In[ ]:=
```
(* Initialisation *)
(* Evaluate before start writing "real code" *)
(* Usage e.g.: "ld [Spacekey]" becomes "⊨",
so writing "a ld 5" turns into "a ⊨ 5" *)
SetOptions[EvaluationNotebook[],
              InputAutoReplacements → {(* special AceGen assignment operators: *)
   "ld" → "⊨", "ls" → "⊢", "rd" → "⥤", "rs" → "⊣",
                                       (* brackets and symbols: *) "dbl" → "⟦",
   "dbr" → "⟧", "lcb" → "{", "rcb" → "}", "lsb" → "[", "rsb" → "]", "->" → "→",
                                       (* shortcuts for
    starting/ending a comment block: *) "co" → "(*", "cc" → "*)"
                                     }
           ]
(* Output the current time,
so we know when AceGen has been executed the last time *)
Now
```

Out[ ]=
```
Wed 22 May 2024 11:25:37 GMT+2
```

In[ ]:=
```
(*initialization *)
ClearAll["Global`*"](*all variables are initially cleared*)

(* NAME OF SUBROUTINE YOU WANT TO PRODUCE *)
NAME = "stabilisation_Q1LES ";
```

In[ ]:=
```
<< AceGen`;(*AceGen is started*)
```

```
In[ ]:= shape_functions_Hex8 [ξ1_, ξ2_, ξ3_] :=
          (
              1 / 8 * {(1 - ξ1) × (1 - ξ2) × (1 - ξ3), (*N1*)
                    (1 + ξ1) × (1 - ξ2) × (1 - ξ3), (*N2*)
                    (1 + ξ1) × (1 + ξ2) × (1 - ξ3), (*N3*)
                    (1 - ξ1) × (1 + ξ2) × (1 - ξ3), (*N4*)
                    (1 - ξ1) × (1 - ξ2) × (1 + ξ3), (*N5*)
                    (1 + ξ1) × (1 - ξ2) × (1 + ξ3), (*N6*)
                    (1 + ξ1) × (1 + ξ2) × (1 + ξ3), (*N7*)
                    (1 - ξ1) × (1 + ξ2) × (1 + ξ3)  (*N8*)
                   }
          );
        quadrature_rule_full  :=
        (
                s1by3 ⧦ Sqrt[1 / 3];
                QP_coords ⧦ Transpose[{{-s1by3 , -s1by3 , -s1by3}, (*Q1*)
                                      {+s1by3 , -s1by3 , -s1by3}, (*Q2*)
                                      {+s1by3 , +s1by3 , -s1by3}, (*Q3*)
                                      {-s1by3 , +s1by3 , -s1by3}, (*Q4*)
                                      {-s1by3 , -s1by3 , +s1by3}, (*Q5*)
                                      {+s1by3 , -s1by3 , +s1by3}, (*Q6*)
                                      {+s1by3 , +s1by3 , +s1by3}, (*Q7*)
                                      {-s1by3 , +s1by3 , +s1by3}  (*Q8*)
                                      }];
                QP_weights ⧦ {1, 1, 1, 1, 1, 1, 1, 1};
                Return[{QP_coords , QP_weights }];
          );
        quadrature_rule_reduced  :=
        (
                 QP_coords ⧦ {{0}, {0}, {0}};  (*Q9*)
                 QP_weights ⧦ {8};

                 Return[{QP_coords , QP_weights }];
        );

In[ ]:= dofs_per_node = 3;
        n_nodes = 8;
        ndtot = n_nodes * dofs_per_node ;
```

In[ ]:=
```
(* Programming language, Mode: Debug/Prototype/Optimal *)
SMSInitialize[NAME, "Language" → "Fortran", "Mode" → "Optimal"];
(* Create the module named NAME with all inputs and outputs *)
(* Inputs:
 - X: (8,3) array with undeformed nodal
    coordinates of the 8 nodes of this 3D hexahedral element
  - u: (24) array with displacement values (degrees of freedom) listed as
        { u_N1_x, u_N1_y, u_N1_z, u_N2_x, u_N2_y, ..., u_N8_z } for each Node N1...
      N8 with components {x,y,z}
     - bulkModkappa: bulk modulus of the base material
     - shearModmu: shear modulus of the base material
       - HGscale: hourglass control coefficient,
e.g. 1e-4 (higher values cause stronger stabilisation)
 - istif: integer/boolean to request the stiffness matrix,
istif=1: stiffness matrix is requested thus computed therein,
istif=0: stiffness matrix is not requested and not computed herein
   Outputs:
  - forceHG: (24) array of internal force components for hourglass
      stabilisation for each degree of freedom of the current element
     - stiffHG: (24x24) array with component of the stiffness
       matrix for the stabilisation ´for the current element
*)
SMSModule[NAME, Real[X$$[n_nodes, dofs_per_node],
   u$$[n_nodes, dofs_per_node], bulkModkappa$$, shearModmu$$, HGscale$$],
  Integer[istif$$], Real[forceHG$$[ndtot], stiffHG$$[ndtot, ndtot]],
  "Input" → {X$$, u$$, bulkModkappa$$, shearModmu$$, HGscale$$, istif$$},
  "Output" → {forceHG$$, stiffHG$$}];
```

In[ ]:=
```
(* Input declaration / copy Acegen variables to Mathematica variables *)
XIO ⊨ SMSReal[Table[X$$[iNode, jdof], {iNode, n_nodes}, {jdof, dofs_per_node}]];
uIO ⊨ SMSReal[Table[u$$[iNode, jdof], {iNode, n_nodes}, {jdof, dofs_per_node}]];
pe ⊨ Flatten[uIO];

κ ⊨ SMSReal[bulkModkappa$$];
μ ⊨ SMSReal[shearModmu$$];
λ ⊨ κ - 2 / 3 * μ;

HGscaleValue ⊨ SMSReal[HGscale$$];
istif ⊨ SMSInteger[istif$$];
```

```
(* Initialise output variables to zero
 (can be included optionally, if not done by the caller subroutine) *)
 (* SMSExport[Table[0,{idof,ndtot}],forceHG$$];
  SMSExport[Table[0,{idof,ndtot},{jdof,ndtot}],stiffHG$$];*)

(* Standard QP coordinates and weights
 for full integration (FuI) of stabilisation *)
 {QP_coords_FuI, QP_weights_FuI} ⊨ quadrature_rule_full;
(* Reduced integration (RI) for energy compensation *)
 {QP_coords_RI, QP_weights_RI} ⊨ quadrature_rule_reduced;
(* Merge the two lists, with first FuI followed by RI *)
 QP_coords = Join[QP_coords_FuI, QP_coords_RI, 2];
 QP_weights = Join[QP_weights_FuI, QP_weights_RI];

(* Loop over combined list of QPs *)
 SMSDo[qpoint, 1, 9];
   (* Using SMSPart to access qpoint's entry: *)
    ξ1 ⊢ SMSReal[SMSPart[QP_coords⟦1⟧, qpoint]];
    ξ2 ⊢ SMSReal[SMSPart[QP_coords⟦2⟧, qpoint]];
    ξ3 ⊢ SMSReal[SMSPart[QP_coords⟦3⟧, qpoint]];
   Ξ = {ξ1, ξ2, ξ3};
   weight_qpoint ⊢ SMSPart[QP_weights, qpoint];
   Nh ⊨ shape_functions_Hex8 [ξ1, ξ2, ξ3];
   X ⊢ SMSFreeze[Nh . XIO];
   u ⊨ Nh . uIO;
   Je ⊨ SMSD[X, Ξ];
   Jed ⊨ Det[Je];
   ℍ ⊨ SMSD[u, X, "Dependency" → {Ξ, X, SMSInverse[Je]}];
   ϵ ⊨ 1/2 * (ℍ^T + ℍ);

   (* For the fully integrated QPs we add the
  stabilising energy with a positive HG scaling factor *)
    SMSIf[qpoint ≤ 8];
      HGscale ⊣ HGscaleValue * 1;
   (* and for the reduced integrated centre QP (qpoint==9),
 we remove the "same" energy (1 QP with weight=8)
  leaving "only" the hourglass stabilising energy *)
    SMSElse[];
      HGscale ⊣ HGscaleValue * (-1);
    SMSEndIf[HGscale];

   (* Linear elastic energy *)
```

```
W := HGscale * (λ/2 * (Tr[ϵ])^2 + μ * Tr[ϵ . ϵ]);

(* Compute the residual/force and the stiffness matrix *)
 SMSDo[m, 1, ndtot];
  ℝgm := Jed * SMSD[W, pe, m];
  SMSExport[weight_qpoint * ℝgm, forceHG$$[m], "AddIn" → True];
  SMSDo[n, 1, ndtot];
    𝕂gmn := SMSIf[istif == 1, SMSD[ℝgm, pe, n], 0.0];
    SMSExport[weight_qpoint * 𝕂gmn, stiffHG$$[m, n], "AddIn" → True];
 SMSEndDo[];
 SMSEndDo[];

SMSEndDo[]; (*End Gauss Quadrature Loop*)
```

In[ ]:=
```
(* write output file *)
SMSWrite[NAME, "LocalAuxiliaryVariables " → True];

(* print file on screen *)
NAME_FileExtension = Which[SMSLanguage == "Fortran",
   ".f", SMSLanguage == "Matlab", ".m", SMSLanguage == "C++", ".cpp"];
FilePrint[StringJoin[NAME, NAME_FileExtension]]
```

File: stabilisation_Q1LES .f **Size:** 15 895 **Time:** 7

Method	stabilisation_Q1LES
No.Formulae	235
No.Leafs	4136

```
!************************************************************
!* AceGen     7.505 Linux (16 Aug 22)                       *
!*            Co. J. Korelc   2020            22 May 24 11:25:46 *
!************************************************************
! User      : Full professional version
! Notebook  : stabilisation_Q1LES
! Evaluation time                  : 7 s      Mode   : Optimal
! Number of formulae               : 235      Method: Automatic
! Subroutine                       : stabilisation_Q1LES   size: 4136
! Total size of Mathematica   code : 4136 subexpressions
! Total size of Fortran code       : 15274 bytes

!******************   S U B R O U T I N E ********************
       SUBROUTINE stabilisation_Q1LES(X,u,bulkModkappa,shearModmu
      &,HGscale,istif,forceHG,stiffHG)
       IMPLICIT NONE
       include 'sms.h'
       INTEGER istif,i55,i191,i203
       LOGICAL b188,b204,b205,b270
```

```
 DOUBLE PRECISION v(734),X(8,3),u(8,3),bulkModkappa
&,shearModmu,HGscale,forceHG(24),stiffHG(24,24)
 v(348)=1d0
 v(349)=1d0
 v(350)=1d0
 v(351)=1d0
 v(352)=1d0
 v(353)=1d0
 v(354)=1d0
 v(355)=1d0
 v(356)=8d0
 v(339)=(-0.5773502691896257d0)
 v(340)=(-0.5773502691896257d0)
 v(341)=(-0.5773502691896257d0)
 v(342)=(-0.5773502691896257d0)
 v(343)=0.5773502691896257d0
 v(344)=0.5773502691896257d0
 v(345)=0.5773502691896257d0
 v(346)=0.5773502691896257d0
 v(347)=0d0
 v(330)=(-0.5773502691896257d0)
 v(331)=(-0.5773502691896257d0)
 v(332)=0.5773502691896257d0
 v(333)=0.5773502691896257d0
 v(334)=(-0.5773502691896257d0)
 v(335)=(-0.5773502691896257d0)
 v(336)=0.5773502691896257d0
 v(337)=0.5773502691896257d0
 v(338)=0d0
 v(321)=(-0.5773502691896257d0)
 v(322)=0.5773502691896257d0
 v(323)=0.5773502691896257d0
 v(324)=(-0.5773502691896257d0)
 v(325)=(-0.5773502691896257d0)
 v(326)=0.5773502691896257d0
 v(327)=0.5773502691896257d0
 v(328)=(-0.5773502691896257d0)
 v(329)=0d0
 v(1)=X(1,1)
 v(2)=X(1,2)
 v(3)=X(1,3)
 v(4)=X(2,1)
 v(263)=-v(1)+v(4)
 v(5)=X(2,2)
 v(259)=-v(2)+v(5)
 v(6)=X(2,3)
 v(255)=-v(3)+v(6)
 v(7)=X(3,1)
 v(252)=v(4)-v(7)
 v(8)=X(3,2)
 v(248)=v(5)-v(8)
 v(9)=X(3,3)
 v(244)=v(6)-v(9)
 v(10)=X(4,1)
 v(264)=-v(10)+v(7)
```

```
v(251)=v(1)-v(10)
v(11)=X(4,2)
v(260)=-v(11)+v(8)
v(247)=-v(11)+v(2)
v(12)=X(4,3)
v(256)=-v(12)+v(9)
v(243)=-v(12)+v(3)
v(13)=X(5,1)
v(238)=v(1)-v(13)
v(14)=X(5,2)
v(234)=-v(14)+v(2)
v(15)=X(5,3)
v(230)=-v(15)+v(3)
v(16)=X(6,1)
v(265)=-v(13)+v(16)
v(239)=-v(16)+v(4)
v(17)=X(6,2)
v(261)=-v(14)+v(17)
v(235)=-v(17)+v(5)
v(18)=X(6,3)
v(257)=-v(15)+v(18)
v(231)=-v(18)+v(6)
v(19)=X(7,1)
v(254)=v(16)-v(19)
v(240)=-v(19)+v(7)
v(20)=X(7,2)
v(250)=v(17)-v(20)
v(236)=-v(20)+v(8)
v(21)=X(7,3)
v(246)=v(18)-v(21)
v(232)=-v(21)+v(9)
v(22)=X(8,1)
v(266)=v(19)-v(22)
v(253)=v(13)-v(22)
v(241)=v(10)-v(22)
v(23)=X(8,2)
v(262)=v(20)-v(23)
v(249)=v(14)-v(23)
v(237)=v(11)-v(23)
v(24)=X(8,3)
v(258)=v(21)-v(24)
v(245)=v(15)-v(24)
v(233)=v(12)-v(24)
v(25)=u(1,1)
v(26)=u(1,2)
v(27)=u(1,3)
v(28)=u(2,1)
v(29)=u(2,2)
v(30)=u(2,3)
v(31)=u(3,1)
v(32)=u(3,2)
v(33)=u(3,3)
v(34)=u(4,1)
v(35)=u(4,2)
v(36)=u(4,3)
```

```
v(37)=u(5,1)
v(38)=u(5,2)
v(39)=u(5,3)
v(40)=u(6,1)
v(41)=u(6,2)
v(42)=u(6,3)
v(43)=u(7,1)
v(44)=u(7,2)
v(45)=u(7,3)
v(46)=u(8,1)
v(47)=u(8,2)
v(48)=u(8,3)
v(50)=shearModmu
v(227)=2d0*v(50)
v(51)=bulkModkappa+(-2d0/3d0)*v(50)
v(52)=HGscale
b204=istif.eq.1
DO i55=1,9
 v(56)=v(320+i55)
 v(67)=1d0-v(56)
 v(63)=1d0+v(56)
 v(57)=v(329+i55)
 v(73)=1d0+v(57)
 v(85)=v(73)/8d0
 v(93)=-(v(67)*v(85))
 v(92)=-(v(63)*v(85))
 v(69)=1d0-v(57)
 v(86)=v(69)/8d0
 v(91)=-(v(63)*v(86))
 v(89)=-(v(67)*v(86))
 v(104)=v(230)*v(89)+v(231)*v(91)+v(232)*v(92)+v(233)*v(93)
 v(101)=v(234)*v(89)+v(235)*v(91)+v(236)*v(92)+v(237)*v(93)
 v(98)=v(238)*v(89)+v(239)*v(91)+v(240)*v(92)+v(241)*v(93)
 v(58)=v(338+i55)
 v(84)=(1d0+v(58))/8d0
 v(95)=-(v(63)*v(84))
 v(94)=-(v(67)*v(84))
 v(76)=v(73)*v(84)
 v(71)=v(69)*v(84)
 v(87)=(1d0-v(58))/8d0
 v(90)=-(v(63)*v(87))
 v(88)=-(v(67)*v(87))
 v(103)=v(243)*v(88)+v(244)*v(90)+v(245)*v(94)+v(246)*v(95)
 v(100)=v(247)*v(88)+v(248)*v(90)+v(249)*v(94)+v(250)*v(95)
 v(224)=-(v(101)*v(103))+v(100)*v(104)
 v(97)=v(251)*v(88)+v(252)*v(90)+v(253)*v(94)+v(254)*v(95)
 v(66)=v(73)*v(87)
 v(61)=v(69)*v(87)
 v(102)=v(255)*v(61)+v(256)*v(66)+v(257)*v(71)+v(258)*v(76)
 v(99)=v(259)*v(61)+v(260)*v(66)+v(261)*v(71)+v(262)*v(76)
 v(226)=-(v(100)*v(102))+v(103)*v(99)
 v(225)=v(101)*v(102)-v(104)*v(99)
 v(96)=v(263)*v(61)+v(264)*v(66)+v(265)*v(71)+v(266)*v(76)
 v(59)=v(347+i55)
 v(105)=v(224)*v(96)+v(225)*v(97)+v(226)*v(98)
```

```
v(106)=-(v(224)/v(105))
v(169)=-(v(106)*v(76))
v(160)=v(106)*v(71)
v(151)=-(v(106)*v(66))
v(142)=v(106)*v(61)
v(107)=(v(104)*v(97)-v(103)*v(98))/v(105)
v(171)=-(v(107)*v(76))
v(162)=v(107)*v(71)
v(153)=-(v(107)*v(66))
v(144)=v(107)*v(61)
v(108)=(-(v(101)*v(97))+v(100)*v(98))/v(105)
v(173)=-(v(108)*v(76))
v(164)=v(108)*v(71)
v(155)=-(v(108)*v(66))
v(146)=v(108)*v(61)
v(109)=v(225)/v(105)
v(130)=v(109)*v(87)
v(118)=v(109)*v(84)
v(110)=(v(104)*v(96)-v(102)*v(98))/v(105)
v(133)=v(110)*v(87)
v(120)=v(110)*v(84)
v(111)=(-(v(101)*v(96))+v(98)*v(99))/v(105)
v(136)=v(111)*v(87)
v(122)=v(111)*v(84)
v(112)=v(226)/v(105)
v(131)=v(112)*v(86)
v(124)=v(112)*v(85)
v(113)=(-(v(103)*v(96))+v(102)*v(97))/v(105)
v(134)=v(113)*v(86)
v(126)=v(113)*v(85)
v(114)=(v(100)*v(96)-v(97)*v(99))/v(105)
v(137)=v(114)*v(86)
v(128)=v(114)*v(85)
v(115)=v(118)+v(124)
v(116)=v(120)+v(126)
v(117)=v(122)+v(128)
v(119)=-v(118)+v(131)
v(121)=-v(120)+v(134)
v(123)=-v(122)+v(137)
v(125)=-v(124)+v(130)
v(127)=-v(126)+v(133)
v(129)=-v(128)+v(136)
v(132)=-v(130)-v(131)
v(135)=-v(133)-v(134)
v(138)=-v(136)-v(137)
v(139)=v(142)+v(132)*v(67)
v(140)=v(144)+v(135)*v(67)
v(141)=v(146)+v(138)*v(67)
v(143)=-v(142)+v(132)*v(63)
v(145)=-v(144)+v(135)*v(63)
v(147)=-v(146)+v(138)*v(63)
v(148)=v(151)+v(125)*v(63)
v(149)=v(153)+v(127)*v(63)
v(150)=v(155)+v(129)*v(63)
v(152)=-v(151)+v(125)*v(67)
```

```
    v(154)=-v(153)+v(127)*v(67)
    v(156)=-v(155)+v(129)*v(67)
    v(157)=v(160)+v(119)*v(67)
    v(158)=v(162)+v(121)*v(67)
    v(159)=v(164)+v(123)*v(67)
    v(161)=-v(160)+v(119)*v(63)
    v(163)=-v(162)+v(121)*v(63)
    v(165)=-v(164)+v(123)*v(63)
    v(166)=v(169)+v(115)*v(63)
    v(167)=v(171)+v(116)*v(63)
    v(168)=v(173)+v(117)*v(63)
    v(170)=-v(169)+v(115)*v(67)
    v(172)=-v(171)+v(116)*v(67)
    v(174)=-v(173)+v(117)*v(67)
    v(175)=v(139)*v(25)+v(143)*v(28)+v(148)*v(31)+v(152)*v(34)+v
&  (157)*v(37)+v(161)*v(40)+v(166)*v(43)+v(170)*v(46)
    v(198)=v(140)*v(25)+v(139)*v(26)+v(145)*v(28)+v(143)*v(29)+v
&  (149)*v(31)+v(148)*v(32)+v(154)*v(34)+v(152)*v(35)+v(158)*v(37
&  )+v(157)*v(38)+v(163)*v(40)+v(161)*v(41)+v(167)*v(43)+v(166)*v
&  (44)+v(172)*v(46)+v(170)*v(47)
    v(179)=v(140)*v(26)+v(145)*v(29)+v(149)*v(32)+v(154)*v(35)+v
&  (158)*v(38)+v(163)*v(41)+v(167)*v(44)+v(172)*v(47)
    v(199)=v(141)*v(25)+v(139)*v(27)+v(147)*v(28)+v(143)*v(30)+v
&  (150)*v(31)+v(148)*v(33)+v(156)*v(34)+v(152)*v(36)+v(159)*v(37
&  )+v(157)*v(39)+v(165)*v(40)+v(161)*v(42)+v(168)*v(43)+v(166)*v
&  (45)+v(174)*v(46)+v(170)*v(48)
    v(200)=v(141)*v(26)+v(140)*v(27)+v(147)*v(29)+v(145)*v(30)+v
&  (150)*v(32)+v(149)*v(33)+v(156)*v(35)+v(154)*v(36)+v(159)*v(38
&  )+v(158)*v(39)+v(165)*v(41)+v(163)*v(42)+v(168)*v(44)+v(167)*v
&  (45)+v(174)*v(47)+v(172)*v(48)
    v(436)=v(140)*v(198)+v(141)*v(199)
    v(437)=v(139)*v(198)+v(141)*v(200)
    v(438)=v(139)*v(199)+v(140)*v(200)
    v(439)=v(145)*v(198)+v(147)*v(199)
    v(440)=v(143)*v(198)+v(147)*v(200)
    v(441)=v(143)*v(199)+v(145)*v(200)
    v(442)=v(149)*v(198)+v(150)*v(199)
    v(443)=v(148)*v(198)+v(150)*v(200)
    v(444)=v(148)*v(199)+v(149)*v(200)
    v(445)=v(154)*v(198)+v(156)*v(199)
    v(446)=v(152)*v(198)+v(156)*v(200)
    v(447)=v(152)*v(199)+v(154)*v(200)
    v(448)=v(158)*v(198)+v(159)*v(199)
    v(449)=v(157)*v(198)+v(159)*v(200)
    v(450)=v(157)*v(199)+v(158)*v(200)
    v(451)=v(163)*v(198)+v(165)*v(199)
    v(452)=v(161)*v(198)+v(165)*v(200)
    v(453)=v(161)*v(199)+v(163)*v(200)
    v(454)=v(167)*v(198)+v(168)*v(199)
    v(455)=v(166)*v(198)+v(168)*v(200)
    v(456)=v(166)*v(199)+v(167)*v(200)
    v(457)=v(172)*v(198)+v(174)*v(199)
    v(458)=v(170)*v(198)+v(174)*v(200)
    v(459)=v(170)*v(199)+v(172)*v(200)
    v(183)=v(141)*v(27)+v(147)*v(30)+v(150)*v(33)+v(156)*v(36)+v
```

```
& (159)*v(39)+v(165)*v(42)+v(168)*v(45)+v(174)*v(48)
  v(195)=(v(175)+v(179)+v(183))*v(51)
  IF(i55.le.8)  THEN
   v(189)=v(52)
  ELSE
   v(189)=-v(52)
  ENDIF
  v(217)=v(189)*v(227)
  v(212)=(v(105)*v(217))/2d0
  v(194)=v(189)*(v(195)+v(183)*v(227))
  v(196)=v(189)*(v(195)+v(179)*v(227))
  v(197)=v(189)*(v(195)+v(175)*v(227))
  v(388)=v(139)*v(197)
  v(389)=v(140)*v(196)
  v(390)=v(141)*v(194)
  v(391)=v(143)*v(197)
  v(392)=v(145)*v(196)
  v(393)=v(147)*v(194)
  v(394)=v(148)*v(197)
  v(395)=v(149)*v(196)
  v(396)=v(150)*v(194)
  v(397)=v(152)*v(197)
  v(398)=v(154)*v(196)
  v(399)=v(156)*v(194)
  v(400)=v(157)*v(197)
  v(401)=v(158)*v(196)
  v(402)=v(159)*v(194)
  v(403)=v(161)*v(197)
  v(404)=v(163)*v(196)
  v(405)=v(165)*v(194)
  v(406)=v(166)*v(197)
  v(407)=v(167)*v(196)
  v(408)=v(168)*v(194)
  v(409)=v(170)*v(197)
  v(410)=v(172)*v(196)
  v(411)=v(174)*v(194)
  IF(b204)  THEN
   v(587)=v(140)
   v(588)=v(139)
   v(589)=0d0
   v(590)=v(145)
   v(591)=v(143)
   v(592)=0d0
   v(593)=v(149)
   v(594)=v(148)
   v(595)=0d0
   v(596)=v(154)
   v(597)=v(152)
   v(598)=0d0
   v(599)=v(158)
   v(600)=v(157)
   v(601)=0d0
   v(602)=v(163)
   v(603)=v(161)
   v(604)=0d0
```

```
v(605)=v(167)
v(606)=v(166)
v(607)=0d0
v(608)=v(172)
v(609)=v(170)
v(610)=0d0
v(563)=v(141)
v(564)=0d0
v(565)=v(139)
v(566)=v(147)
v(567)=0d0
v(568)=v(143)
v(569)=v(150)
v(570)=0d0
v(571)=v(148)
v(572)=v(156)
v(573)=0d0
v(574)=v(152)
v(575)=v(159)
v(576)=0d0
v(577)=v(157)
v(578)=v(165)
v(579)=0d0
v(580)=v(161)
v(581)=v(168)
v(582)=0d0
v(583)=v(166)
v(584)=v(174)
v(585)=0d0
v(586)=v(170)
v(539)=0d0
v(540)=v(141)
v(541)=v(140)
v(542)=0d0
v(543)=v(147)
v(544)=v(145)
v(545)=0d0
v(546)=v(150)
v(547)=v(149)
v(548)=0d0
v(549)=v(156)
v(550)=v(154)
v(551)=0d0
v(552)=v(159)
v(553)=v(158)
v(554)=0d0
v(555)=v(165)
v(556)=v(163)
v(557)=0d0
v(558)=v(168)
v(559)=v(167)
v(560)=0d0
v(561)=v(174)
v(562)=v(172)
v(515)=0d0
```

```
v(516)=0d0
v(517)=v(141)
v(518)=0d0
v(519)=0d0
v(520)=v(147)
v(521)=0d0
v(522)=0d0
v(523)=v(150)
v(524)=0d0
v(525)=0d0
v(526)=v(156)
v(527)=0d0
v(528)=0d0
v(529)=v(159)
v(530)=0d0
v(531)=0d0
v(532)=v(165)
v(533)=0d0
v(534)=0d0
v(535)=v(168)
v(536)=0d0
v(537)=0d0
v(538)=v(174)
v(491)=0d0
v(492)=v(140)
v(493)=0d0
v(494)=0d0
v(495)=v(145)
v(496)=0d0
v(497)=0d0
v(498)=v(149)
v(499)=0d0
v(500)=0d0
v(501)=v(154)
v(502)=0d0
v(503)=0d0
v(504)=v(158)
v(505)=0d0
v(506)=0d0
v(507)=v(163)
v(508)=0d0
v(509)=0d0
v(510)=v(167)
v(511)=0d0
v(512)=0d0
v(513)=v(172)
v(514)=0d0
v(467)=v(139)
v(468)=0d0
v(469)=0d0
v(470)=v(143)
v(471)=0d0
v(472)=0d0
v(473)=v(148)
v(474)=0d0
```

```
      v(475)=0d0
      v(476)=v(152)
      v(477)=0d0
      v(478)=0d0
      v(479)=v(157)
      v(480)=0d0
      v(481)=0d0
      v(482)=v(161)
      v(483)=0d0
      v(484)=0d0
      v(485)=v(166)
      v(486)=0d0
      v(487)=0d0
      v(488)=v(170)
      v(489)=0d0
      v(490)=0d0
    ELSE
    ENDIF
    DO i191=1,24
     forceHG(i191)=forceHG(i191)+v(105)*v(59)*(v(387+i191)+v(189
&    )*v(50)*v(435+i191))
     DO i203=1,24
      IF(b204) THEN
       v(208)=v(105)*v(466+i191)
       v(209)=v(105)*v(490+i191)
       v(210)=v(105)*v(514+i191)
       v(211)=v(212)*v(538+i191)
       v(213)=v(212)*v(562+i191)
       v(214)=v(212)*v(586+i191)
       v(218)=v(189)*(v(208)+v(209)+v(210))*v(51)
       v(216)=v(210)*v(217)+v(218)
       v(219)=v(209)*v(217)+v(218)
       v(220)=v(208)*v(217)+v(218)
       v(611)=v(141)*v(213)+v(140)*v(214)+v(139)*v(220)
       v(612)=v(141)*v(211)+v(139)*v(214)+v(140)*v(219)
       v(613)=v(140)*v(211)+v(139)*v(213)+v(141)*v(216)
       v(614)=v(147)*v(213)+v(145)*v(214)+v(143)*v(220)
       v(615)=v(147)*v(211)+v(143)*v(214)+v(145)*v(219)
       v(616)=v(145)*v(211)+v(143)*v(213)+v(147)*v(216)
       v(617)=v(150)*v(213)+v(149)*v(214)+v(148)*v(220)
       v(618)=v(150)*v(211)+v(148)*v(214)+v(149)*v(219)
       v(619)=v(149)*v(211)+v(148)*v(213)+v(150)*v(216)
       v(620)=v(156)*v(213)+v(154)*v(214)+v(152)*v(220)
       v(621)=v(156)*v(211)+v(152)*v(214)+v(154)*v(219)
       v(622)=v(154)*v(211)+v(152)*v(213)+v(156)*v(216)
       v(623)=v(159)*v(213)+v(158)*v(214)+v(157)*v(220)
       v(624)=v(159)*v(211)+v(157)*v(214)+v(158)*v(219)
       v(625)=v(158)*v(211)+v(157)*v(213)+v(159)*v(216)
       v(626)=v(165)*v(213)+v(163)*v(214)+v(161)*v(220)
       v(627)=v(165)*v(211)+v(161)*v(214)+v(163)*v(219)
       v(628)=v(163)*v(211)+v(161)*v(213)+v(165)*v(216)
       v(629)=v(168)*v(213)+v(167)*v(214)+v(166)*v(220)
       v(630)=v(168)*v(211)+v(166)*v(214)+v(167)*v(219)
       v(631)=v(167)*v(211)+v(166)*v(213)+v(168)*v(216)
       v(632)=v(174)*v(213)+v(172)*v(214)+v(170)*v(220)
```

```
      v(633)=v(174)*v(211)+v(170)*v(214)+v(172)*v(219)
      v(634)=v(172)*v(211)+v(170)*v(213)+v(174)*v(216)
      v(222)=v(610+i203)
     ELSE
      v(222)=0d0
     ENDIF
     stiffHG(i191,i203)=stiffHG(i191,i203)+v(222)*v(59)
    ENDDO
   ENDDO
ENDDO
END
```