How to create and run Abaqus input files for use with the piezoelectricity user-element subroutine (UEL).

Mechanics of Materials Lab at MIT

May 7, 2021

1 Introduction

This supporting document serves as a guide for the reader on how to construct Abaqus input files for use with user-elements and the user-element subroutine (UEL). At the present moment, Abaqus does not allow for the inclusion of user-elements in a model via Abaqus/CAE. Thus, the addition and use of user-elements must be performed by direct modification of an Abaqus input file.

This guide is intended to help the reader create and/or modify their own input files for use with user-elements, it is not however intended as detailed instructions on how to modify the user-element subroutines (UEL) included with this work. Although this guide is aimed at readers of all skill levels, we presume some basic knowledge of the workings of Abaqus.

The workflow for creating an Abaqus input file for use with user-elements and running a simulation is as follows:

1. Create the geometry and the mesh of the desired model in Abaqus/CAE using standard built-in elements. Output the input file (.inp) for further modification. This step is detailed in Section 2.
2. On a text editor, modify the input file for use with user-elements. This step is detailed in Section 3.
3. If necessary, adjust parameters in the provided fortran file (.for) containing the `UEL` subroutine. This step is detailed in Section 4.
4. From the Abaqus terminal, run the simulation. This step is detailed in Section 5.

In all of the steps that follow, for more details or further clarification, refer to the Abaqus/Standard (2017) reference manual.

2 Create a model in Abaqus/CAE

For completeness, a very brief overview of the steps used in Abaqus/CAE for creating an input file to be modified for use with user-elements is provided below. Since we will modify the input file for use with user-elements, we do not need to create a working model in Abaqus/CAE. All we wish to output from Abaqus/CAE is the geometry and the mesh of the model that we want.

The Abaqus/CAE GUI is divided into modules, which are selected on the drop down menu next to "Module:". Below we list the steps necessary under each module to create the desired input file:

1. **Part Module:**
   - Draw Part. This can be a 2D planar or a 3D drawing.[1]
2. **Assembly Module:**

[1] PVDF and PZTS1 have different material properties in the three different basis directions, and so at this time we have only created a 3D UEL.

- Insert part into the Assembly. Choose any instance type (Dependent or Independent) since it does not matter for this purpose.
- Create all necessary element and node sets. As will be detailed in Section 3, we need node and element sets in order to prescribe boundary conditions, initial conditions, and material properties. These sets should be created in this module.
  - On the top bar, click on "Tools" $\longrightarrow$ "Set" $\longrightarrow$ "Manager". From the set manager click "Create..." and proceed to make sets of all the regions of interest.

3. **Mesh Module:**

- Seed the part with nodes.
- On the top bar, click "Mesh" $\longrightarrow$ "Element Type..." and assign the element type. **Important:** Make sure the elements you use have the same number of nodes as the user-elements that you wish to use. You must choose between hexahedra and tetrahedra, and between linear and quadratic elements.
- For complicated geometries, in a 3D example, abaqus might use a combination of tetrahedral and hexahedral elements to mesh the geometry. You may then wish to set the mesh controls to force Abaqus to use only quadrilaterals. To do this, on the top bar, click "Mesh" $\longrightarrow$ "Controls..." and change the "Element Shape" box to "Hex", rather than 'Hex-dominated".
- Mesh the part. On the top bar, click "Mesh" $\longrightarrow$ "Part...".

4. **Job Module:**

- Create a job. On the top bar, click "Job" $\longrightarrow$ "Create...". Give any job name you'd like, this will be the input file name.
- Before outputting the input file, on the model tree on the left hand side of the window, right click on your model name (the default is "Model-1") and then click on "Edit Attributes". Check the box that reads "Do not use parts and assemblies in input files". This will greatly simplify the structure of the resulting input file.
- Write the input file. On the top bar, click "Job" $\longrightarrow$ "Write Input" and then click on your jobname.
- The input file will be written to your working directory, or if you launched Abaqus/CAE from the terminal, to whichever directory you launched CAE from.

This will create an un-edited Abaqus input file which contains the finite-element description of the problem of interest. That is, it contains the location of all the nodes, the element connectivity, and all the desired node and element sets.

An example input file was created using the exact procedure described above and is provided in Section 7. It describes a $1\,\text{mm} \times 1\,\text{mm} \times 1\,\text{mm}$ three-dimensional cube domain meshed with eight 8-noded hexahedral elements. Node and element sets have been created for all six faces of the cube domain, which are simply denoted "top", "bottom", "right", "left", "front", and "back" corresponding in the standard isometric view orientation in Abaqus/CAE. A node and element set has also been created for the entire domain and is simply called "all".

3 Modification of an input file for use with user-elements

In this Section we describe in detail how one may modify an input file created with Abaqus/CAE, as was detailed in Section 2, for use with user-elements. In order to make this guide as clear as possible, we include in Section 7 an **un-modified** input file created with Abaqus/CAE and in Section 8 a **modified** input file which is ready for use with piezoelectricity user-elements. We provide detailed instructions on input file construction for a piezoelectric material with **isotropic elasticity**, and a similarly constructed example input file for a piezoelectric material with **orthotropic elasticity** is provided in Section 9.

We recommend that the reader print Sections 7 and 8 separately and use them to follow this guide. In the steps detailed below, we will denote by "Original text" lines of code from the un-modified input file, by "Modified text" lines of code from the un-modified input file which have been modified, and by "New text" lines of code which are not in the un-modified file and are introduced in the modified file.

1. **Heading and instance name:** The first lines of the original input file read:

```
*Heading
** Job name: Job-1 Model name: Model-1
** Generated by: Abaqus/CAE 2017
*Preprint, echo=NO, model=NO, history=NO, contact=NO
** ----------------------------------------------------------------
**
** PART INSTANCE: Part-1-1
**
```

} Original text

The only command in these lines that is necessary is `*Heading`, everything else can be erased. Typically, we simply modify this to

```
*Heading
Sample UEL Input File
```

} Modified text

The text underneath the `*Heading` command describes the model and will appear in the output (.odb) file once the simulation is performed. This is useful in keeping track of what model the input file describes.

2. **Node definitions:** The `*Node` command begins the node definitions. In this sample it reads:

```
*Node
      1, 0.00100000005, 0.00100000005, 0.00100000005
      2, 0.00100000005, 0.000500000024, 0.00100000005
      3, 0.00100000005,           0., 0.00100000005
      4, 0.00100000005, 0.00100000005, 0.000500000024
      5, 0.00100000005, 0.000500000024, 0.000500000024
      6, 0.00100000005,           0., 0.000500000024
      7, 0.00100000005, 0.00100000005,           0.
      8, 0.00100000005, 0.000500000024,           0.
      9, 0.00100000005,           0.,           0.
     10, 0.000500000024, 0.00100000005, 0.00100000005
     11, 0.000500000024, 0.000500000024, 0.00100000005
     12, 0.000500000024,           0., 0.00100000005
     13, 0.000500000024, 0.00100000005, 0.000500000024
     14, 0.000500000024, 0.000500000024, 0.000500000024
     15, 0.000500000024,           0., 0.000500000024
     16, 0.000500000024, 0.00100000005,           0.
     17, 0.000500000024, 0.000500000024,           0.
     18, 0.000500000024,           0.,           0.
     19,           0., 0.00100000005, 0.00100000005
     20,           0., 0.000500000024, 0.00100000005
     21,           0.,           0., 0.00100000005
     22,           0., 0.00100000005, 0.000500000024
     23,           0., 0.000500000024, 0.000500000024
     24,           0.,           0., 0.000500000024
     25,           0., 0.00100000005,           0.
     26,           0., 0.000500000024,           0.
     27,           0.,           0.,           0.
```

} Original text

Each row following the `*Node` command defines a node, where the value in the 1st column is the node number, the value in the 2nd column is the x-location, the value in the 3rd column is the y-location, and the 4th column is the z-location of the nodes. This part does not need to be modified.

3. **Element definitions:** In the un-modified sample input file the elements are defined through:[2]

```
*Element, type=C3D8R
1, 10, 11, 14, 13,  1,  2,  5,  4
2, 11, 12, 15, 14,  2,  3,  6,  5
3, 13, 14, 17, 16,  4,  5,  8,  7
4, 14, 15, 18, 17,  5,  6,  9,  8
5, 19, 20, 23, 22, 10, 11, 14, 13
6, 20, 21, 24, 23, 11, 12, 15, 14
7, 22, 23, 26, 25, 13, 14, 17, 16
8, 23, 24, 27, 26, 14, 15, 18, 17
```

} Original text

We modify this part of the code for use with user-elements. The modified text reads

```
*User Element,Nodes=8,Type=U1,Iproperties=2,Properties=10,Coordinates=3,Variables=1,Unsymm
1,2,3,11
**
*Element, type=U1, elset=el_real
1, 10, 11, 14, 13,  1,  2,  5,  4
2, 11, 12, 15, 14,  2,  3,  6,  5
3, 13, 14, 17, 16,  4,  5,  8,  7
4, 14, 15, 18, 17,  5,  6,  9,  8
5, 19, 20, 23, 22, 10, 11, 14, 13
6, 20, 21, 24, 23, 11, 12, 15, 14
7, 22, 23, 26, 25, 13, 14, 17, 16
8, 23, 24, 27, 26, 14, 15, 18, 17
```

} Modified text

Here, the `*User Element` command defines the user elements, and the arguments given have the following purpose:

- `Type=` sets the type of user element. Specifically, use:
  - `Type=U1`

  which is used to identify the User Element type in element definition.
- `Nodes=` is the number of nodes for the user element. For U3D8 `Nodes=8`.
- `Coordinates=` is the number of spatial coordinates, that is `Coordinates=3` for 3D simulations.
- `Properties=` is the number of **non-integer** (floating point) properties passed into the UEL subroutine. These may be material properties, element properties, etc...
- `Iproperties=` is the number of **integer** properties passed into the UEL subroutine.
- `Variables=` is the total number of state dependent variables for this element. Typically, this is equal to the number of internal variables multiplied by the number of integration points. For example if you have ten internal variables and four integration points, one would use `Variables=40`. In the Piezoelectricity UEL we have no state variables, so we simply set `Variables=1`.
- `Unsymm` tells Abaqus that we wish to use the unsymmetric solver.

The line immediately following the `*User element` command determines the degrees of freedom for the user element. In this case we have

[2] Depending of the specific element chosen when creating the starting input file with Abaqus/CAE, the argument `type=C3D8R` might have a different value.

```
1,2,3,11
```

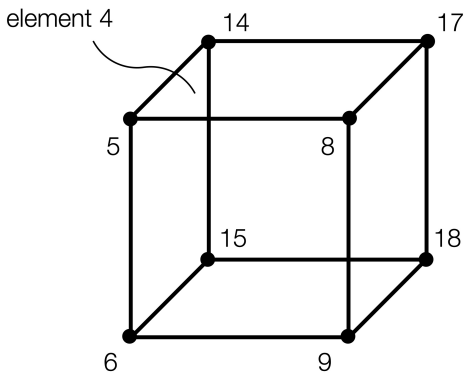
which in a standard Abaqus built-in element correspond to the x-displacement, the y-displacement, the z-displacement, and the temperature. The conventional definitions of degrees of freedom in Abaqus are listed in the user manual[3]. However, in the user-elements we are free to modify these degrees of freedom as necessary. Thus, degree of freedom `11` in this work represents the electrostatic potential ϕ.

Having defined the user element, we must now define the element connectivity. This is done in the lines:

```
*Element, type=U1, elset=el_real
1, 10, 11, 14, 13,  1,  2,  5,  4
2, 11, 12, 15, 14,  2,  3,  6,  5
3, 13, 14, 17, 16,  4,  5,  8,  7
4, 14, 15, 18, 17,  5,  6,  9,  8
5, 19, 20, 23, 22, 10, 11, 14, 13
6, 20, 21, 24, 23, 11, 12, 15, 14
7, 22, 23, 26, 25, 13, 14, 17, 16
8, 23, 24, 27, 26, 14, 15, 18, 17
```

In the first line, one would modify the command `type=U1` to match the type used when defining the user element under `*User element`. We also simultaneously create an element set named `el_real` containing all of the elements defined by stating `elset=el_real`.

The remaining lines define the element connectivity. Each row defines an element, where the 1st column is the element number. The 2nd through 9th columns define the eight nodes which make up the element, where the connectivity follows a counter-clockwise convention. For example, element `4` is connected by nodes `14, 15, 18, 17, 5, 6, 9, 8` which must be arranged in a counter-clockwise fashion such as



4. **Defining the "dummy" mesh:** Abaqus does not currently support visualization of user elements. Thus, for the purpose of visualizing, we superimpose a "dummy" mesh over the "real" user-element mesh. **The "dummy" mesh shares the same nodes as the user-element mesh but has a constant offset in element numbering**.

   In this example, the "dummy" mesh is defined through

[3]Abaqus conventions: `https://abaqus-docs.mit.edu/2017/English/SIMACAEMODRefMap/simamod-c-conventions.htm`

```
*Element, type=C3D8, elset=el_dummy
101, 10, 11, 14, 13,  1,  2,  5,  4
102, 11, 12, 15, 14,  2,  3,  6,  5
103, 13, 14, 17, 16,  4,  5,  8,  7
104, 14, 15, 18, 17,  5,  6,  9,  8
105, 19, 20, 23, 22, 10, 11, 14, 13
106, 20, 21, 24, 23, 11, 12, 15, 14
107, 22, 23, 26, 25, 13, 14, 17, 16
108, 23, 24, 27, 26, 14, 15, 18, 17
```

New text

where we note that we have **the same connectivity** however with a **constant offset in element numbering**. In this case the offset is equal to 100, this number must also be set in the fortran file (.for) containing the `UEL` subroutine, see Section 4. We also simultaneously create an element set named `el_dummy` containing all of the dummy elements defined by stating `elset=el_dummy`.

Further, we note that the element type is `type=C3D8`, which is an 8-noded fully-integrated element having only displacement degrees of freedom (degrees of freedom `1`, `2`, and `3`). The type of element chosen for the "dummy" mesh **must match** the number of nodes and the number of integration points used in the user-elements.

Important: The "dummy" elements should be purely mechanical — that is they do not have degree of freedom 11, such that their presence will in no way affect the behavior of the electrostatics problem. Further, in order for the "dummy" mesh to also not affect the mechanical behavior of the user-elements, as will be detailed later, we simply prescribe it a negligibly small elastic stiffness.

Finally we use the Abaqus UVARM subroutine to transfer the state dependent variables from the user-element mesh to the "dummy" mesh. We may then visualize any desired result through the "dummy" mesh. This technique for visualizing user-elements is suggested in the Abaqus documentation (29.16.1), although the additional use of the UVARM subroutine is a product of this work.

5. **Node and Element Sets:** The node and element sets are defined through the commands:

```
*Nset, nset=all, generate
  1,  27,   1
*Elset, elset=all, generate
 1,  8,  1
*Nset, nset=back
  7,  8,  9, 16, 17, 18, 25, 26, 27
*Elset, elset=back
 3, 4, 7, 8
*Nset, nset=bottom, generate
  3,  27,   3
*Elset, elset=bottom, generate
 2,  8,  2
*Nset, nset=front
  1,  2,  3, 10, 11, 12, 19, 20, 21
*Elset, elset=front
 1, 2, 5, 6
*Nset, nset=left, generate
 19,  27,   1
*Elset, elset=left, generate
 5,  8,  1
*Nset, nset=right, generate
 1,  9,  1
*Elset, elset=right, generate
 1,  4,  1
*Nset, nset=top, generate
  1,  25,   3
*Elset, elset=top, generate
 1,  7,  2
```

} Original text

Element sets are defined through the `*Elset` command. For example, the command above

```
*Elset, elset=all, generate
 1,  8,  1
```

defines an element set named `all` containing the elements `1 - 8`. The `generate` argument is used to generate the set without listing all of the element numbers. Instead, one provides three values, the element number of the first element, the element number of the last element, and the increment used to generate the list. Node sets are created in the same fashion using the `*Nset` command, and the generate argument applies equally to node sets.

6. **Simulation parameters.** Typically, it is useful to define simulation parameters (variables) which might define certain aspects of the loading or geometry of the problem. This is done through the `*Parameter` command. In this example we define

```
*Parameter
******************************
****** Step Definition ******
******************************
DISPL = 0.01e-3
TOTTIME = 60
dt0 = TOTTIME/1.e2
dtmin = TOTTIME/1.e5
dtmax = TOTTIME/1.e1
******************************
****** Elastic Constants ******
******************************
Eyoung = 3.6e9
anu = 0.34
******************************
******  Piezo Constants  ******
******************************
E31 = -36.0e-3
E32 = -60.0e-3
E33 = -151.0e-3
E24 = -62.0e-3
E15 = -73.0e-3
******************************
*** Permittivity Constants  ***
******************************
permitt0 = 8.85e-12
eps11 = 6.903*permitt0
eps22 = 8.604*permitt0
eps33 = 10.23*permitt0
******************************
****  Integer Parameters   ****
******************************
nlSdv = 1
ngSdv = 9
nInt = 8
nvars = nlSdv*nInt
```

New text

which define the relevant material parameters and integer parameters for the problem under study. Then, if one wishes to modify these aspects of the simulation this may be easily done by varying the defined parameters. For details on the parameters please see the sample input file.

7. **Defining material properties.** For details on the manner in which Abaqus handles units see Section 6.

   We must now define material properties for all the elements in our simulation. These are divided into three element sets, the set `el_real` to which all of the user-elements belong, the set `el_dummy` to which the "dummy" elements belong.

   a) First we prescribed the properties for the user-elements through the command

```
*uel property, elset=el_real
<Eyoung>, <anu>, <E31>, <E32>, <E33>, <E24>, <E15>, <eps11>,
<eps22>, <eps33>, <nlSdv>, <ngSdv>
```

   New text

   The command `*uel property` indicates that these are properties for user-elements, the command `elset=el_real` then tells Abaqus to which element set these properties should be prescribed.

   The subsequent lines include all of the non-integer, and integer properties, **with the non-integer properties listed before the integer properties**. A few things to note:

- You can at most list 8 properties per row.
- The number of non-integer and integer properties listed here **must match** that given when the `*User element` command was used to define the elements. Recall that we gave `Properties=10` and `Iproperties=2`, and as you may note above we have 10 non-integer properties, and 2 integer properties.
- If you wish to insert one of the previously defined parameters, this is done by including the variable name inside `<>`, as was done for here for the Young's Modulus `<Eyoung>`, for the Poisson's ratio `<anu>`, and so on.

Important: The first integer property, `nlSdv`, is the number of internal state variables per integration point. Further, recall that the number of state variables multiplied by the number of integration points per element, must be set in the `*User Element` command under the `Variables=` property.

The second integer property, `ngSdv`, is the number of user variables (for visualization) per element that we wish to have. This number should match the number used under the `*User output variables` command in the material definition for the "dummy" mesh.

b) We now prescribe the "dummy" element material properties through the commands

```
*Solid section, elset=el_dummy, material=umaterial
*Material, name=umaterial
**
*Elastic
1.e-20
**
*User output variables
<ngSdv>
**
*Density
1.0
**
*Specific heat
1.0
```

} New text

Here

- The command `*Solid section` assigns the material properties defined under the name `umaterial` to the elements in the set `el_dummy`.
- The command `*Material` indicates the start of a material definition with the name `umaterial`.
- The command `*User output variables` activates the UVARM subroutine and is followed by the number of variables that you wish to transfer to the "dummy" element for visualization.
- The piezoelectricity UEL has nine UVARM outputs which correspond to:

User Variable	Meaning
UVARM1	S^{R}_{11}
UVARM2	S^{R}_{22}
UVARM3	S^{R}_{33}
UVARM4	S^{R}_{23}
UVARM5	S^{R}_{13}
UVARM6	S^{R}_{12}
UVARM7	$\mathsf{d}^{\mathrm{R}}_{1}$
UVARM8	$\mathsf{d}^{\mathrm{R}}_{2}$
UVARM9	$\mathsf{d}^{\mathrm{R}}_{3}$

8. **Initial Conditions.** We now prescribe initial conditions. Specifically, we wish to apply an initial condition to the electrostatic potential ϕ, degree of freedom 11, which is 0.

Since the electrostatic potential ϕ is modeled using degree of freedom 11, that is the "temperature", we use the following command to prescribe its initial condition

```
*Initial conditions, type=temperature
all,0.0
```
} New text

The first item in the second line defines the node set to which the initial condition is applied, while the second item gives the corresponding value for the initial condition.

9. **Defining amplitudes.** As an example boundary condition, we wish to apply a displacement boundary condition on the top face of the cube. However, we wish to do so by ramping from 0 to DISPL in TOTTIME. We will apply this boundary condition, as detailed below, by first creating the following amplitude

```
*Amplitude, name=DispAmp
0.0,0.0,<TOTTIME>,<DISPL>
```
} New text

which ramps linearly from 0 at $t = 0$, to `DISPL` at `TOTTIME`.

10. **Defining steps.** We now define one or more simulation steps. In this particular example we define a single simulation step through the commands

```
*Step, Name=Deform, nlgeom=yes, inc=50000
*Coupled temperature-displacement, deltmx=10000
<dt0>,<TOTTIME>,<dtmin>,<dtmax>
*CONTROLS, PARAMETERS=TIME INCREMENTATION
,,,,,,,12,,,,,,
**        BOUNDARY CONDITIONS
*Boundary
back, 11,11,0
back, encastre
**
*Boundary, amplitude=DispAmp
front, 3, 3, -1.0
**
**
**        OUTPUT
*Output, field
*Node output
U, NT
*Element output, elset=el_dummy
UVARM
**
*End step
```
} New text

The above commands have the following specific purposes:

- `*Step` initializes the step and is followed by three arguments: `Name=Deform` assigns a particular name to the step, `nlgeom=yes` tells Abaqus to consider non-linear geometry, and `inc=50000` sets the maximum number of increments for this step.
- `*Coupled temperature-displacement` denotes the type of step, in this case we **must** use a coupled temperature-displacement step since we want the displacement and "temperature" degrees of freedom active for use in the user-elements. The argument `deltmx=10000` sets a maximum change of the temperature per increment, which will effectively set a maximum change of the electrostatic potential ϕ per increment (here we set this value arbitrarily high as we do not desire to limit the change in ϕ per step). The four parameters following the `*Coupled temperature-displacement` line define:

(1) initial time increment,
(2) total simulation time for the step,
(3) minimum time increment allowed, and
(4) maximum time increment allowed.

- `*Boundary` defines boundary conditions (BCs) at particular nodes or node sets. After this command one must prescribe four quantities

  (1) Node number or node set label,
  (2) first degree of freedom to which the BC is applied,
  (3) last degree of freedom to which the BC is applied, and
  (4) magnitude of the applied degree of freedom.

  For example, under the first `*Boundary` command, the first line

  ```
  back, 11,11,0
  ```

  constrains the nodes contained in the node set named `back`, by assigning to the degree of freedom 11 (electrostatic potential ϕ) a value equal to zero. This effectively electrically "grounds" the back surface of the cube. The second line

  ```
  back,encastre
  ```

  uses the built-in Abaqus 'fixed' BC `encastre`, which corresponds to setting DOFs `1-6` equal to zero, or the so-called "built-in" BC.

  The command

  ```
  *Boundary, amplitude=DispAmp
  front, 3, 3, -1.0
  ```

  is used to apply boundary conditions in the same fashion as before however here the magnitude of the applied degree of freedom (4th entry in the data lines) is multiplied by the amplitude given in `amplitude=DispAmp`. Here, the boundary condition applied amounts to linearly ramping the strain applied to the `front` face from 0% to 1%.

- `*Output` is used to define variables which should be output for visualization or analysis.

  The argument `field` tells Abaqus that the variables selected for output should be made available as field outputs, typically so that they may be visualized as contours over the simulation domain. You can also use the argument `history` to request a data array which contains the values of a particular variable on a particular node or element set. These `history` outputs are viewable in Abaqus Viewer by selecting `Result` → `History output...`

  All subsequent commands following the `*Output` command then define specific variables which should be saved. The command `*Node output` is used to output nodal quantities, namely the degrees of freedom. For example, the command

  ```
  *Output, field
  *Node output, nset=all
  U, NT
  ```

  creates field outputs for the displacements `u` and the "temperature" `nt` ($= \phi$) for all of the nodes in the node set `all`. This will allow one to visualize the deformation, and also visualize contours of the electrostatic potential ϕ. **Important**, if you do not output the displacements `u` as field outputs, you will not be able to visualize the deformation of the body.

  The command `*Element output` is used to output integration point quantities, such as the user variables (UVARMs). For example, the command

```
*Element output, elset=el_dummy
UVARM, le
```

outputs the user variables (UVARMs), as well as the logarithmic strains `le` for all of the elements in the element set `ElDummy`, that is for the "dummy" elements.

- `*End Step` must be invoked to finish the step definition. After this command subsequent steps may be defined as needed.

This completes our brief guide to the modification of an Abaqus/CAE input file for use with user-elements and the UEL subroutine. Before running a simulation, there are certain parameters in the input file which must also be set in the fortran file (.for) containing the `UVARM` and `UEL` subroutines. These modifications are briefly described in Section 4.

4 Modifications in the fortran file (.for)

Before running a simulation, there are some parameters which are set in the input file which must be **accordingly set** within the fortran file (.for) containing the `UVARM` and `UEL` subroutines. These are:

1. Under global module modify the following:
   - `parameter(numElem=8)` must be set equal to the total number of **user elements** used in the model. In this example that number is 4.
   - `parameter(ElemOffset=100)` must be set to the constant offset used between the UEL mesh and the "dummy" mesh. In this example that difference is 100.
2. If you wish to define additional `UVARM` user variables in the body of the UEL, you must then also modify the `ngSdv` parameter in the `*Parameters` of the input file to match the new total number of `UVARM`s. This will appropriately update the memory allocated to `UVARM`s in the `global` module.

5 Running a simulation

From the Abaqus command terminal, one may run a simulation by typing the following command:

```
abaqus double inp=[input file name].inp user=[UEL file name].for job=[desired identifier]
```

Note that you must run this command from the same directory containing both the input file and the fortran file containing all of the subroutines.

6 Regarding units in ABAQUS

Abaqus does not have a predefined system of units. The user defines what units are being used by simply being consistent with the values given. In the example described above, the basic unit system chosen is as follows

Basic Units	Length	m
	Mass	kg
	Time	seconds
	Temperature	K

and the dependent units are

Dependent Units	Force	N
	Stress	Pa
	Energy	J
	Density	kg/m^3

You can alternatively choose to work in, e.g. millimeters (mm), seconds (s), and kilograms (kg), in which case MPa becomes the new base unit of stress. In this unit system, for example, you would define a Young's modulus of 70 GPa in Abaqus as `70e3`.

7 Sample un-modified input file from Abaqus/CAE

The following input file was generated using Abaqus/CAE, and is a minimal example of an input file which can be subsequently modified for use with user-elements and the UEL subroutine. This file only contains information regarding the geometry of the problem, that is the location of the nodes, the mesh connectivity, and the necessary node and elements sets, and is not meant as a working input file.

```
*Heading
** Job name: 1mmCube Model name: Model-1
** Generated by: Abaqus/CAE 2017
*Preprint, echo=NO, model=NO, history=NO, contact=NO
** ----------------------------------------------------------------
**
** PART INSTANCE: Part-1-1
**
*Node
      1, 0.00100000005, 0.00100000005, 0.00100000005
      2, 0.00100000005, 0.000500000024, 0.00100000005
      3, 0.00100000005,           0., 0.00100000005
      4, 0.00100000005, 0.00100000005, 0.000500000024
      5, 0.00100000005, 0.000500000024, 0.000500000024
      6, 0.00100000005,           0., 0.000500000024
      7, 0.00100000005, 0.00100000005,           0.
      8, 0.00100000005, 0.000500000024,           0.
      9, 0.00100000005,           0.,           0.
     10, 0.000500000024, 0.00100000005, 0.00100000005
     11, 0.000500000024, 0.000500000024, 0.00100000005
     12, 0.000500000024,           0., 0.00100000005
     13, 0.000500000024, 0.00100000005, 0.000500000024
     14, 0.000500000024, 0.000500000024, 0.000500000024
     15, 0.000500000024,           0., 0.000500000024
     16, 0.000500000024, 0.00100000005,           0.
     17, 0.000500000024, 0.000500000024,           0.
     18, 0.000500000024,           0.,           0.
     19,           0., 0.00100000005, 0.00100000005
     20,           0., 0.000500000024, 0.00100000005
     21,           0.,           0., 0.00100000005
     22,           0., 0.00100000005, 0.000500000024
     23,           0., 0.000500000024, 0.000500000024
     24,           0.,           0., 0.000500000024
     25,           0., 0.00100000005,           0.
     26,           0., 0.000500000024,           0.
     27,           0.,           0.,           0.
*Element, type=C3D8R
1, 10, 11, 14, 13,  1,  2,  5,  4
2, 11, 12, 15, 14,  2,  3,  6,  5
3, 13, 14, 17, 16,  4,  5,  8,  7
4, 14, 15, 18, 17,  5,  6,  9,  8
5, 19, 20, 23, 22, 10, 11, 14, 13
6, 20, 21, 24, 23, 11, 12, 15, 14
7, 22, 23, 26, 25, 13, 14, 17, 16
8, 23, 24, 27, 26, 14, 15, 18, 17
*System
*Nset, nset=all, generate
```

```
  1,  27,   1
*Elset, elset=all, generate
 1,  8,  1
*Nset, nset=back
  7,  8,  9, 16, 17, 18, 25, 26, 27
*Elset, elset=back
 3, 4, 7, 8
*Nset, nset=bottom, generate
  3,  27,   3
*Elset, elset=bottom, generate
 2,  8,  2
*Nset, nset=front
  1,  2,  3, 10, 11, 12, 19, 20, 21
*Elset, elset=front
 1, 2, 5, 6
*Nset, nset=left, generate
 19,  27,   1
*Elset, elset=left, generate
 5,  8,  1
*Nset, nset=right, generate
 1,  9,  1
*Elset, elset=right, generate
 1,  4,  1
*Nset, nset=top, generate
  1,  25,   3
*Elset, elset=top, generate
 1,  7,  2
```

8 Sample modified input file for use with UEL (isotropic elasticity)

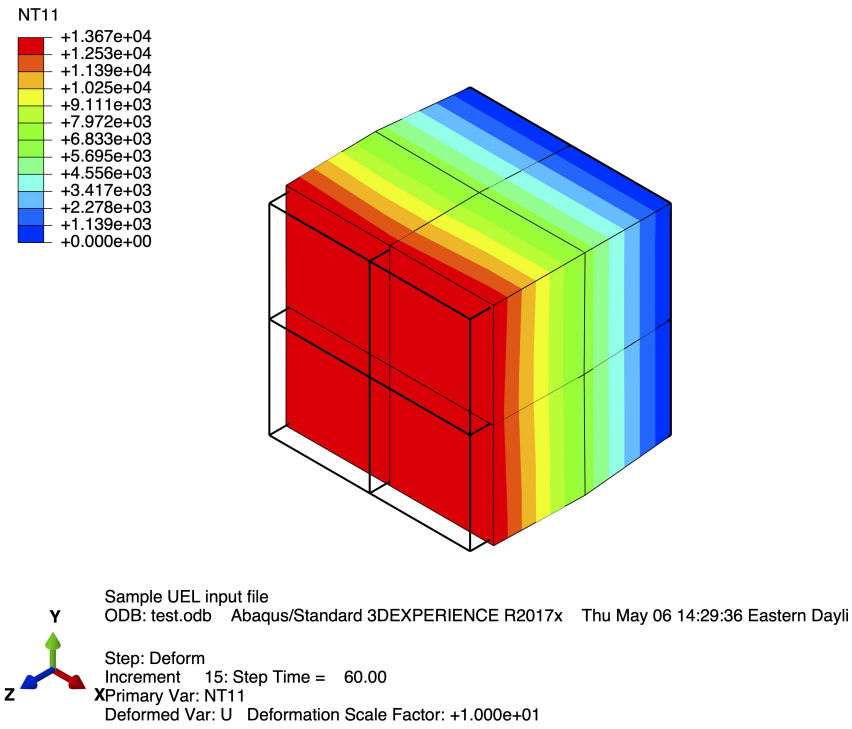
The following input file code is provided in its entirety as a minimal example of an input file which will work with the **isotropic elasticity** piezoelectricity user-element developed in the main body of this work. If desired, this code may be copied onto a .inp file and run with the UEL fortran subroutines also provided in this work. Helpful comments are added in addition to the modifications discussed in Section 3. Example output contours and the deformed geometry of the cube are also included, with minimal viewport modifications so as to appear similar to what a user will see upon opening the results file (`*.odb`) after running this file.

```
*Heading
Sample UEL input file
**
** UNITS: SI
**
*******************************
******** PARAMETERS ***********
*******************************
*Parameter
**
*******************************
****** Step Definition ********
*******************************
DISPL = 0.01e-3
TOTTIME = 60
dt0 = TOTTIME/1.e2
dtmin = TOTTIME/1.e5
dtmax = TOTTIME/1.e1
*******************************
****** Elastic Constants ******
*******************************
Eyoung = 3.6e9
anu = 0.34
*******************************
******  Piezo Constants  ******
*******************************
E31 = -36.0e-3
E32 = -60.0e-3
E33 = -151.0e-3
E24 = -62.0e-3
E15 = -73.0e-3
*******************************
*** Permittivity Constants  ***
*******************************
permitt0 = 8.85e-12
eps11 = 6.903*permitt0
eps22 = 8.604*permitt0
eps33 = 10.23*permitt0
*******************************
****  Integer Parameters   ****
*******************************
nlSdv = 1
ngSdv = 9
nInt = 8
```

```
nvars = nlSdv*nInt
**
**************************************************************************
** DOMAIN DEFINITION
**************************************************************************
**
*Node
      1, 0.00100000005, 0.00100000005, 0.00100000005
      2, 0.00100000005, 0.000500000024, 0.00100000005
      3, 0.00100000005,           0., 0.00100000005
      4, 0.00100000005, 0.00100000005, 0.000500000024
      5, 0.00100000005, 0.000500000024, 0.000500000024
      6, 0.00100000005,           0., 0.000500000024
      7, 0.00100000005, 0.00100000005,           0.
      8, 0.00100000005, 0.000500000024,           0.
      9, 0.00100000005,           0.,           0.
     10, 0.000500000024, 0.00100000005, 0.00100000005
     11, 0.000500000024, 0.000500000024, 0.00100000005
     12, 0.000500000024,           0., 0.00100000005
     13, 0.000500000024, 0.00100000005, 0.000500000024
     14, 0.000500000024, 0.000500000024, 0.000500000024
     15, 0.000500000024,           0., 0.000500000024
     16, 0.000500000024, 0.00100000005,           0.
     17, 0.000500000024, 0.000500000024,           0.
     18, 0.000500000024,           0.,           0.
     19,           0., 0.00100000005, 0.00100000005
     20,           0., 0.000500000024, 0.00100000005
     21,           0.,           0., 0.00100000005
     22,           0., 0.00100000005, 0.000500000024
     23,           0., 0.000500000024, 0.000500000024
     24,           0.,           0., 0.000500000024
     25,           0., 0.00100000005,           0.
     26,           0., 0.000500000024,           0.
     27,           0.,           0.,           0.
*User Element,Nodes=8,Type=U1,Iproperties=2,Properties=10,Coordinates=3,Variables=1,Unsymm
1,2,3,11
*Element, type=U1, elset=el_real
1, 10, 11, 14, 13,  1,  2,  5,  4
2, 11, 12, 15, 14,  2,  3,  6,  5
3, 13, 14, 17, 16,  4,  5,  8,  7
4, 14, 15, 18, 17,  5,  6,  9,  8
5, 19, 20, 23, 22, 10, 11, 14, 13
6, 20, 21, 24, 23, 11, 12, 15, 14
7, 22, 23, 26, 25, 13, 14, 17, 16
8, 23, 24, 27, 26, 14, 15, 18, 17
*Element, type=C3D8, elset=el_dummy
101, 10, 11, 14, 13,  1,  2,  5,  4
102, 11, 12, 15, 14,  2,  3,  6,  5
103, 13, 14, 17, 16,  4,  5,  8,  7
104, 14, 15, 18, 17,  5,  6,  9,  8
105, 19, 20, 23, 22, 10, 11, 14, 13
106, 20, 21, 24, 23, 11, 12, 15, 14
107, 22, 23, 26, 25, 13, 14, 17, 16
108, 23, 24, 27, 26, 14, 15, 18, 17
```

```
*Nset, nset=all, generate
  1,  27,   1
*Elset, elset=all, generate
 1,  8,  1
*Nset, nset=back
  7,  8,  9, 16, 17, 18, 25, 26, 27
*Elset, elset=back
 3, 4, 7, 8
*Nset, nset=bottom, generate
  3,  27,   3
*Elset, elset=bottom, generate
 2,  8,  2
*Nset, nset=front
  1,  2,  3, 10, 11, 12, 19, 20, 21
*Elset, elset=front
 1, 2, 5, 6
*Nset, nset=left, generate
 19,  27,   1
*Elset, elset=left, generate
 5,  8,  1
*Nset, nset=right, generate
 1,  9,  1
*Elset, elset=right, generate
 1,  4,  1
*Nset, nset=top, generate
  1,  25,   3
*Elset, elset=top, generate
 1,  7,  2
**
************************************************************************
** MATERIAL DEFINITION
************************************************************************
**
*uel property, elset=el_real
<Eyoung>, <anu>, <E31>, <E32>, <E33>, <E24>, <E15>, <eps11>,
<eps22>, <eps33>, <nlSdv>, <ngSdv>
**
*Solid section, elset=el_dummy, material=umaterial
*Material, name=umaterial
**
*Elastic
1.e-20
**
*User output variables
<ngSdv>
**
*Density
1.0
**
*Specific heat
1.0
**
************************************************************************
** INITIAL CONDITIONS
```

```
**********************************************************************
**
*Initial conditions, type=temperature
all,0.0
**
**********************************************************************
** AMPLITUDE DEFINITION
**********************************************************************
**
*Amplitude, name=DispAmp
0.0,0.0,<TOTTIME>,<DISPL>
**
**********************************************************************
** STEP DEFINITION
**********************************************************************
*Step, Name=Deform, nlgeom=yes, inc=50000
*Coupled temperature-displacement, deltmx=10000
<dt0>,<TOTTIME>,<dtmin>,<dtmax>
*CONTROLS, PARAMETERS=TIME INCREMENTATION
,,,,,,,12,,,,,,
**        BOUNDARY CONDITIONS
*Boundary
back, encastre
back, 11,11,0
**
*Boundary, amplitude=DispAmp
front, 3, 3, -1.0
**
**
**        OUTPUT
*Output, field
*Node output, nset=all
U, NT
*Element output, elset=el_dummy
UVARM, le
**
*End step
```


NT11
+1.367e+04
+1.253e+04
+1.139e+04
+1.025e+04
+9.111e+03
+7.972e+03
+6.833e+03
+5.695e+03
+4.556e+03
+3.417e+03
+2.278e+03
+1.139e+03
+0.000e+00
Sample UEL input file
ODB: test.odb Abaqus/Standard 3DEXPERIENCE R2017x Thu May 06 14:29:36 Eastern Dayli
Y
Z
X
Step: Deform
Increment 15: Step Time = 60.00
Primary Var: NT11
Deformed Var: U Deformation Scale Factor: +1.000e+01

9 Sample modified input file for use with UEL (orthotropic elasticity)

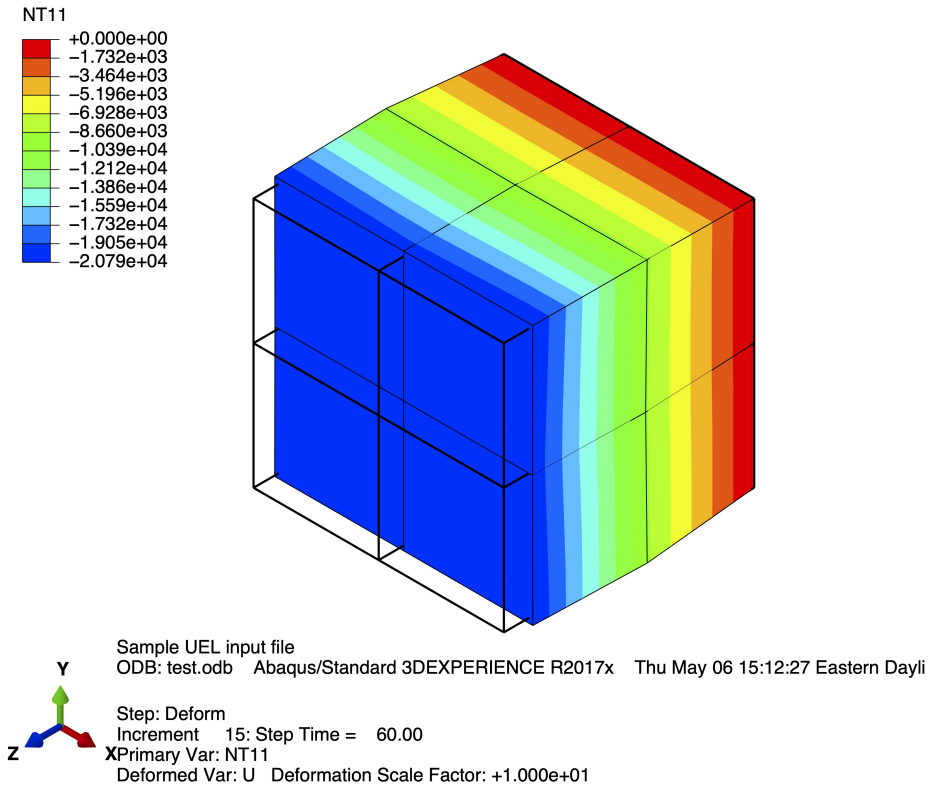
The following input file code is provided in its entirety as a minimal example of an input file which will work with the **orthotropic elasticity** piezoelectricity user-element developed in the main body of this work. If desired, this code may be copied onto a .inp file and run with the UEL fortran subroutines also provided in this work. Helpful comments are added in addition to the modifications discussed in Section 3. Example output contours and the deformed geometry of the cube are also included, with minimal viewport modifications so as to appear similar to what a user will see upon opening the results file (`*.odb`) after running this file.

```
*Heading
Sample UEL input file
**
** UNITS: SI
**
*******************************
******** PARAMETERS ************
*******************************
*Parameter
**
*******************************
****** Step Definition ********
*******************************
DISPL = 0.01e-3
TOTTIME = 60
dt0 = TOTTIME/1.e2
dtmin = TOTTIME/1.e5
dtmax = TOTTIME/1.e1
*******************************
****** Elastic Constants ******
*******************************
C1111 = 1.68E+11
C1122 = 1.11E+11
C2222 = 1.60E+11
C1133 = 1.01E+11
C2233 = 1.01E+11
C3333 = 1.23E+11
C2323 = 3.00E+10
C1313 = 3.00E+10
C1212 = 2.80E+10
*******************************
******  Piezo Constants  ******
*******************************
E31 = -2.8
E32 = -2.8
E33 = 14.72
E24 = 9.84
E15 = 9.84
*******************************
*** Permittivity Constants  ***
*******************************
eps11 = 1e-8
eps22 = 1e-8
```

```
eps33 = 8.09e-9
******************************
****  Integer Parameters   ****
******************************
nlSdv = 1
ngSdv = 9
nInt = 8
nvars = nlSdv*nInt
**
************************************************************************
** DOMAIN DEFINITION
************************************************************************
**
*Node
      1, 0.00100000005, 0.00100000005, 0.00100000005
      2, 0.00100000005, 0.000500000024, 0.00100000005
      3, 0.00100000005,           0., 0.00100000005
      4, 0.00100000005, 0.00100000005, 0.000500000024
      5, 0.00100000005, 0.000500000024, 0.000500000024
      6, 0.00100000005,           0., 0.000500000024
      7, 0.00100000005, 0.00100000005,           0.
      8, 0.00100000005, 0.000500000024,           0.
      9, 0.00100000005,           0.,           0.
     10, 0.000500000024, 0.00100000005, 0.00100000005
     11, 0.000500000024, 0.000500000024, 0.00100000005
     12, 0.000500000024,           0., 0.00100000005
     13, 0.000500000024, 0.00100000005, 0.000500000024
     14, 0.000500000024, 0.000500000024, 0.000500000024
     15, 0.000500000024,           0., 0.000500000024
     16, 0.000500000024, 0.00100000005,           0.
     17, 0.000500000024, 0.000500000024,           0.
     18, 0.000500000024,           0.,           0.
     19,           0., 0.00100000005, 0.00100000005
     20,           0., 0.000500000024, 0.00100000005
     21,           0.,           0., 0.00100000005
     22,           0., 0.00100000005, 0.000500000024
     23,           0., 0.000500000024, 0.000500000024
     24,           0.,           0., 0.000500000024
     25,           0., 0.00100000005,           0.
     26,           0., 0.000500000024,           0.
     27,           0.,           0.,           0.
*User Element,Nodes=8,Type=U1,Iproperties=2,Properties=17,Coordinates=3,Variables=1,Unsymm
1,2,3,11
*Element, type=U1, elset=el_real
1, 10, 11, 14, 13,  1,  2,  5,  4
2, 11, 12, 15, 14,  2,  3,  6,  5
3, 13, 14, 17, 16,  4,  5,  8,  7
4, 14, 15, 18, 17,  5,  6,  9,  8
5, 19, 20, 23, 22, 10, 11, 14, 13
6, 20, 21, 24, 23, 11, 12, 15, 14
7, 22, 23, 26, 25, 13, 14, 17, 16
8, 23, 24, 27, 26, 14, 15, 18, 17
*Element, type=C3D8, elset=el_dummy
101, 10, 11, 14, 13,  1,  2,  5,  4
```

```
102, 11, 12, 15, 14,  2,  3,  6,  5
103, 13, 14, 17, 16,  4,  5,  8,  7
104, 14, 15, 18, 17,  5,  6,  9,  8
105, 19, 20, 23, 22, 10, 11, 14, 13
106, 20, 21, 24, 23, 11, 12, 15, 14
107, 22, 23, 26, 25, 13, 14, 17, 16
108, 23, 24, 27, 26, 14, 15, 18, 17
*Nset, nset=all, generate
  1,  27,   1
*Elset, elset=all, generate
 1,  8,  1
*Nset, nset=back
  7,  8,  9, 16, 17, 18, 25, 26, 27
*Elset, elset=back
 3, 4, 7, 8
*Nset, nset=bottom, generate
  3,  27,   3
*Elset, elset=bottom, generate
 2,  8,  2
*Nset, nset=front
  1,  2,  3, 10, 11, 12, 19, 20, 21
*Elset, elset=front
 1, 2, 5, 6
*Nset, nset=left, generate
 19,  27,   1
*Elset, elset=left, generate
 5,  8,  1
*Nset, nset=right, generate
 1,  9,  1
*Elset, elset=right, generate
 1,  4,  1
*Nset, nset=top, generate
  1,  25,   3
*Elset, elset=top, generate
 1,  7,  2
**
**********************************************************************
** MATERIAL DEFINITION
**********************************************************************
**
*uel property,elset=el_real
<C1111>, <C1122>, <C2222>, <C1133>, <C2233>, <C3333>, <C1212>, <C1313>,
<C2323>, <E31>, <E32>, <E33>, <E24>, <E15>, <eps11>, <eps22>,
<eps33>, <nlSdv>, <ngSdv>
**
*Solid section, elset=el_dummy, material=umaterial
*Material, name=umaterial
**
*Elastic
1.e-20
**
*User output variables
<ngSdv>
**
```

```
*Density
1.0
**
*Specific heat
1.0
**
**********************************************************************
** INITIAL CONDITIONS
**********************************************************************
**
*Initial conditions, type=temperature
all,0.0
**
**********************************************************************
** AMPLITUDE DEFINITION
**********************************************************************
**
*Amplitude, name=DispAmp
0.0,0.0,<TOTTIME>,<DISPL>
**
**********************************************************************
** STEP DEFINITION
**********************************************************************
*Step, Name=Deform, nlgeom=yes, inc=50000
*Coupled temperature-displacement, deltmx=10000
<dt0>,<TOTTIME>,<dtmin>,<dtmax>
*CONTROLS, PARAMETERS=TIME INCREMENTATION
,,,,,,,12,,,,,,
**        BOUNDARY CONDITIONS
*Boundary
back, encastre
back, 11,11,0
**
*Boundary, amplitude=DispAmp
front, 3, 3, -1.0
**
**
**        OUTPUT
*Output, field
*Node output, nset=all
U, NT
*Element output, elset=el_dummy
UVARM, le
**
*End step
```



References

Abaqus/Standard, 2017. SIMULIA, Providence, RI.