Appendix A

Matlab code

The brought modifications are reported by comments starting and finishing by "!!!!!"
Example: "!!!!! This is a modification !!!!!"

Modifications: mbs_coord_part.m

Changes from line 65 to line 96.

```
1  function [ind_u, ind_v, ind_cont, ind_red, err] =
       mbs_coord_part(ind_u_des,ind_c,IS_ROW_PERM,callfct,
       first_partitioning)
2  % ------------------------------
3  % UCL-CEREM-MBS
4  %
5  % @version MBsysLab_m 1.7.a
6  %
7  % Creation : 2005
8  % Last update : 26/10/2005
9  % ------------------------------
10 %@MBS_COORD_PART ...
11 %
12 %   [IND_U,IND_V,IND_CONT,IND_RED] = MBS_COORD_PART(
       IND_U_DES,IND_C,IS_ROW_PERM,CALLFCT)
13 %
14 %   IND_V          meilleur choix ET permutation des
       variables v
15 %   IND_U          meilleur choix des variables u (dont
       certaines sont choisies)
```

```
16 %   IND_CONT       meilleure permutation des contraintes
17 %   IND_RED        meilleur choix des contraintes
   redondantes (si existent)
18 %   IS_ROW_PERM    flag de permutation des lignes
19 %   first_aprtitioning, flag : = 1 la première fois (
   systeme non fermé), 0 la
20 %   seconde (systeme fermé)
21 %
22
23 %   See also
24 %
25 %   Gestion via Bugzilla :
26 %   > 08/04/2009 : PF-JFC : Détonyfication du module de
   partitionnement -
27 %   retour à la version de base - voir Bug n°54
28 %
29 %---
30
31 global MBS_data;
32
33 err = 0;
34
35 % Calcul du rang réel de la matrice Jacobienne
36
37 qrand = rand(MBS_data.nbody,1);
38 q_safe = MBS_data.q;
39 MBS_data.q = qrand;
40
41 [h,Jac] = mbs_calc_hJac(callfct,MBS_data);
42
43 Real_Rank = rank(Jac);                    % calcul du rang
   reel de Jac pour q random
44 [nl_JAC, nc_JAC] = size(Jac);
45
46 MBS_data.q = q_safe;
47
48 %  Proposition de partitionnement pour un u_des fourni (
   vide), (partiel,
49 %  plein, valide ou non)
50
```

```
51 ind_u_des_com = [ind_u_des ind_c];      % mise en commun des
       u_des et des c
52
53 ind = 1:MBS_data.nbody;
54 ind_vu = setdiff(ind,ind_u_des_com);     % variables (v et u
       ) a partitionner
55
56 if first_partitioning == 1                    % on ne
       perturbe les 0 que la première fois (car seconde =
       système fermé)
57     for i = 1:length(ind_vu)                  % perturbation
          random (10-5) des valeurs nulles des (v,u_non_des):
          éviter les singularités
58         if abs(MBS_data.q(ind_vu(i))) < 1e-6
59             MBS_data.q(ind_vu(i)) = 1e-5*rand(1);
60         end
61     end
62 end
63
64
65 %!!!!! On définit les options pour fminsearch ou lsqnonlin
       (optimset ou optimoptions respectivement !!!!!
66
67 %options = optimset('TolFun',1e-20,'TolX',1e-20,'MaxIter
       ',1000000,'MaxFunEvals',100000);
68 options = optimoptions(@lsqnonlin,'Algorithm','trust-region
       -reflective','FiniteDifferenceType','central','
       MaxFunctionEvaluations',1500,'FunctionTolerance',1e-40,'
       StepTolerance',1e-40,'MaxIterations',40000000000,'
       OptimalityTolerance',1e-40)
69
70 %!!!!! On définit les valeurs initiales du problème d'
       optimisation à résoudre, il s'agit des valeurs des
       contraintes dépendantes !!!!!
71 q0 = MBS_data.q(sort(MBS_data.qv));
72
73 %!!!!! On fait soit appel à fminsearch, soit à lsqnonlin en
       utilisant la fonction custom Func_h!!!!!
74 %q = fminsearch(@(q) Func_h(q,MBS_data),q0,options);
75 q = lsqnonlin(@(q) Func_h(q,MBS_data),q0,[],[],options);
```

```
76
77 %!!!!! On range les valeurs dépendantes obtenues après l'
       optimisation dans la structure MBS_data!!!!!
78 MBS_data.q(sort(MBS_data.qv))=q;
79
80 %!!!!! Vérification des résidus (doivent être proche de
       zéro, à décommenter si besoin!!!!!
81 %H = Func_h(q,MBS_data)
82
83
84 [h,Jac] = mbs_calc_hJac(callfct,MBS_data);
85 % Pivoted LU Factorisation
86
87 %!!!!! Calcul du rang post fermeture, ne marche pas
       toujours mais l'élimination a bien lieu dans mbs_lutot
       !!!!!
88 Real_Rank = rank(Jac);
89 %!!!!! Si le calcul du rang ne fonctionne pas correctement
       après fermeture des boucles, rentrer la valeur pour
       éviter erreur au prochain point!!!!!
90 Real_Rank = 12;
91
92 [Jv, irk, row_perm, col_perm] = mbs_lutot(Jac(:,ind_vu),
     IS_ROW_PERM);
93
94 % Check the user choice for independent coordinates
95
96 %!!!!! Normalement pas de problème, sauf si le calcul du
       vrai rang ne repère pas une déficience de rang, à
       vérifier manuellement!!!!!
97
98 if irk <Real_Rank   % choix mauvais - erreur
99     ind_u = NaN; ind_v = NaN; ind_cont = NaN; ind_red = NaN
           ;
100    disp('>>MBS>>  ');
101    disp('>>MBS>> mbs_coord_part: wrong choice for the
         independent variable set - Jv matrix singular');
102    disp('>>MBS>>  ');
103    err = -1;
104 else                  % choix valide
```

```
105       ind_aux = ind_vu(col_perm);
106       ind_v = ind_aux(1:irk);       % Liste definitive des v
107
108       if length(ind_aux) > irk      % additional independent
             variabes : les dernières
109           ind_u_new = ind_aux(irk+1:length(ind_aux));
110       else
111           ind_u_new = [];           % no additional
                 independent variable
112       end
113
114       ind_u = sort([ind_u_new ind_u_des]);
115
116       MBS_data.q(ind_u)=q_safe(ind_u);
117
118       if irk == nl_JAC                        % Case : no
             redundant constraints
119           ind_cont = row_perm;
120           ind_red = [];
121       else                                    % Case : redundant
             constraints
122           ind_cont = row_perm(1:irk);
123           ind_red = sort(row_perm(irk+1:nl_JAC));
124       end
125   end
```

Modifications: mbs_lutot.m

Changes from line 64 to line 106.

```
1   function [a, irk, row_perm, col_perm, ierr] = mbs_lutot(a,
       IS_ROW_PERM)
2   %MBS_LUTOT LU factorization with full pivoting of the
       constraint Jacobian
3   %matrix.
4   %
5   %   [A, IRK, ROW_PERM, COL_PERM, IERR] = MBS_LUTOT(A,
       IS_ROW_PERM)
6   %   a (input) : rectangular (nl,nc) matrix to be factorized
       :
7   %   a (output) : factorized square full-rank sub-matrix
8   %                - lower triangular part (diagonal
```

```
    excluded) : L matrix
9   %                 - Upper triangular part (diagonal
    included) : U matrix
10  %
11  %   irk (output) : Rank of the matrix (output)
12  %   ierr (output):  = -1 if a is singular (rank deficiency)
13  %                   = 0 if a is regular (full rank = min(nl
    , nc))
14  %   row_perm (output) : row permutation vector (length nl)
15  %   col_perm (output) : columns permutation vector (length
    nc)
16
17
18  %   See also
19
20  %   Copyright 2005 Universite catholique de Louvain
21  %   Tony Postiau & Paul Fisette - 2005
22  %
23  %   support@mbsyslab.be
24  %
25  %   Modifications
26  %   > 12/01/2006 : MD: en-tête
27  %
28  %---
29
30
31  [nl, nc] = size(a);
32
33  % Initialization
34
35  epslu = 1.0e-15;
36  irk = 0;
37  ipl = [];
38  ipc = [];
39
40  for iter = 1:nl
41
42  %   Maximum pivot search : element (il, ic)
43
44      if strcmp(IS_ROW_PERM,'yes')
```

```
45          ilmax = nl;
46      else
47          ilmax = iter;
48      end
49      p = 0.0;
50      for j = iter:nc
51          for i = iter:ilmax
52              z = a(i,j);
53              if abs(z) > abs(p)
54                  p = z;
55                  il = i;
56                  ic = j;
57              end
58          end
59      end
60
61  %  Case of a singular matrix (redundant constraints)
62
63      if abs(p) < epslu
64  %!!!!!The program should not return ierr = -1 anymore!!!!!
65          ierr = 0;
66  %!!!!!The returned Jv is the square matrix of size irk, i.e
        . iter-1!!!!!
67          a = a(1:irk,1:irk);
68          [row_perm] =  mbs_vec_perm([1:nl], ipl, irk);
69          [col_perm] =  mbs_vec_perm([1:nc], ipc, irk);
70          return;
71      end
72
73  %   Update of the rank and the permutation vectors
74
75      irk =iter;
76      ipl(iter) = il;
77      ipc(iter) = ic;
78
79  %   Permutation of rows iter and il
80
81      if il > iter
82          for j = 1:nc
83              z = a(iter,j);
```

```
84              a(iter,j) = a(il,j);
85              a(il,j) = z;
86          end
87      end
88
89  %   Permutation of columns iter and ic
90
91      if ic > iter
92          for i = 1:nl
93              z = a(i,iter);
94              a(i,iter) = a(i,ic);
95              a(i,ic) = z;
96          end
97      end
98
99  %   Non pathological case : the process ends at iter = nl
    or nc
100
101     if (iter == nl) | (iter == nc)
102         ierr = 0;
103         [row_perm] =  mbs_vec_perm([1:nl], ipl, irk);
104         [col_perm] =  mbs_vec_perm([1:nc], ipc, irk);
105 %!!!!!Return the square matrix Jv of size iter!!!!!
106         a = a(1:iter, 1: iter);
107         return;
108     end
109
110 %   Transformation of the sub-matrix
111
112     it1 = iter + 1;
113     for i = it1:nl
114         z = a(i,iter)/p;
115         a(i,iter) = z;
116         for j = it1:nc
117             a(i,j) = a(i,j) - z * a(iter,j);
118         end
119     end
120
121 end
```

Custom: Func_h.m

This Function is nothing more than the modified symbolic file mbs_cons_hJ_... .m in order to put it in lsqnonlin or fminsearch. This Func_h function is the one linked to the agile eye.

```
function [h] = Func_h( q,s )

%!!!!! Initialisation of the vector containing the
    constraints !!!!!
h = zeros();

%!!!!! Creation of a vector containing all the generalised
    coordinates (dependent, independent, forced) with their
    initial values !!!!!
D = s.q;

%!!!!! Replacing the value of all the dependent variables
    with the symbolic vector q !!!!!
D(sort(s.qv))=q;


%!!!!! Exactly the same as the mbs_cons_hj_ symbolic file,
    except the q have been replaced by D !!!!!
% Trigonometric Variables

 C1 = cos(D(1));
  S1 = sin(D(1));
  C2 = cos(D(2));
  S2 = sin(D(2));
  C3 = cos(D(3));
  S3 = sin(D(3));
  C4 = cos(D(4));
  S4 = sin(D(4));
  C5 = cos(D(5));
  S5 = sin(D(5));
  C6 = cos(D(6));
  S6 = sin(D(6));
  C7 = cos(D(7));
  S7 = sin(D(7));
```

```
30    C8 = cos(D(8));
31    S8 = sin(D(8));
32    C9 = cos(D(9));
33    S9 = sin(D(9));
34
35  % == Block_0_0_0_0_0_2 ==
36
37  % Trigonometric Variables
38
39    C13 = cos(D(13));
40    S13 = sin(D(13));
41    C14 = cos(D(14));
42    S14 = sin(D(14));
43    C15 = cos(D(15));
44    S15 = sin(D(15));
45
46  % == Block_0_0_0_0_0_3 ==
47
48  % Trigonometric Variables
49
50    C16 = cos(D(16));
51    S16 = sin(D(16));
52    C17 = cos(D(17));
53    S17 = sin(D(17));
54    C18 = cos(D(18));
55    S18 = sin(D(18));
56    C19 = cos(D(19));
57    S19 = sin(D(19));
58    C20 = cos(D(20));
59    S20 = sin(D(20));
60    C21 = cos(D(21));
61    S21 = sin(D(21));
62    C22 = cos(D(22));
63    S22 = sin(D(22));
64    C23 = cos(D(23));
65    S23 = sin(D(23));
66    C24 = cos(D(24));
67    S24 = sin(D(24));
68
69  % == Block_0_0_0_0_0_4 ==
```

```
70
71 % Trigonometric Variables
72
73     C25 = cos(D(25));
74     S25 = sin(D(25));
75     C26 = cos(D(26));
76     S26 = sin(D(26));
77     C27 = cos(D(27));
78     S27 = sin(D(27));
79     C28 = cos(D(28));
80     S28 = sin(D(28));
81     C29 = cos(D(29));
82     S29 = sin(D(29));
83     C30 = cos(D(30));
84     S30 = sin(D(30));
85     C31 = cos(D(31));
86     S31 = sin(D(31));
87     C32 = cos(D(32));
88     S32 = sin(D(32));
89     C33 = cos(D(33));
90     S33 = sin(D(33));
91
92 % == Block_0_1_0_0_0_1 ==
93
94 % Constraints and Constraints Jacobian
95
96 %
97     RO_1_12 = C1*C2;
98     RO_1_22 = S1*C2;
99     RO_1_72 = C1*S2;
100    RO_1_82 = S1*S2;
101    RO_1_43 = RO_1_72*S3-S1*C3;
102    RO_1_53 = RO_1_82*S3+C1*C3;
103    RO_1_63 = C2*S3;
104    RO_1_73 = RO_1_72*C3+S1*S3;
105    RO_1_83 = RO_1_82*C3-C1*S3;
106    RO_1_93 = C2*C3;
107    RO_1_14 = RO_1_12*C4-RO_1_73*S4;
108    RO_1_24 = RO_1_22*C4-RO_1_83*S4;
109    RO_1_34 = -(RO_1_93*S4+S2*C4);
```

```
RO_1_74 = RO_1_12*S4+RO_1_73*C4;
RO_1_84 = RO_1_22*S4+RO_1_83*C4;
RO_1_94 = RO_1_93*C4-S2*S4;
RO_1_45 = RO_1_43*C5+RO_1_74*S5;
RO_1_55 = RO_1_53*C5+RO_1_84*S5;
RO_1_65 = RO_1_63*C5+RO_1_94*S5;
RO_1_75 = -(RO_1_43*S5-RO_1_74*C5);
RO_1_85 = -(RO_1_53*S5-RO_1_84*C5);
RO_1_95 = -(RO_1_63*S5-RO_1_94*C5);
RO_1_16 = RO_1_14*C6+RO_1_45*S6;
RO_1_26 = RO_1_24*C6+RO_1_55*S6;
RO_1_36 = RO_1_34*C6+RO_1_65*S6;
RO_1_46 = -(RO_1_14*S6-RO_1_45*C6);
RO_1_56 = -(RO_1_24*S6-RO_1_55*C6);
RO_1_66 = -(RO_1_34*S6-RO_1_65*C6);
RO_1_17 = RO_1_16*C7-RO_1_75*S7;
RO_1_27 = RO_1_26*C7-RO_1_85*S7;
RO_1_37 = RO_1_36*C7-RO_1_95*S7;
RO_1_77 = RO_1_16*S7+RO_1_75*C7;
RO_1_87 = RO_1_26*S7+RO_1_85*C7;
RO_1_97 = RO_1_36*S7+RO_1_95*C7;
RO_1_48 = RO_1_46*C8+RO_1_77*S8;
RO_1_58 = RO_1_56*C8+RO_1_87*S8;
RO_1_68 = RO_1_66*C8+RO_1_97*S8;
RO_1_78 = -(RO_1_46*S8-RO_1_77*C8);
RO_1_88 = -(RO_1_56*S8-RO_1_87*C8);
RO_1_98 = -(RO_1_66*S8-RO_1_97*C8);
RL_1_14 = RO_1_12*s.dpt(1,3)+RO_1_73*s.dpt(3,3);
RL_1_24 = RO_1_22*s.dpt(1,3)+RO_1_83*s.dpt(3,3);
RL_1_34 = RO_1_93*s.dpt(3,3)-s.dpt(1,3)*S2;
RL_1_16 = RO_1_14*s.dpt(1,4)+RO_1_45*s.dpt(2,4);
RL_1_26 = RO_1_24*s.dpt(1,4)+RO_1_55*s.dpt(2,4);
RL_1_36 = RO_1_34*s.dpt(1,4)+RO_1_65*s.dpt(2,4);
RL_1_18 = RO_1_46*s.dpt(2,5);
RL_1_28 = RO_1_56*s.dpt(2,5);
RL_1_38 = RO_1_66*s.dpt(2,5);
JT_1_18_1 = -(RL_1_24+RL_1_26+RL_1_28);
JT_1_28_1 = RL_1_14+RL_1_16+RL_1_18;
JT_1_18_2 = C1*(RL_1_34+RL_1_36+RL_1_38);
JT_1_28_2 = S1*(RL_1_34+RL_1_36+RL_1_38);
```

```
150     JT_1_38_2 = -(RL_1_28*S1+C1*(RL_1_14+RL_1_16+RL_1_18)+S1
            *(RL_1_24+RL_1_26));
151     JT_1_18_3 = RL_1_28*S2+RL_1_38*RO_1_22+RO_1_22*(RL_1_34+
            RL_1_36)+S2*(RL_1_24+RL_1_26);
152     JT_1_28_3 = -(RL_1_18*S2+RL_1_38*RO_1_12+RO_1_12*(RL_1_34
            +RL_1_36)+S2*(RL_1_14+RL_1_16));
153     JT_1_38_3 = RO_1_12*(RL_1_24+RL_1_26)-RO_1_22*(RL_1_14+
            RL_1_16)-RL_1_18*RO_1_22+RL_1_28*RO_1_12;
154     JT_1_18_4 = RO_1_53*(RL_1_36+RL_1_38)-RO_1_63*(RL_1_26+
            RL_1_28);
155     JT_1_28_4 = -(RO_1_43*(RL_1_36+RL_1_38)-RO_1_63*(RL_1_16+
            RL_1_18));
156     JT_1_38_4 = RO_1_43*(RL_1_26+RL_1_28)-RO_1_53*(RL_1_16+
            RL_1_18);
157     JT_1_18_5 = RO_1_24*(RL_1_36+RL_1_38)-RO_1_34*(RL_1_26+
            RL_1_28);
158     JT_1_28_5 = -(RO_1_14*(RL_1_36+RL_1_38)-RO_1_34*(RL_1_16+
            RL_1_18));
159     JT_1_38_5 = RO_1_14*(RL_1_26+RL_1_28)-RO_1_24*(RL_1_16+
            RL_1_18);
160     JT_1_18_6 = -(RL_1_28*RO_1_95-RL_1_38*RO_1_85);
161     JT_1_28_6 = RL_1_18*RO_1_95-RL_1_38*RO_1_75;
162     JT_1_38_6 = -(RL_1_18*RO_1_85-RL_1_28*RO_1_75);
163     JT_1_18_7 = -(RL_1_28*RO_1_66-RL_1_38*RO_1_56);
164     JT_1_28_7 = RL_1_18*RO_1_66-RL_1_38*RO_1_46;
165     JT_1_38_7 = -(RL_1_18*RO_1_56-RL_1_28*RO_1_46);
166
167 %==Block_0_1_0_0_0_2==
168
169 %Constraints and Constraints Jacobian
170
171 %
172     RO_0_214 = S13*S14;
173     RO_0_314 = -C13*S14;
174     RO_0_814 = -S13*C14;
175     RO_0_914 = C13*C14;
176     RO_0_115 = C14*C15;
177     RO_0_215 = RO_0_214*C15+C13*S15;
178     RO_0_315 = RO_0_314*C15+S13*S15;
179     RO_0_415 = -C14*S15;
```

```
    RO_0_515 = -(RO_0_214*S15-C13*C15);
    RO_0_615 = -(RO_0_314*S15-S13*C15);

% == Block_0_1_0_0_0_3 ==

% Constraints and Constraints Jacobian

%
    RO_3_117 = C16*C17;
    RO_3_217 = S16*C17;
    RO_3_717 = C16*S17;
    RO_3_817 = S16*S17;
    RO_3_418 = RO_3_717*S18-S16*C18;
    RO_3_518 = RO_3_817*S18+C16*C18;
    RO_3_618 = C17*S18;
    RO_3_718 = RO_3_717*C18+S16*S18;
    RO_3_818 = RO_3_817*C18-C16*S18;
    RO_3_918 = C17*C18;
    RO_3_119 = RO_3_117*C19-RO_3_718*S19;
    RO_3_219 = RO_3_217*C19-RO_3_818*S19;
    RO_3_319 = -(RO_3_918*S19+S17*C19);
    RO_3_719 = RO_3_117*S19+RO_3_718*C19;
    RO_3_819 = RO_3_217*S19+RO_3_818*C19;
    RO_3_919 = RO_3_918*C19-S17*S19;
    RO_3_420 = RO_3_418*C20+RO_3_719*S20;
    RO_3_520 = RO_3_518*C20+RO_3_819*S20;
    RO_3_620 = RO_3_618*C20+RO_3_919*S20;
    RO_3_720 = -(RO_3_418*S20-RO_3_719*C20);
    RO_3_820 = -(RO_3_518*S20-RO_3_819*C20);
    RO_3_920 = -(RO_3_618*S20-RO_3_919*C20);
    RO_3_121 = RO_3_119*C21+RO_3_420*S21;
    RO_3_221 = RO_3_219*C21+RO_3_520*S21;
    RO_3_321 = RO_3_319*C21+RO_3_620*S21;
    RO_3_421 = -(RO_3_119*S21-RO_3_420*C21);
    RO_3_521 = -(RO_3_219*S21-RO_3_520*C21);
    RO_3_621 = -(RO_3_319*S21-RO_3_620*C21);
    RO_3_122 = RO_3_121*C22-RO_3_720*S22;
    RO_3_222 = RO_3_221*C22-RO_3_820*S22;
    RO_3_322 = RO_3_321*C22-RO_3_920*S22;
    RO_3_722 = RO_3_121*S22+RO_3_720*C22;
```

```
220  RO_3_822 = RO_3_221*S22+RO_3_820*C22;
221  RO_3_922 = RO_3_321*S22+RO_3_920*C22;
222  RO_3_423 = RO_3_421*C23+RO_3_722*S23;
223  RO_3_523 = RO_3_521*C23+RO_3_822*S23;
224  RO_3_623 = RO_3_621*C23+RO_3_922*S23;
225  RO_3_723 = -(RO_3_421*S23-RO_3_722*C23);
226  RO_3_823 = -(RO_3_521*S23-RO_3_822*C23);
227  RO_3_923 = -(RO_3_621*S23-RO_3_922*C23);
228  RL_3_119 = RO_3_117*s.dpt(1,6)+RO_3_718*s.dpt(3,6);
229  RL_3_219 = RO_3_217*s.dpt(1,6)+RO_3_818*s.dpt(3,6);
230  RL_3_319 = RO_3_918*s.dpt(3,6)-s.dpt(1,6)*S17;
231  RL_3_121 = RO_3_119*s.dpt(1,7)+RO_3_420*s.dpt(2,7);
232  RL_3_221 = RO_3_219*s.dpt(1,7)+RO_3_520*s.dpt(2,7);
233  RL_3_321 = RO_3_319*s.dpt(1,7)+RO_3_620*s.dpt(2,7);
234  RL_3_123 = RO_3_421*s.dpt(2,8);
235  RL_3_223 = RO_3_521*s.dpt(2,8);
236  RL_3_323 = RO_3_621*s.dpt(2,8);
237  JT_3_123_16 = -(RL_3_219+RL_3_221+RL_3_223);
238  JT_3_223_16 = RL_3_119+RL_3_121+RL_3_123;
239  JT_3_123_17 = C16*(RL_3_319+RL_3_321+RL_3_323);
240  JT_3_223_17 = S16*(RL_3_319+RL_3_321+RL_3_323);
241  JT_3_323_17 = -(RL_3_223*S16+C16*(RL_3_119+RL_3_121+
         RL_3_123)+S16*(RL_3_219+RL_3_221));
242  JT_3_123_18 = RL_3_223*S17+RL_3_323*RO_3_217+RO_3_217*(
         RL_3_319+RL_3_321)+S17*(RL_3_219+RL_3_221);
243  JT_3_223_18 = -(RL_3_123*S17+RL_3_323*RO_3_117+RO_3_117*(
         RL_3_319+RL_3_321)+S17*(RL_3_119+RL_3_121));
244  JT_3_323_18 = RO_3_117*(RL_3_219+RL_3_221)-RO_3_217*(
         RL_3_119+RL_3_121)-RL_3_123*RO_3_217+RL_3_223*RO_3_117
         ;
245  JT_3_123_19 = RO_3_518*(RL_3_321+RL_3_323)-RO_3_618*(
         RL_3_221+RL_3_223);
246  JT_3_223_19 = -(RO_3_418*(RL_3_321+RL_3_323)-RO_3_618*(
         RL_3_121+RL_3_123));
247  JT_3_323_19 = RO_3_418*(RL_3_221+RL_3_223)-RO_3_518*(
         RL_3_121+RL_3_123);
248  JT_3_123_20 = RO_3_219*(RL_3_321+RL_3_323)-RO_3_319*(
         RL_3_221+RL_3_223);
249  JT_3_223_20 = -(RO_3_119*(RL_3_321+RL_3_323)-RO_3_319*(
         RL_3_121+RL_3_123));
```

```
250     JT_3_323_20 = RO_3_119*(RL_3_221+RL_3_223)-RO_3_219*(
            RL_3_121+RL_3_123);
251     JT_3_123_21 = -(RL_3_223*RO_3_920-RL_3_323*RO_3_820);
252     JT_3_223_21 = RL_3_123*RO_3_920-RL_3_323*RO_3_720;
253     JT_3_323_21 = -(RL_3_123*RO_3_820-RL_3_223*RO_3_720);
254     JT_3_123_22 = -(RL_3_223*RO_3_621-RL_3_323*RO_3_521);
255     JT_3_223_22 = RL_3_123*RO_3_621-RL_3_323*RO_3_421;
256     JT_3_323_22 = -(RL_3_123*RO_3_521-RL_3_223*RO_3_421);
257
258 %% Block_0_1_0_0_0_4 %%
259
260 % Constraints and Constraints Jacobian
261
262 %
263     RO_5_126 = C25*C26;
264     RO_5_226 = S25*C26;
265     RO_5_726 = C25*S26;
266     RO_5_826 = S25*S26;
267     RO_5_427 = RO_5_726*S27-S25*C27;
268     RO_5_527 = RO_5_826*S27+C25*C27;
269     RO_5_627 = C26*S27;
270     RO_5_727 = RO_5_726*C27+S25*S27;
271     RO_5_827 = RO_5_826*C27-C25*S27;
272     RO_5_927 = C26*C27;
273     RO_5_128 = RO_5_126*C28-RO_5_727*S28;
274     RO_5_228 = RO_5_226*C28-RO_5_827*S28;
275     RO_5_328 = -(RO_5_927*S28+S26*C28);
276     RO_5_728 = RO_5_126*S28+RO_5_727*C28;
277     RO_5_828 = RO_5_226*S28+RO_5_827*C28;
278     RO_5_928 = RO_5_927*C28-S26*S28;
279     RO_5_429 = RO_5_427*C29+RO_5_728*S29;
280     RO_5_529 = RO_5_527*C29+RO_5_828*S29;
281     RO_5_629 = RO_5_627*C29+RO_5_928*S29;
282     RO_5_729 = -(RO_5_427*S29-RO_5_728*C29);
283     RO_5_829 = -(RO_5_527*S29-RO_5_828*C29);
284     RO_5_929 = -(RO_5_627*S29-RO_5_928*C29);
285     RO_5_130 = RO_5_128*C30+RO_5_429*S30;
286     RO_5_230 = RO_5_228*C30+RO_5_529*S30;
287     RO_5_330 = RO_5_328*C30+RO_5_629*S30;
288     RO_5_430 = -(RO_5_128*S30-RO_5_429*C30);
```

```
RO_5_530 = -(RO_5_228*S30-RO_5_529*C30);
RO_5_630 = -(RO_5_328*S30-RO_5_629*C30);
RO_5_131 = RO_5_130*C31-RO_5_729*S31;
RO_5_231 = RO_5_230*C31-RO_5_829*S31;
RO_5_331 = RO_5_330*C31-RO_5_929*S31;
RO_5_731 = RO_5_130*S31+RO_5_729*C31;
RO_5_831 = RO_5_230*S31+RO_5_829*C31;
RO_5_931 = RO_5_330*S31+RO_5_929*C31;
RO_5_432 = RO_5_430*C32+RO_5_731*S32;
RO_5_532 = RO_5_530*C32+RO_5_831*S32;
RO_5_632 = RO_5_630*C32+RO_5_931*S32;
RO_5_732 = -(RO_5_430*S32-RO_5_731*C32);
RO_5_832 = -(RO_5_530*S32-RO_5_831*C32);
RO_5_932 = -(RO_5_630*S32-RO_5_931*C32);
RL_5_128 = RO_5_126*s.dpt(1,9)+RO_5_727*s.dpt(3,9);
RL_5_228 = RO_5_226*s.dpt(1,9)+RO_5_827*s.dpt(3,9);
RL_5_328 = RO_5_927*s.dpt(3,9)-s.dpt(1,9)*S26;
RL_5_130 = RO_5_128*s.dpt(1,10)+RO_5_429*s.dpt(2,10);
RL_5_230 = RO_5_228*s.dpt(1,10)+RO_5_529*s.dpt(2,10);
RL_5_330 = RO_5_328*s.dpt(1,10)+RO_5_629*s.dpt(2,10);
RL_5_132 = RO_5_430*s.dpt(2,11);
RL_5_232 = RO_5_530*s.dpt(2,11);
RL_5_332 = RO_5_630*s.dpt(2,11);
JT_5_132_25 = -(RL_5_228+RL_5_230+RL_5_232);
JT_5_232_25 = RL_5_128+RL_5_130+RL_5_132;
JT_5_132_26 = C25*(RL_5_328+RL_5_330+RL_5_332);
JT_5_232_26 = S25*(RL_5_328+RL_5_330+RL_5_332);
JT_5_332_26 = -(RL_5_232*S25+C25*(RL_5_128+RL_5_130+
    RL_5_132)+S25*(RL_5_228+RL_5_230));
JT_5_132_27 = RL_5_232*S26+RL_5_332*RO_5_226+RO_5_226*(
    RL_5_328+RL_5_330)+S26*(RL_5_228+RL_5_230);
JT_5_232_27 = -(RL_5_132*S26+RL_5_332*RO_5_126+RO_5_126*(
    RL_5_328+RL_5_330)+S26*(RL_5_128+RL_5_130));
JT_5_332_27 = RO_5_126*(RL_5_228+RL_5_230)-RO_5_226*(
    RL_5_128+RL_5_130)-RL_5_132*RO_5_226+RL_5_232*RO_5_126
    ;
JT_5_132_28 = RO_5_527*(RL_5_330+RL_5_332)-RO_5_627*(
    RL_5_230+RL_5_232);
JT_5_232_28 = -(RO_5_427*(RL_5_330+RL_5_332)-RO_5_627*(
    RL_5_130+RL_5_132));
```

```
322     JT_5_332_28 = RO_5_427*(RL_5_230+RL_5_232)-RO_5_527*(
            RL_5_130+RL_5_132);
323     JT_5_132_29 = RO_5_228*(RL_5_330+RL_5_332)-RO_5_328*(
            RL_5_230+RL_5_232);
324     JT_5_232_29 = -(RO_5_128*(RL_5_330+RL_5_332)-RO_5_328*(
            RL_5_130+RL_5_132));
325     JT_5_332_29 = RO_5_128*(RL_5_230+RL_5_232)-RO_5_228*(
            RL_5_130+RL_5_132);
326     JT_5_132_30 = -(RL_5_232*RO_5_929-RL_5_332*RO_5_829);
327     JT_5_232_30 = RL_5_132*RO_5_929-RL_5_332*RO_5_729;
328     JT_5_332_30 = -(RL_5_132*RO_5_829-RL_5_232*RO_5_729);
329     JT_5_132_31 = -(RL_5_232*RO_5_630-RL_5_332*RO_5_530);
330     JT_5_232_31 = RL_5_132*RO_5_630-RL_5_332*RO_5_430;
331     JT_5_332_31 = -(RL_5_132*RO_5_530-RL_5_232*RO_5_430);
332
333 % == Block_0_1_0_0_1_0 ==
334
335 % Constraints and Constraints Jacobian
336
337 %
338     h_1 = -(RL_1_14+RL_1_16+RL_1_18-D(10));
339     h_2 = -(RL_1_24+RL_1_26+RL_1_28-D(11));
340     h_3 = -(RL_1_34+RL_1_36+RL_1_38-D(12));
341     h_4 = RO_0_315*(RO_1_27*C9+RO_1_58*S9)-RO_0_615*(RO_1_27*
            S9-RO_1_58*C9)+RO_0_914*RO_1_88;
342     h_5 = RO_0_115*(RO_1_37*C9+RO_1_68*S9)-RO_0_415*(RO_1_37*
            S9-RO_1_68*C9)+RO_1_98*S14;
343     h_6 = RO_0_215*(RO_1_17*C9+RO_1_48*S9)-RO_0_515*(RO_1_17*
            S9-RO_1_48*C9)+RO_0_814*RO_1_78;
344 %
345     h_7 = -(RL_3_119+RL_3_121+RL_3_123-D(10)+s.dpt(1,1));
346     h_8 = -(RL_3_219+RL_3_221+RL_3_223-D(11));
347     h_9 = -(RL_3_319+RL_3_321+RL_3_323-D(12));
348     h_10 = RO_0_315*(RO_3_222*C24+RO_3_523*S24)-RO_0_615*(
            RO_3_222*S24-RO_3_523*C24)+RO_0_914*RO_3_823;
349     h_11 = RO_0_115*(RO_3_322*C24+RO_3_623*S24)-RO_0_415*(
            RO_3_322*S24-RO_3_623*C24)+RO_3_923*S14;
350     h_12 = RO_0_215*(RO_3_122*C24+RO_3_423*S24)-RO_0_515*(
            RO_3_122*S24-RO_3_423*C24)+RO_0_814*RO_3_723;
351 %
```

```
352     h_13 = -(RL_5_128+RL_5_130+RL_5_132-D(10)+s.dpt(1,2));
353     h_14 = -(RL_5_228+RL_5_230+RL_5_232-D(11)+s.dpt(2,2));
354     h_15 = -(RL_5_328+RL_5_330+RL_5_332-D(12));
355     h_16 = RO_0_315*(RO_5_231*C33+RO_5_532*S33)-RO_0_615*(
            RO_5_231*S33-RO_5_532*C33)+RO_0_914*RO_5_832;
356     h_17 = RO_0_115*(RO_5_331*C33+RO_5_632*S33)-RO_0_415*(
            RO_5_331*S33-RO_5_632*C33)+RO_5_932*S14;
357     h_18 = RO_0_215*(RO_5_131*C33+RO_5_432*S33)-RO_0_515*(
            RO_5_131*S33-RO_5_432*C33)+RO_0_814*RO_5_732;
358
359 % == Block_0_3_0_0_0_0 ==
360
361 % Symbolic Outputs
362
363     h(1) = h_1;
364     h(2) = h_2;
365     h(3) = h_3;
366     h(4) = h_4;
367     h(5) = h_5;
368     h(6) = h_6;
369     h(7) = h_7;
370     h(8) = h_8;
371     h(9) = h_9;
372     h(10) = h_10;
373     h(11) = h_11;
374     h(12) = h_12;
375     h(13) = h_13;
376     h(14) = h_14;
377     h(15) = h_15;
378     h(16) = h_16;
379     h(17) = h_17;
380     h(18) = h_18;
381     Jac(1,1) = -JT_1_18_1;
382     Jac(1,2) = -JT_1_18_2;
383     Jac(1,3) = -JT_1_18_3;
384     Jac(1,4) = -JT_1_18_4;
385     Jac(1,5) = -JT_1_18_5;
386     Jac(1,6) = -JT_1_18_6;
387     Jac(1,7) = -JT_1_18_7;
388     Jac(1,10) = (1.0);
```

```
389     Jac(2,1) = -JT_1_28_1;
390     Jac(2,2) = -JT_1_28_2;
391     Jac(2,3) = -JT_1_28_3;
392     Jac(2,4) = -JT_1_28_4;
393     Jac(2,5) = -JT_1_28_5;
394     Jac(2,6) = -JT_1_28_6;
395     Jac(2,7) = -JT_1_28_7;
396     Jac(2,11) = (1.0);
397     Jac(3,2) = -JT_1_38_2;
398     Jac(3,3) = -JT_1_38_3;
399     Jac(3,4) = -JT_1_38_4;
400     Jac(3,5) = -JT_1_38_5;
401     Jac(3,6) = -JT_1_38_6;
402     Jac(3,7) = -JT_1_38_7;
403     Jac(3,12) = (1.0);
404     Jac(4,2) = S1;
405     Jac(4,3) = -RO_1_12;
406     Jac(4,4) = -RO_1_43;
407     Jac(4,5) = -RO_1_14;
408     Jac(4,6) = -RO_1_75;
409     Jac(4,7) = -RO_1_46;
410     Jac(4,8) = -RO_1_17;
411     Jac(4,9) = -RO_1_78;
412     Jac(4,13) = (1.0);
413     Jac(4,15) = S14;
414     Jac(5,2) = -C1;
415     Jac(5,3) = -RO_1_22;
416     Jac(5,4) = -RO_1_53;
417     Jac(5,5) = -RO_1_24;
418     Jac(5,6) = -RO_1_85;
419     Jac(5,7) = -RO_1_56;
420     Jac(5,8) = -RO_1_27;
421     Jac(5,9) = -RO_1_88;
422     Jac(5,14) = C13;
423     Jac(5,15) = RO_0_814;
424     Jac(6,1) = -(1.0);
425     Jac(6,3) = S2;
426     Jac(6,4) = -RO_1_63;
427     Jac(6,5) = -RO_1_34;
428     Jac(6,6) = -RO_1_95;
```

```
429     Jac(6,7) = -RO_1_66;
430     Jac(6,8) = -RO_1_37;
431     Jac(6,9) = -RO_1_98;
432     Jac(6,14) = S13;
433     Jac(6,15) = RO_0_914;
434     Jac(7,10) = (1.0);
435     Jac(7,16) = -JT_3_123_16;
436     Jac(7,17) = -JT_3_123_17;
437     Jac(7,18) = -JT_3_123_18;
438     Jac(7,19) = -JT_3_123_19;
439     Jac(7,20) = -JT_3_123_20;
440     Jac(7,21) = -JT_3_123_21;
441     Jac(7,22) = -JT_3_123_22;
442     Jac(8,11) = (1.0);
443     Jac(8,16) = -JT_3_223_16;
444     Jac(8,17) = -JT_3_223_17;
445     Jac(8,18) = -JT_3_223_18;
446     Jac(8,19) = -JT_3_223_19;
447     Jac(8,20) = -JT_3_223_20;
448     Jac(8,21) = -JT_3_223_21;
449     Jac(8,22) = -JT_3_223_22;
450     Jac(9,12) = (1.0);
451     Jac(9,17) = -JT_3_323_17;
452     Jac(9,18) = -JT_3_323_18;
453     Jac(9,19) = -JT_3_323_19;
454     Jac(9,20) = -JT_3_323_20;
455     Jac(9,21) = -JT_3_323_21;
456     Jac(9,22) = -JT_3_323_22;
457     Jac(10,13) = (1.0);
458     Jac(10,15) = S14;
459     Jac(10,17) = S16;
460     Jac(10,18) = -RO_3_117;
461     Jac(10,19) = -RO_3_418;
462     Jac(10,20) = -RO_3_119;
463     Jac(10,21) = -RO_3_720;
464     Jac(10,22) = -RO_3_421;
465     Jac(10,23) = -RO_3_122;
466     Jac(10,24) = -RO_3_723;
467     Jac(11,14) = C13;
468     Jac(11,15) = RO_0_814;
```

```
469     Jac(11,17) = -C16;
470     Jac(11,18) = -RO_3_217;
471     Jac(11,19) = -RO_3_518;
472     Jac(11,20) = -RO_3_219;
473     Jac(11,21) = -RO_3_820;
474     Jac(11,22) = -RO_3_521;
475     Jac(11,23) = -RO_3_222;
476     Jac(11,24) = -RO_3_823;
477     Jac(12,14) = S13;
478     Jac(12,15) = RO_0_914;
479     Jac(12,16) = -(1.0);
480     Jac(12,18) = S17;
481     Jac(12,19) = -RO_3_618;
482     Jac(12,20) = -RO_3_319;
483     Jac(12,21) = -RO_3_920;
484     Jac(12,22) = -RO_3_621;
485     Jac(12,23) = -RO_3_322;
486     Jac(12,24) = -RO_3_923;
487     Jac(13,10) = (1.0);
488     Jac(13,25) = -JT_5_132_25;
489     Jac(13,26) = -JT_5_132_26;
490     Jac(13,27) = -JT_5_132_27;
491     Jac(13,28) = -JT_5_132_28;
492     Jac(13,29) = -JT_5_132_29;
493     Jac(13,30) = -JT_5_132_30;
494     Jac(13,31) = -JT_5_132_31;
495     Jac(14,11) = (1.0);
496     Jac(14,25) = -JT_5_232_25;
497     Jac(14,26) = -JT_5_232_26;
498     Jac(14,27) = -JT_5_232_27;
499     Jac(14,28) = -JT_5_232_28;
500     Jac(14,29) = -JT_5_232_29;
501     Jac(14,30) = -JT_5_232_30;
502     Jac(14,31) = -JT_5_232_31;
503     Jac(15,12) = (1.0);
504     Jac(15,26) = -JT_5_332_26;
505     Jac(15,27) = -JT_5_332_27;
506     Jac(15,28) = -JT_5_332_28;
507     Jac(15,29) = -JT_5_332_29;
508     Jac(15,30) = -JT_5_332_30;
```

```
509     Jac(15,31) = -JT_5_332_31;
510     Jac(16,13) = (1.0);
511     Jac(16,15) = S14;
512     Jac(16,26) = S25;
513     Jac(16,27) = -RO_5_126;
514     Jac(16,28) = -RO_5_427;
515     Jac(16,29) = -RO_5_128;
516     Jac(16,30) = -RO_5_729;
517     Jac(16,31) = -RO_5_430;
518     Jac(16,32) = -RO_5_131;
519     Jac(16,33) = -RO_5_732;
520     Jac(17,14) = C13;
521     Jac(17,15) = RO_0_814;
522     Jac(17,26) = -C25;
523     Jac(17,27) = -RO_5_226;
524     Jac(17,28) = -RO_5_527;
525     Jac(17,29) = -RO_5_228;
526     Jac(17,30) = -RO_5_829;
527     Jac(17,31) = -RO_5_530;
528     Jac(17,32) = -RO_5_231;
529     Jac(17,33) = -RO_5_832;
530     Jac(18,14) = S13;
531     Jac(18,15) = RO_0_914;
532     Jac(18,25) = -(1.0);
533     Jac(18,27) = S26;
534     Jac(18,28) = -RO_5_627;
535     Jac(18,29) = -RO_5_328;
536     Jac(18,30) = -RO_5_929;
537     Jac(18,31) = -RO_5_630;
538     Jac(18,32) = -RO_5_331;
539     Jac(18,33) = -RO_5_932;
540
541
542
543 %!!!!! If the user wants to use fminsearch instead of
        lsqnonlin, uncomment the next line to create the correct
        objective function!!!!!
544  % h = sum(h.^2);
545
546 % ======= END Task 0 ======
```