LS-DYNA 的材料模型二次开发过程

Zhidong Han, Brian Wainscott
Livermore Software Technology Corp.

摘要

上期我们介绍了 LS-DYNA 新一代二次开发环境的编译连接过程和新增功能，多个用户子程序可以通过动态连接库的方式同时动态加载。本文是其续篇，将以一个简单的材料模型来演示在新的环境下的一个完整的开发过程，包括编译，连接，动态加载，源程序跟踪调试，以及模型验证等环节。

引言

LS-DYNA 作为一个大型的通用有限元程序，对于多重非线性的大规模问题的解决具有独特的优势，在实际工程中也得到非常广泛的应用。目前材料库有 300 种材料模型，其中多数都提供二维平面应力和三维应力两个版本。LS-DYNA 提供完整的用户材料模型的开发模板，让用户可以开发自己的材料模型。与一般的隐式算法相比，显式有限元分析的时间步长很小，计算规模大，导致对用户子程序的调用非常频繁。LS-DYNA 为减少子程序的调用，内部采用批处理的方式调用用户子程序，要求一次调用能处理几百个单元，这也为用户子程序实现矢量化计算提供了方便。因此，考虑到大变形，LS-DYNA 对用户子程序的特殊要求也增加了用户开发的复杂度。另外，对于一个对初次接触 LS-DYNA 的用户来说，主程序的执行码不带调试信息，较难在源程序上跟踪调试，加大了二次开发中的程序查错的难度。

本文以一个简单大变形下的线弹性材料模型为例，演示在新的开发环境下的完整的开发，调试和验证过程。

线弹性材料物理模型

1）应力应变关系

在 LS-DYNA 中应力和应变都是 6 个分量，排列顺序为

应力为：$\underline{\sigma}=\begin{Bmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{Bmatrix}$　　和应变为：$\underline{\varepsilon}=\begin{Bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{Bmatrix}$

线弹性的材料模型为：$\underline{\sigma}=\underline{D}\,\underline{\varepsilon}$

$$\underline{D}=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & \nu & 0 & 0 & 0\\ \nu & 1-\nu & \nu & 0 & 0 & 0\\ \nu & \nu & 1-\nu & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0\\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0\\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix}$$

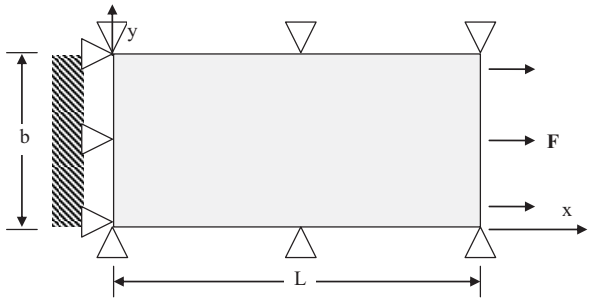
其中 E 和 ν 分别为杨氏模量和泊松比。

2）验证算例

本文的验证算例为只有一个 8 节点体单元的模型，如图一所示，长为 L=2.0m，宽和高均为 b=1.0m。加载条件为在 X 方向单拉，而在 Y 及 Z 方向的位移为零。上述应力应变关系在小变形的情况下则简化为

应变：$\varepsilon_x = \varepsilon = \frac{u}{L}, \quad \varepsilon_y = 0, \quad \varepsilon_z = 0$

应力：$\sigma_x = \sigma = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\varepsilon, \quad \sigma_y = \frac{\nu}{(1-\nu)}\sigma, \quad \sigma_z = \frac{\nu}{(1-\nu)}\sigma$



图一　　简单的单向加载模型

LS-DYNA 用户子程序开发和调试

1）UMAT 子程序的编译和连接

LS-DYNA 中的用户材料号是从 41 号到 50 号，对应的第一级用户入口子程序是 dyn21.f 中的 usrmat，受关键字*MAT_USER_DEFINED_MATERIAL_MODELS 控制。

```
  subroutine usrmat (lft,llt,cm,bqs,capa,eltype,mt,ipt,
 . npc,plc,crv,nnpcrv,rcoor,scoor,tcoor,nnm1,nip,ipt_thk)
```

进入这个子程序后，再根据不同的单元类型选择不同的第二级材料子程序

- urmathn: 体单元的三维材料模型
- urmats: 壳单元的二维平面应力材料模型
- urmatb, urmatd, urmatt: 三种不同的梁单元模型

这三个不同子程序根据各自的单元特点对应力应变进行相应的第二级处理之后，再进入的第三级的用户子程序 umat41, umat42, … , umat50。这 10 个子程序是标准的串行版本模板，演示不同类型的材料模型。一般的开发过程中，第一级和第二级入口程序都不需要改动,只需要从第三级这 10 个子程序模板中选一个较为贴近的开始。本文选择 umat41 这个子程序。

进入 LS-DYNA 开发包的目录后，进行如下几个操作步骤：

1. 把 umat41 子程序从 dyn21.f 中删除，并复制到另一个新的源文件 umat41.f

```
  subroutine umat41 (cm,eps,sig,epsp,hsv,dt1,capa,etype,tt,
 1 temper,failel,crv,nnpcrv,cma,qmat,elsiz,idele,reject)
```

2. 编辑开发包中的 Makefile，把 umat41.f 的 obj 文件加到 MY_OBJS 变量中

```
MY_OBJS = dyn21.o dyn21b.o init_dyn21.o couple2other_user.o dynrfn_user.o umat41.o
```

3. 选择编译器

原来的编译器设置是和主程序一致的，LINUX 系统一般是 INTEL 或者 PGI 的 Fortran 编译器，这些商业编译器的执行代码一般来说效率比较高。在 LS-DYNA 新的编译环境下，用户子程序的编译器不要求和主程序一致，本文采用开源的 gfortran 来演示编译过程。编译环境为：

Linux 系统：OpenSUSE LEAP 42.1

编译器：gfortran 4.8.5

MPI: platformmpi Community Edition 9.1.2

(http://www-03.ibm.com/systems/platformcomputing/products/mpi/)

将 Makefile 中的编译变量设置为

```
MY_FLAG = -g -fPIC -fcray-pointer -I/opt/platform_mpi/include
FC = /usr/bin/gfortran
LD = /usr/bin/gfortran -shared
export MPI_F77 := /usr/bin/gfortran
MY_TARGET = gnu.so
```

其中-g 是让编译的用户模块带源程序的调试跟踪信息。这些变量的详细解释请参阅上期的“LS-DYNA 用户子程序的编译和连接”一节。

4.用 make 命令编译，生成 gnu.so，就完成了编辑和连接。

2）UMAT 子程序的调用

上面编译好的 gnu.so 可以做为开发好的用户模块配合模型使用。这个模块和 LS-DYNA 主执行程序是分开的，即使将来 LS-DYNA 主程序的版本升级也不影响这个模块。调用的方法是在模型的.k 文件里面加入三行

```
*MODULE_LOAD
myumat41
gnu.so
```

其中：第一行是关键字，第二行是这个模块在这个模型的 id，第三行是这个模块的编译后文件。然后就可以按照原来的方法执行 LSDYNA 主程序就可以了。这个关键字有很多匹配规则，详见上期的“LS-DYNA 用户子程序的动态连接库的调用”一节。本文演示的是 MPP 版本的主程序，单个单元模型只能用一个 CPU 来运行：

```
mppdyna i=demo.k
```

3）UMAT 子程序的跟踪调试

当子程序运行遇到问题的时候，最简单直接方法是用打印命令，与 LS-DYNA 的 59 号文件对应的是 message 文件，对于 MPP 程序，每个 CPU 都有一个自己的 message 文件，因此打印方法不容易混乱，也很方便。比如：

```
WRITE(59,*)'sig=',sig(1),sig(2),sig(3)
WRITE(59,*)'hsv=',hsv(1),hsv(2)
```

有些情况下，还是要进入到源程序里面，在源程序上进行跟踪调试。本文以 gdb 为例，启动调试程序，进行以下步骤：

1. 调入主程序

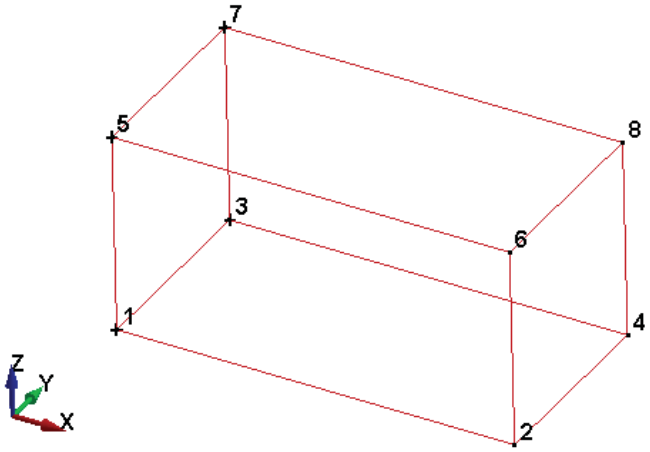
gdb mppdyna

2. 设置断点

b umat41
(此时会显示 umat41 不存在，可能要用 set breakpoint pending on 激活在调用时补设断点)
3. 运行程序
r i=demo.k
4. 程序在进入 umat41 后，就会停下来。比如，打印变量
p cm(1)
这是* MAT_USER_DEFINED_MATERIAL_MODELS 卡片输入的第一个材料常数 P1。

材料模型的验证

根据前面定义的物理模型，利用 LS-Prepost 建立一个有限元模型，包含一个八节点的实体单元，长度 L=2m，宽和高为 b=1m。并用 LS-Prepost 施加所有的边界条件和给定位移以及材料属性后，有限元模型如图二所示：



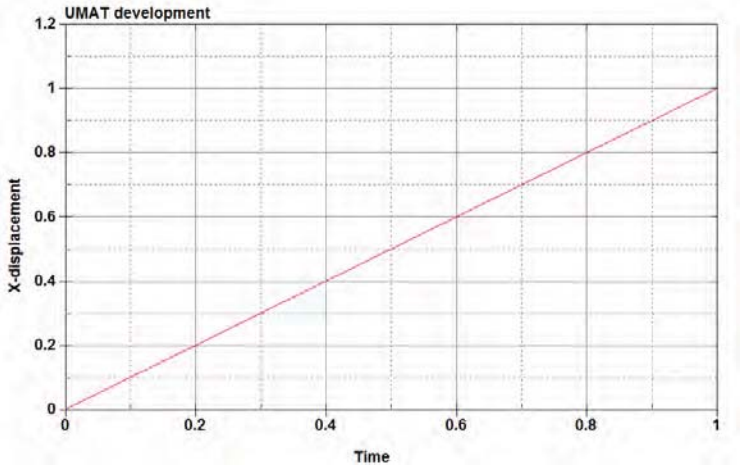
图二　　简单的单向加载有限元模型

所有的节点都固定，只有 2，4，6，8 节点在 X 方向有指定速度为 1.0m/s。分析时间为 1s，则节点位移和工程应变时间曲线为：

$u(t)=t$

$\varepsilon(t)=u(t)/L=t/2$

与图三的结果吻合得很好。

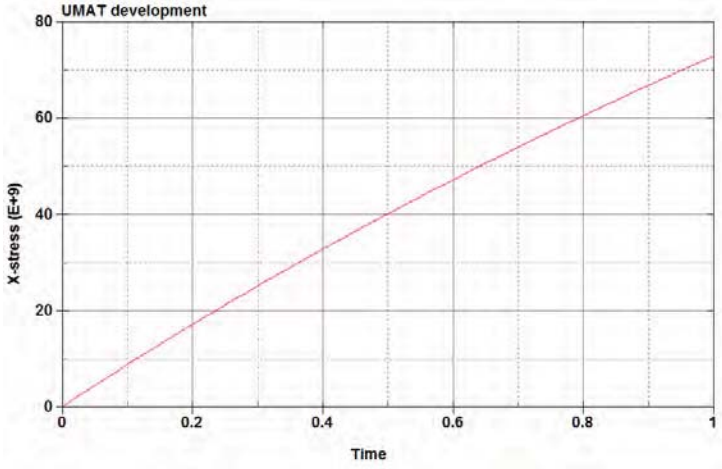


图三　　单向加载位移时间曲线

材料模型为线弹性， $E=150\times10^9$ 及 $\nu=0.25$ ，则应力时间曲线为：

$\sigma(t)=180\times10^9\varepsilon(t)=90\times10^9 t$

而图四中计算结果给出的最大应力则只有 $\sigma_{\max}=73\times10^9$ Pa



图四　　单向加载真实应力时间曲线

原因是 LS-DYNA 中的应变都是真实应变，而不是上面计算的工程应变。真实应变的计算方法为：

$e(t)=\ln[1+\varepsilon(t)]$

得到最大真实应变 $e_{\max}=\ln(1.5)=0.405$

代入应力计算公式可以得到最大真实应力的理论值是 73×10^9 Pa，与有限元的分析结果完全吻合。同时计算结果也表明，当时间 $t=0.001$，工程应变 $\varepsilon(0.001)=0.005$，仅为千分之五，而此时的真实应力为 $\sigma(0.001)=89.7\times10^9 t$，与线性公式相差 0.3%。而当时间 $t=0.2$ 时，工程应变达到百分之十， $\varepsilon(0.2)=0.1$，真实应力为 $\sigma(0.2)=171\times10^9 t$，与线性公式相差 4.7%。

在开发 LS-DYNA 的用户子程序的时候，一定要考虑大变形情况下的本构关系。一般来说，把线性小变形的本构关系直接放进 LS-DYNA，真实应力会有很大的差别。这点与一般的小变形下的本构模型开发有很大的不同，也加大了 LS-DYNA 二次开发的难度。

附录：算例.k 文件

```
*KEYWORD
*TITLE
$#                                                                         title
UMAT development
*MODULE_LOAD
myumat41
gnu.so
*CONTROL_TERMINATION
$#  endtim    endcyc     dtmin    endeng    endmas
  1.000000         0     0.000     0.000 1.0000E+8
*DATABASE_BINARY_D3PLOT
$#      dt      lcdt      beam     npltc    psetid
    1.0E-2         0         0         0         0
*BOUNDARY_PRESCRIBED_MOTION_SET
$#    nsid       dof       vad      lcid        sf       vid     death     birth
         1         1         0         1       1.0         0 1.0000E28       0.0
*SET_NODE_LIST_TITLE
x=L
$#     sid       da1       da2       da3       da4    solver
         1       0.0       0.0       0.0       0.0MECH
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
         2         4         6         8         0         0         0         0
```

```
*BOUNDARY_SPC_SET
$#    nsid       cid      dofx      dofy      dofz     dofrx     dofry     dofrz
         2         0         1         0         0         0         0         0
*SET_NODE_LIST_TITLE
x=0
$#     sid       da1       da2       da3       da4    solver
         2       0.0       0.0       0.0       0.0MECH
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
         1         3         5         7         0         0         0         0
*BOUNDARY_SPC_SET
$#    nsid       cid      dofx      dofy      dofz     dofrx     dofry     dofrz
         3         0         0         1         1         0         0         0
*SET_NODE_LIST_TITLE
all
$#     sid       da1       da2       da3       da4    solver
         3       0.0       0.0       0.0       0.0MECH
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
         1         2         3         4         5         6         7         8
*PART
$#                                                                         title
one-element test
$#     pid     secid       mid     eosid      hgid      grav    adpopt      tmid
         1         1         1         0         0         0         0         0
*SECTION_SOLID
$#   secid    elform       aet
         1         1         0
*MAT_USER_DEFINED_MATERIAL_MODELS
$#     mid        ro        mt       lmc       nhv    iortho     ibulk        ig
         1    7800.0        41         4         0         0         3         4
$#   ivect     ifail    itherm    ihyper      ieos      lmca    unused    unused
         0         0         0         0         0         0
$#      p1        p2        p3        p4        p5        p6        p7        p8
    1.5E11      0.25    1.0E11      6E10
*DEFINE_CURVE_TITLE
X-velocity
$#    lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         1         0       1.0       1.0       0.0       0.0         0         0
$#                a1                  o1
                0.0                 1.0
                1.0                 1.0
*ELEMENT_SOLID
$#   eid     pid      n1      n2      n3      n4      n5      n6      n7      n8
       1       1       1       2       4       3       5       6       8       7
*NODE
$#   nid               x               y               z      tc      rc
       1             0.0             0.0             0.0       0       0
       2             2.0             0.0             0.0       0       0
       3             0.0             1.0             0.0       0       0
       4             2.0             1.0             0.0       0       0
       5             0.0             0.0             1.0       0       0
       6             2.0             0.0             1.0       0       0
       7             0.0             1.0             1.0       0       0
       8             2.0             1.0             1.0       0       0
*MAT_ELASTIC
$#     mid        ro         e        pr        da        db  not used
        11    7800.0    1.5E11      0.25       0.0       0.0         0
*END
```

作者简介

*韩志东/Zhidong Han 博士 1998 年毕业于清华大学计算固体力学专业，于 2011 年加入 LSTC。他目前从事材料损伤断裂分析及厚壳单元等方面研发。