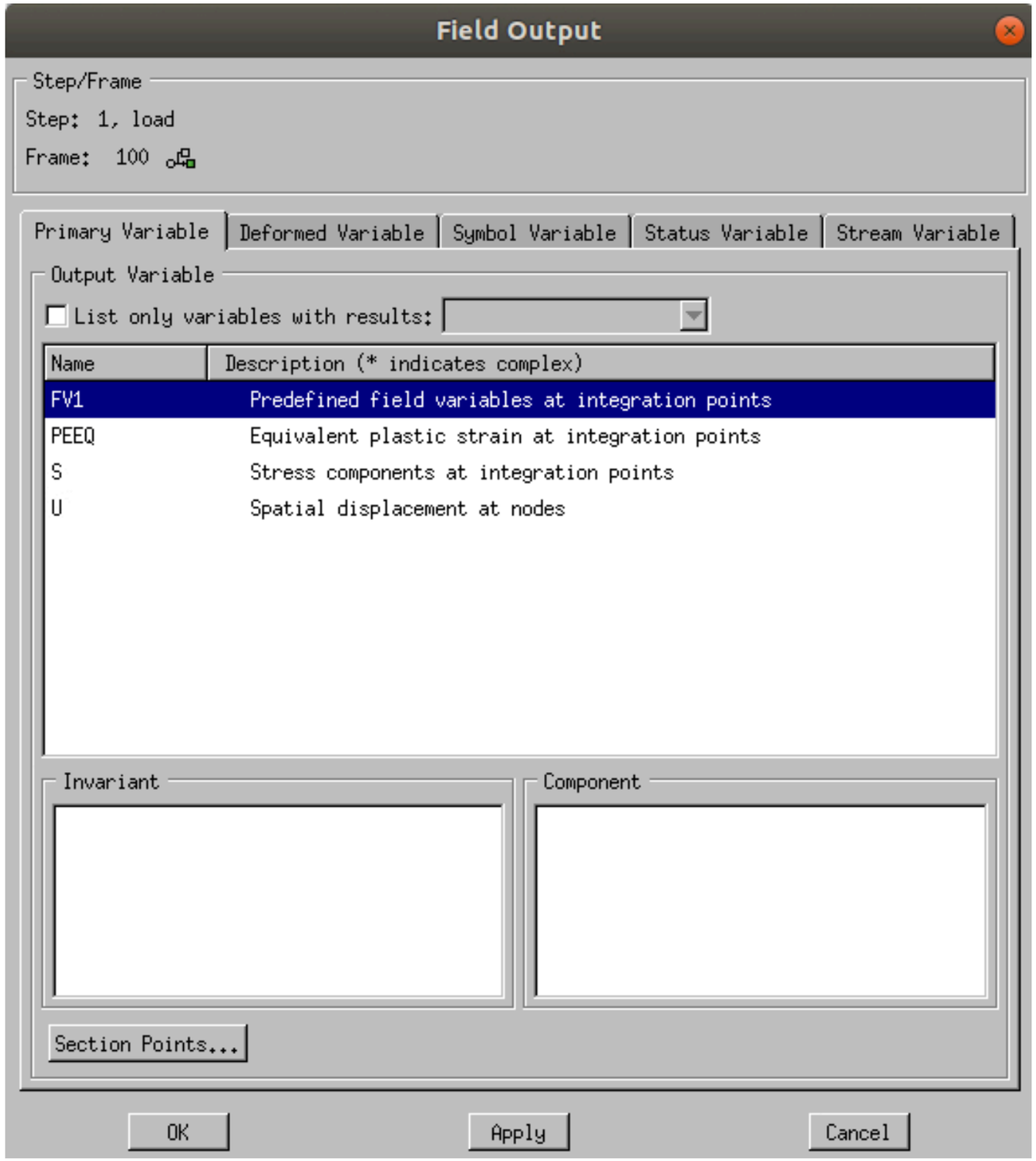
```
56  !-----------------------------------------------------------------
57  !     Compute von Mises equivalent stress and hydrostatic stress
58  !-----------------------------------------------------------------
59        if(nshr.gt.1)then
60           do i=1,nblock
61              SIGH(i)   = (s(i,1)+s(i,2)+s(i,3))/3.0
62              SMISES(i) = sqrt(s(i,1)*s(i,1)+s(i,2)*s(i,2)
63       +                       +s(i,3)*s(i,3)
64       +                       -s(i,1)*s(i,2)-s(i,2)*s(i,3)
65       +                       -s(i,3)*s(i,1)
66       +                    +3.0*(s(i,4)*s(i,4)+s(i,5)*s(i,5)
67       +                       +s(i,6)*s(i,6)))
68           enddo
69        else
70           do i=1,nblock
71              SIGH(i)   = (s(i,1)+s(i,2)+s(i,3))/3.0
72              SMISES(i) = sqrt(s(i,1)*s(i,1)+s(i,2)*s(i,2)
73       +                       -s(i,1)*s(i,2)+3.0*s(i,4)*s(i,4))
74           enddo
75        endif
76  !-----------------------------------------------------------------
77  !     Compute stress triaxiality
78  !-----------------------------------------------------------------
79        do i=1,nblock
80           if(SMISES(i).gt.0.0)then
81              TRIAX(i) = SIGH(i)/SMISES(i)
82           else
83              TRIAX(i) = 0.0
84           endif
85        enddo
86  !-----------------------------------------------------------------
87  !     Update field variable
88  !-----------------------------------------------------------------
89        do i=1,nblock
90           if(TRIAX(i).ge.0.0)then
91              field(i,1) = 1.05
92           else
93              field(i,1) =-1.05
94           endif
95        enddo
96  !-----------------------------------------------------------------
97  !     End of subroutine
98  !-----------------------------------------------------------------
99        return
100       end
```

Field variable f

(V)USDFLD subroutine

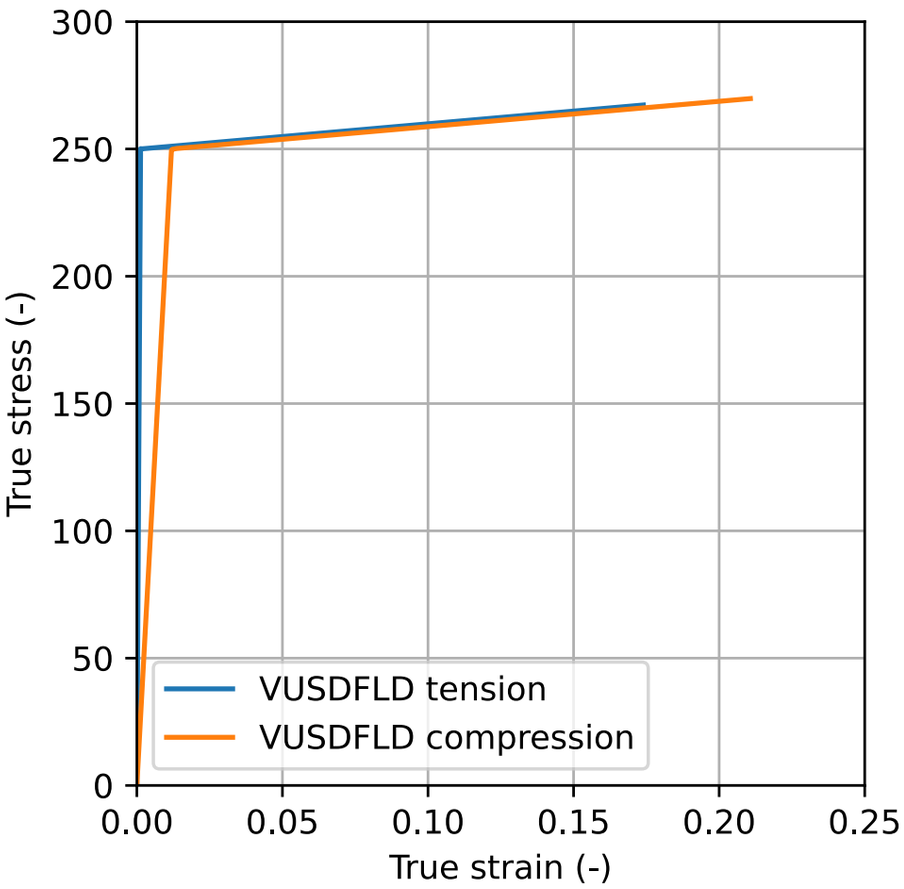
Odb file structure:

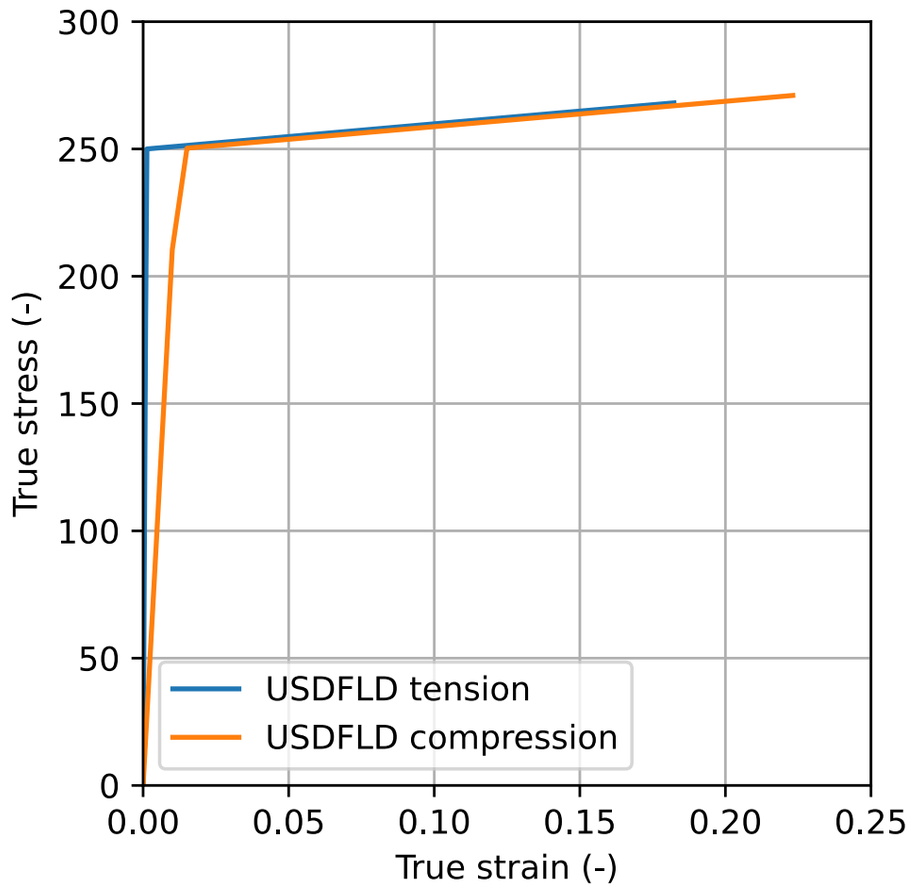
Field variable f



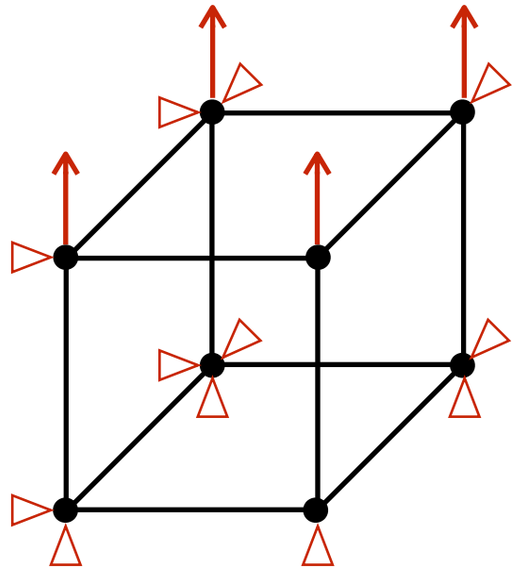
(V)USDFLD subroutine

Results from the USDFLD_V1 and VUSDFLD_V1 subroutines

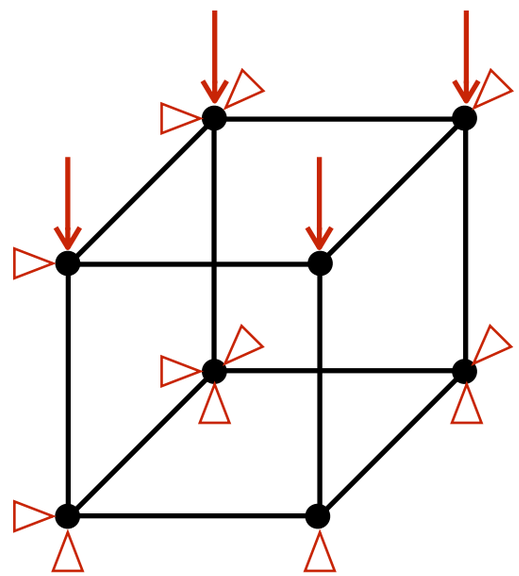




Single solid element
in uniaxial tension



Single solid element in
uniaxial compression



(V)USDFLD subroutine

ABAQUS/standard

Overview

User subroutine **`USDFLD`**:

- allows you to define field variables at a material point as functions of time or of any of the available material point quantities listed in the Output Variable Identifiers table ("Abaqus/Standard output variable identifiers," Section 4.2.1 of the Abaqus Analysis User's Guide) except the user-defined output variables UVARM and UVARM*n*;
- can be used to introduce solution-dependent material properties since such properties can easily be defined as functions of field variables;
- will be called at all material points of elements for which the material definition includes user-defined field variables;
- must call utility routine **`GETVRM`** to access material point data;
- can use and update state variables; and
- can be used in conjunction with user subroutine **`UFIELD`** to prescribe predefined field variables.

State dependent variables

ABAQUS/explicit

Overview

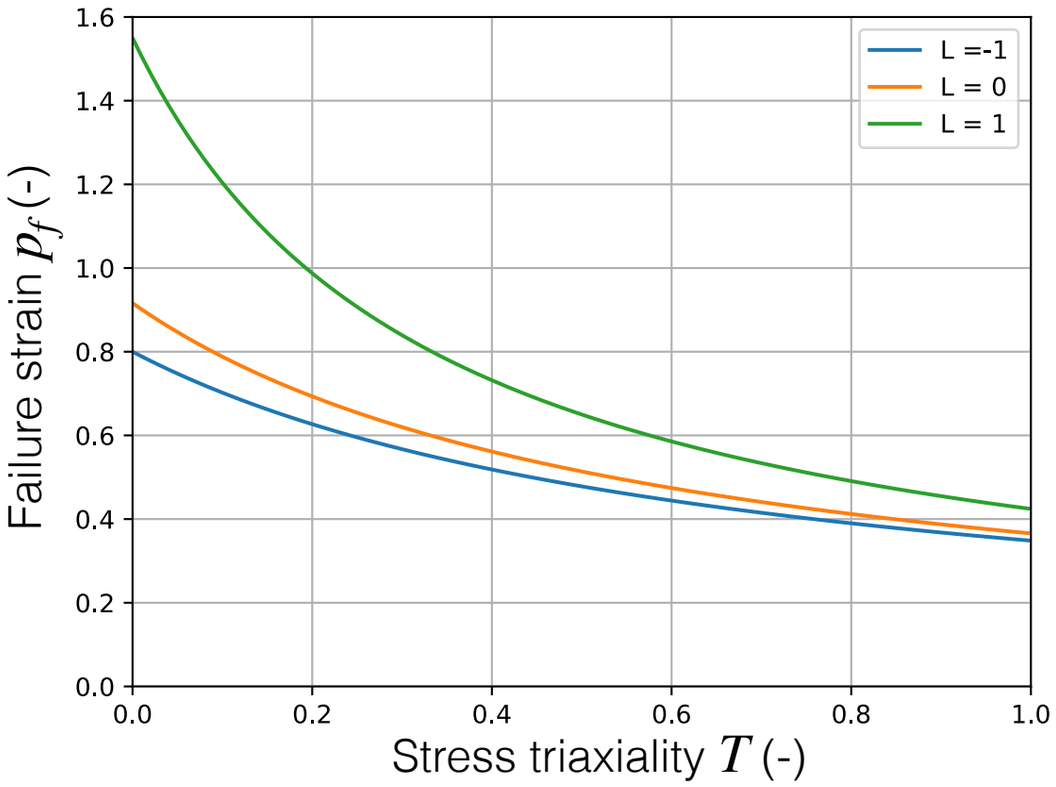
User subroutine **`VUSDFLD`**:

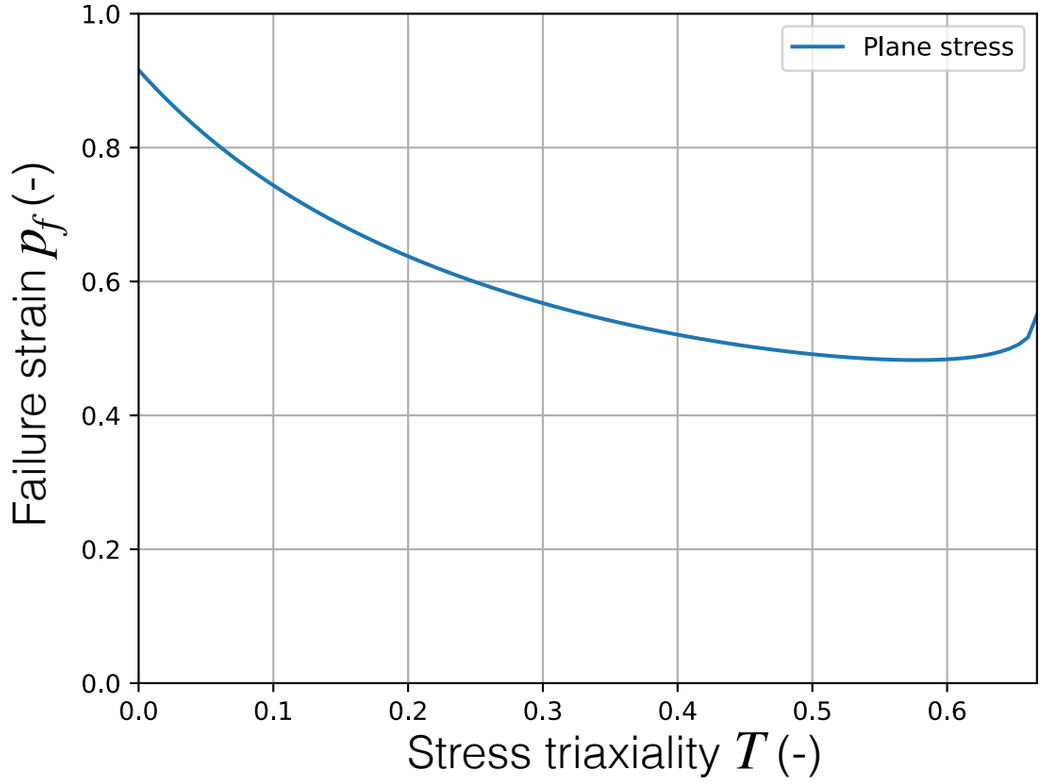
- allows the redefinition of field variables at a material point as functions of time or of any of the available material point quantities listed in "Available output variable keys" in "Obtaining material point information in an Abaqus/Explicit analysis," Section 2.1.7;
- can be used to introduce solution-dependent material properties since such properties can be easily defined as functions of field variables;
- will be called at all material points of elements for which the material definition includes user-defined field variables;
- can call utility routine **`VGETVRM`** to access material point data; and
- can use and update solution-dependent state variables.

(V)USDFLD subroutine

In this example, we want to add a user-defined fracture model to a built-in plasticity model from ABAQUS.

Cockcroft-Latham criterion: $$D = \int_0^{p_f} \frac{\langle \sigma_1 \rangle}{W_c} \dot{p} \leq 1$$





(V)USDFLD subroutine

Cockcroft-Latham criterion:

$$D = \int_0^{p_f} \frac{\langle \sigma_1 \rangle}{W_c} \dot{p} \leq 1$$

Call to the VUSDFLD subroutine

W_c

State dependent variables

```
**---------------------------------
** MATERIALS
**---------------------------------
*material,name=EXAMPLE_VUSDLFD_V2
*density
7.8e-9
*elastic
210000.0, 0.3
*plastic
250.0, 0.0
350.0, 1.0
*User defined field, properties=1
100.0
*Depvar, delete=3
3,
1,PEEQ,'Equivalent plastic strain'
2,D,'Damage variable'
3,STATUS,'Failure status'
**---------------------------------
```

(V)USDFLD subroutine

ABAQUS/standard

```
**---------------------------------
** MATERIALS
**---------------------------------
*material,name=EXAMPLE_VUSDLFD_V2
*elastic
210000.0, 0.3
*plastic
250.0, 0.0
350.0, 1.0
*User defined field
*Depvar, delete=3
3,
1,PEEQ,'Equivalent plastic strain'
2,D,'Damage variable'
3,STATUS,'Failure status'
**---------------------------------
```

ABAQUS/explicit

```
**---------------------------------
** MATERIALS
**---------------------------------
*material,name=EXAMPLE_VUSDLFD_V2
*density
7.8e-9
*elastic
210000.0, 0.3
*plastic
250.0, 0.0
350.0, 1.0
*User defined field, properties=1
100.0
*Depvar, delete=3
3,
1,PEEQ,'Equivalent plastic strain'
2,D,'Damage variable'
3,STATUS,'Failure status'
**---------------------------------
```

Properties can not be given to the ABAQUS/standard USDFLD subroutine

(V)USDFLD subroutine

ABAQUS/standard

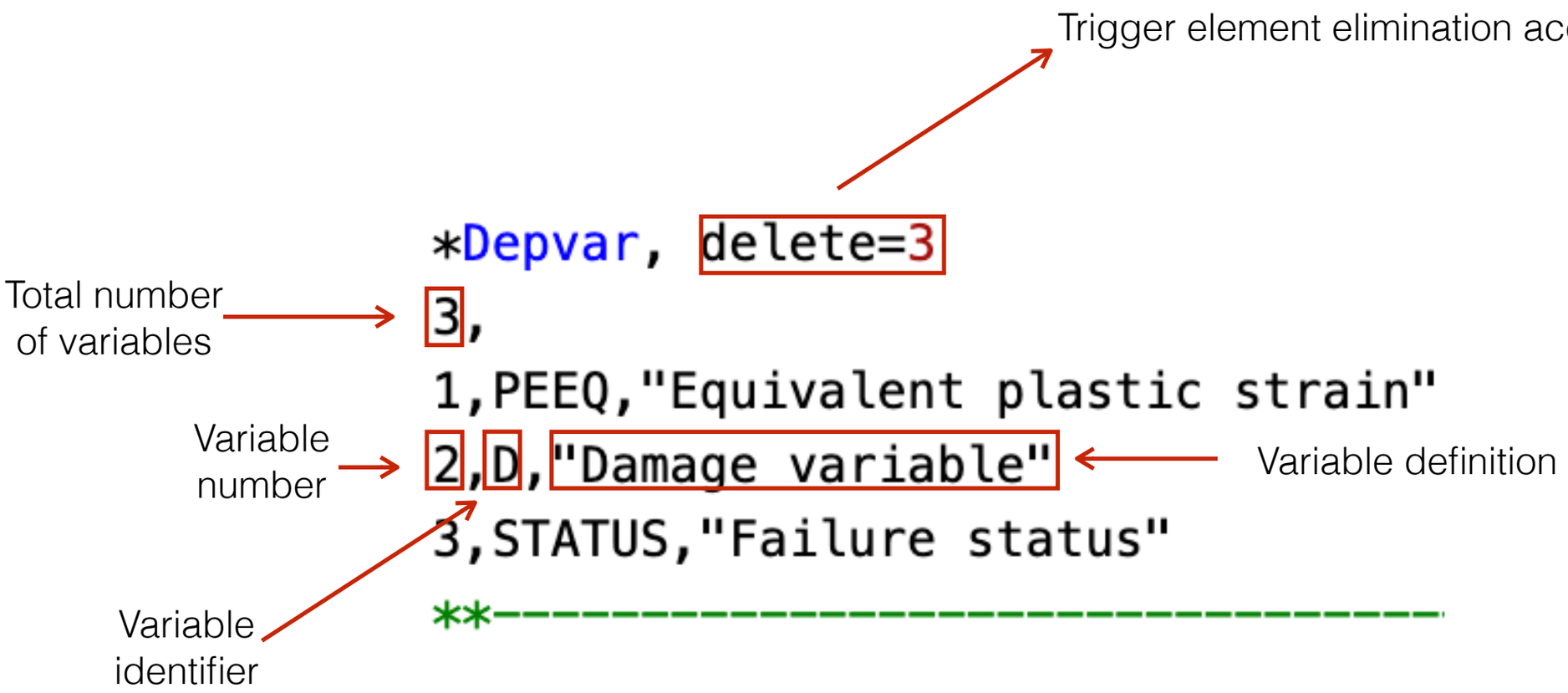
```
**---------------------------------
** MATERIALS
**---------------------------------
*material,name=EXAMPLE_VUSDLFD_V2
*elastic
210000.0, 0.3
*plastic
250.0, 0.0
350.0, 1.0
*User defined field
*Depvar, delete=3
3,
1,PEEQ,'Equivalent plastic strain'
2,D,'Damage variable'
3,STATUS,'Failure status'
**---------------------------------
```

```
**-------------------------------------
** STEP
**-------------------------------------
*Step, name=LOADING, nlgeom=YES, inc=1000
*Static
0.001, 1., 1e-06, 0.01
*FIELD
**-------------------------------------
```

The *FIELD keyword must present in ABAQUS/ Standard to activate the USDFLD subroutine

(V)USDFLD subroutine

State Dependent Variables (SDV):



(V)USDFLD subroutine

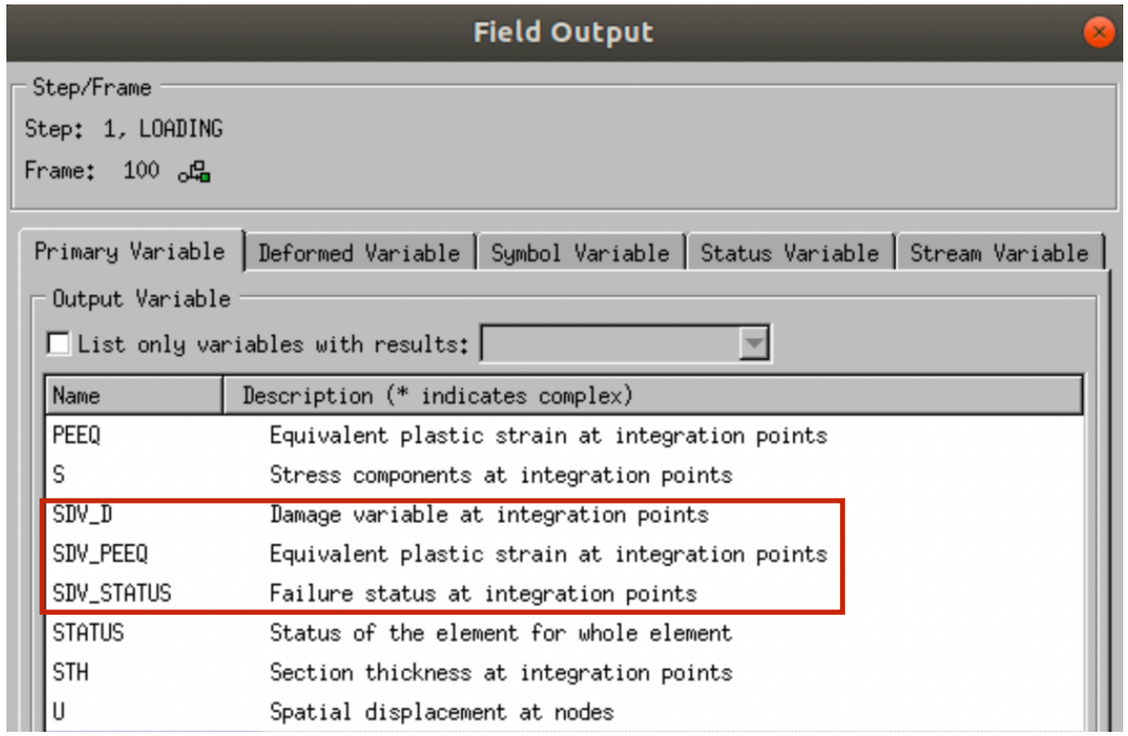
Input file structure:

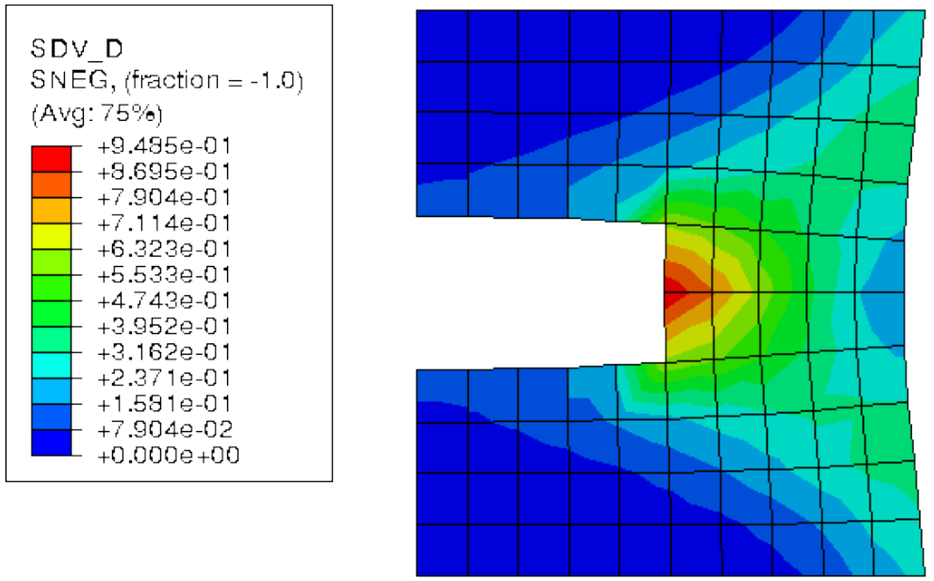
```
**----------------------------------------------
** FIELD OUTPUT
**----------------------------------------------
*Output, field, number interval=100, time marks=YES
*Element Output, directions=YES
PEEQ, S, SDV, STATUS
*Node Output
U
```

State dependent variables

Status for element elimination (from odb file)

Odb file structure:





(V)USDFLD subroutine

ABAQUS/standard

ABAQUS/explicit

Status = 0 means inactive (failed) integration point

```
63    !-----------------------------------
64    !     Check for fracture
65    !-----------------------------------
66          if(damage.ge.1.0)then
67             STATEV(3) = 0.0
68          else
69             STATEV(3) = 1.0
70          endif
71    !-----------------------------------
```

```
101   !-----------------------------------
102   !     Check for fracture
103   !-----------------------------------
104         do i=1,nblock
105            if(damage(i).ge.1.0)then
106               stateNew(i,3) = 0.0
107            else
108               stateNew(i,3) = 1.0
109            endif
110         enddo
```

Status = 1 means active integration point

(V)USDFLD subroutine

Variables at t when USDFLD is called

Variables at $t + \Delta t$ to be updated in USDFLD

ABAQUS/standard

```
      SUBROUTINE USDFLD(FIELD,STATEV,PNEWDT,DIRECT,T,CELENT,
     + TIME,DTIME,CMNAME,ORNAME,NFIELD,NSTATV,NOEL,NPT,LAYER,
     + KSPT,KSTEP,KINC,NDI,NSHR,COORD,JMAC,JMATYP,MATLAYO,LACCFLA)
```

ABAQUS/explicit

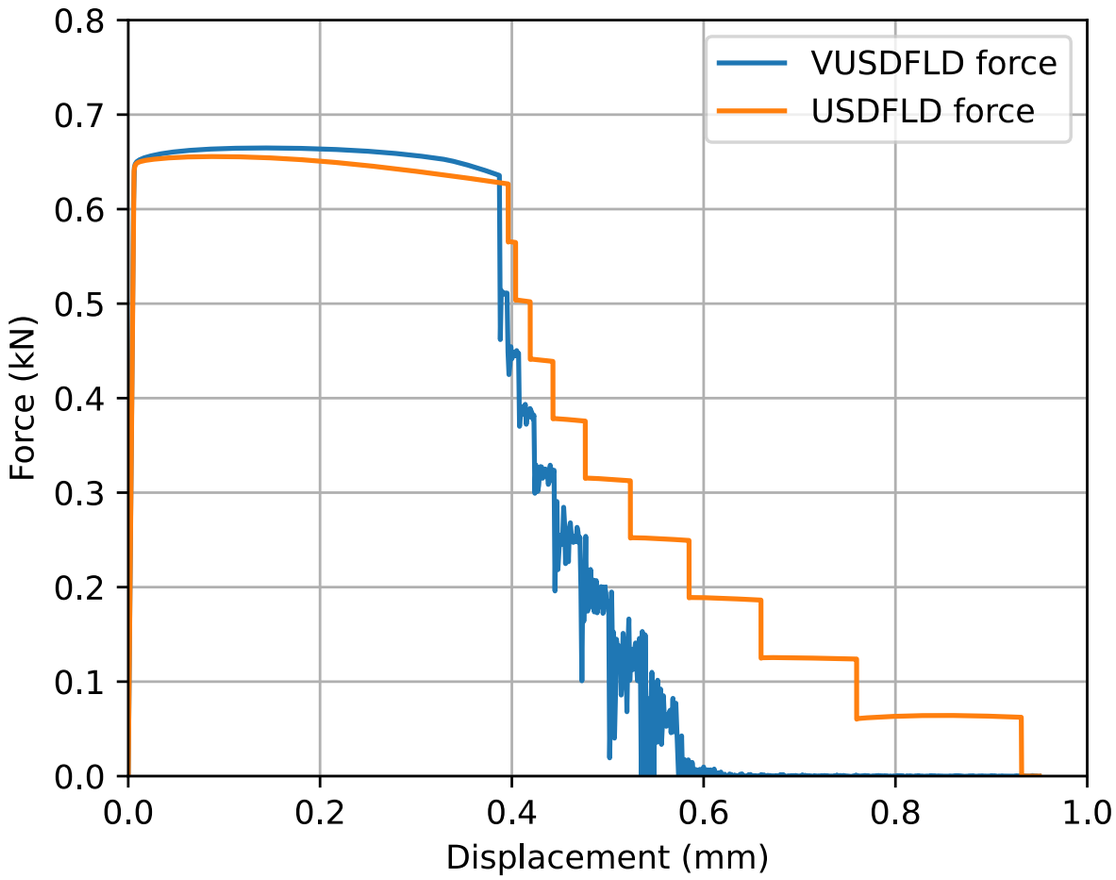
```
      subroutine VUSDFLD(
     + NBLOCK, NSTATEV, NFIELDV, NPROPS, NDIR, NSHR,
     + JELEM, KINTPT, KLAYER, KSECPT,
     + STEPTIME, TOTALTIME, DT, CMNAME,
     + COORDMP, DIRECT, T, CHARLENGTH, PROPS,
     + STATEOLD,STATENEW,FIELD)
```

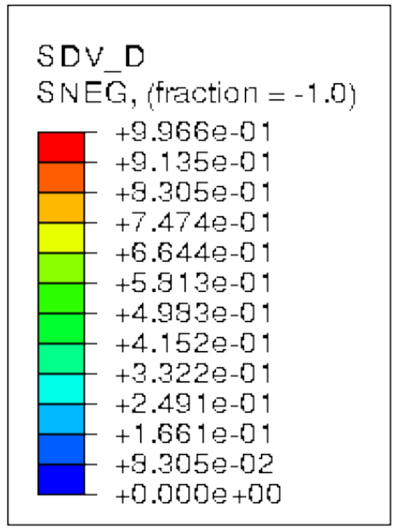
Variables at t

Variables at $t + \Delta t$ to be updated

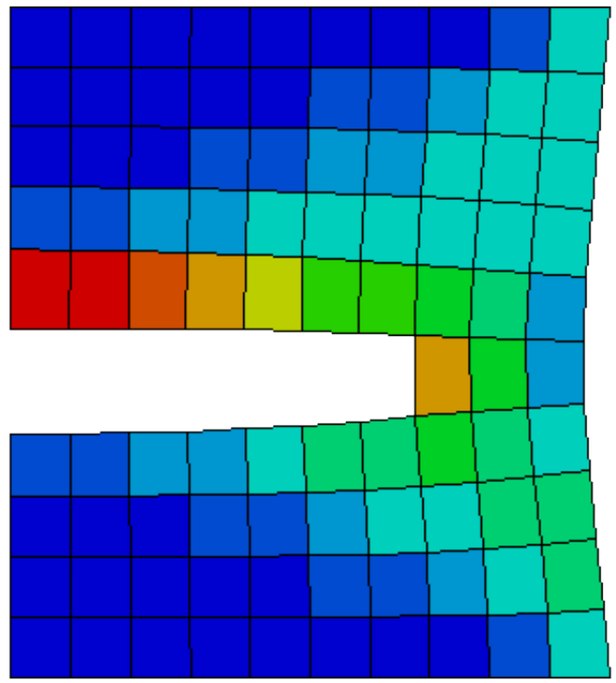
(V)USDFLD subroutine

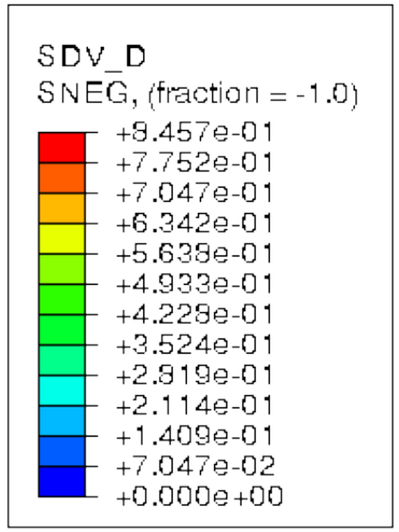
Results from the USDFLD_V2 and VUSDFLD_V2 subroutines



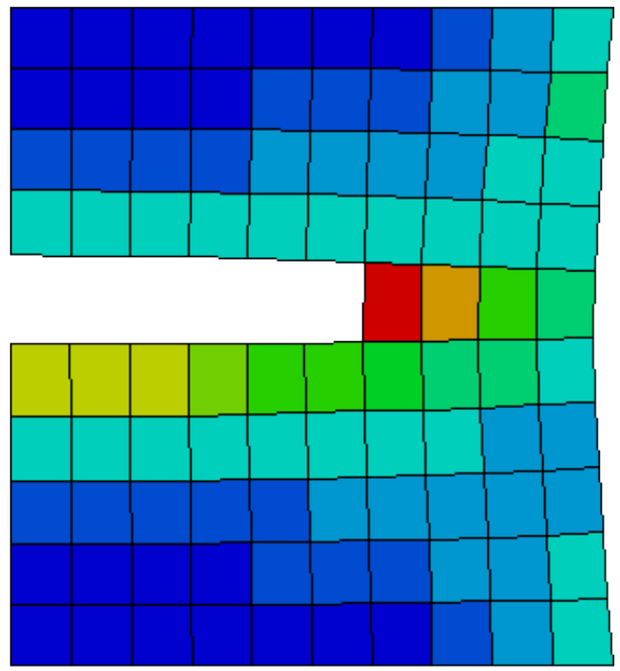


VUSDFLD_V2





USDFLD_V2



(V)USDFLD subroutine

These are quite flexible subroutine which can be used for various applications.

- One limitation for the USDFLD subroutine is that we can not input properties and must therefore “hard-code” them.
- The update of SDVs will be explicit (i.e. updated with previous time-step data) this can lead to strong time-step dependency.
- In ABAQUS/explicit there is a limited number of variables which can be accessed through the vgetvrm utility subroutine.