Read Me

October 18, 2010

1 Introduction

This software is a general method for kinematic analysis of complex gear mechanisms. Its objectives are :

- to compute the rotational speed of all the links of the mechanism,
- to compute the contact frequencies in gear pairs and turning pairs.

The first step is to fill a `.m data` file following a protocol that will be explained further next. Then it is possible to build a main file calling the different functions of the software available.

The first section describes the function of each file available and how to use them. The second section gives information on how to build the input `data` file and the `main file`. Finally, the third section is dedicated to examples so as to help getting started.

2 Software description

2.1 Functions description

This section describes the different files composing the software.

- `jointtable.m` : this function takes as input the structure of `data` and computes the table of links and joints of the mechanism.
- `kinemod.m` : this function also takes the structure of `data` as input. It computes the adjacency matrix. To do so, this function calls two others :

  - `multiturning.m` : it is the first step in the determination of the reference body of each gear pair. It creates a list of all the turning or multi-turning pairs and of all the bodies belonging to each of them.
  - `bodyref.m` : it takes the structure of `data` and the two links of a gear pair as input, and gives the reference link for the gear pair chosen.

- `default.m` : it takes as input the Speed Ratio Vector of the system and the structure input data. It gives back a structure describing all the default frequencies that can be found in the system. Each cell of that structure represents a contact default and contains five fields :
  - ◇ rate : contact frequency,
  - ◇ N : number of sources that can create this default,
  - ◇ joint : indices of the links involved in the pair,
  - ◇ type : type of the joint that bears the defects,
  - ◇ doc : brief description of the defect.

2.2 Getting started

The input array of structures `data` must contain all information available about the mechanism, i.e. about all the pairs (gear pairs or turning pairs) constituing the system. It must be filled following a strict protocole. The element (i, j) of `data` gives all the needed information about the pair (i, j) of the mechanism. Due to symmetry, `data(i,j)` must be filled only for $j > i$.

It is important to understand, that the needed information is not the same for gear pairs and turning pairs. It means that the fields of `data` filled for a turning pair will not be the same as for a gear pair.

Turning pairs First of all, it is necessary to describe the roll or ball bearings of the mechanism involved in the turning pairs. It takes the form of a structured array `bearing` created as follow :

- bearing.Dm : Mean diameter of the ball bearing
- bearing.db : Diameter of the balls or rolls

- bearing.Z : Number of balls or rolls
- bearing.angle : Angle of contact between the balls and the ring of the bearing

Then, this structure `bearing` is used to describe the turning pair $data(i, j)$, the element (i, j) of the structured array `data` as follow :

- `data(i,j).type` : type of the pair : here 'p' for turning pair
- $data(i, j).N$: number of ball bearings engaged in the pair
- $data(i, j).B$: name of each of the ball-bearing structure array
- $data(i, j).level$: level of the turning pair. The level of a turning pair is an integer. Turning pairs which are coaxial must have the same level.

If information on the roll or ball bearing is not available, it is still possible to compute the program. In that case, the type of joint must be given as $'l'$:

- $data(i, j).type =' l'$,
- $data(i, j).level$: level of the turning pair.

Gear pairs There are three fields to fill in the element (i, j) of the structure data for a gear pair :

- $data(i, j).type =' e'$: type of the pair : here 'e' for the gear pairs
- $data(i, j).Z1 = N_i e^{\theta_i \sqrt{-1}}$: complex number of teeth link i, where N_i is the number of teeth on link i and θ_i is the angle between the axis of rotation and the axis of the teeth of link i
- $data(i, j).Z2 = N_j e^{\theta_j \sqrt{-1}}$: complex number of teeth link j, where N_j is the number of teeth on link j and θ_j is the angle between the axis of rotation and the axis of the teeth of link j

3 Examples

In this section, three examples are given so as to illustrate how to use this software. The first one is a very simple system : an epicyclic gear train with a bevel planet. The second one, is a car differential and the last one is the main gear box of an helicopter.

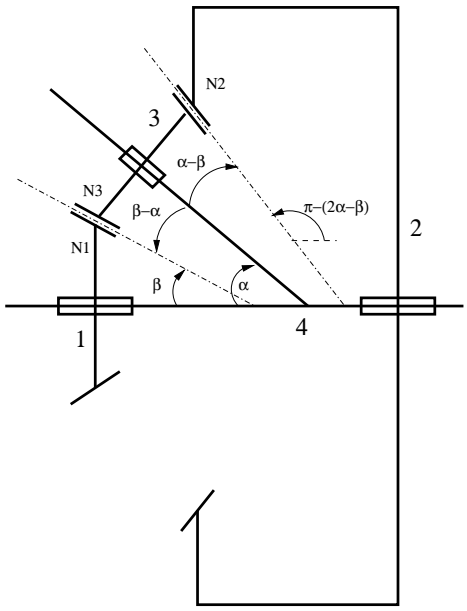

Figure 1: Simple epicyclic gear train

3.1 Example 1 : the simple epicyclic gear train

The Figure (1) gives the representation of the simple epicyclic gear train. The information on the bearings is not available in that example. The field `type` for the turning pairs will then be : 'l'. For the gear pairs $N_1 = 15$, $N_2 = 25$, and $N_3 = 10$, where N_1, N_2 and N_3 are the number of teeth of the links 1, 2 and 3. The angles between the axis of rotation and the axis of teeth of each link are given in the Figure (1). The file `dataexemple.m` gives an example of how to build the data file :

```
 clear all

%% Roll or Ball bearings
data(1,4).type='l';
data(1,4).level=1;

data(2,4).type='l';
data(2,4).level=1;

data(3,4).type='l';
```

```
data(3,4).level=2;

%% Description of the gear pairs of the mechanism
beta=pi/4;
alpha=pi/3;

data(1,3).type='e';
data(1,3).Z1=15*exp(j*beta);
data(1,3).Z2=10*exp(j*(beta-alpha));


data(2,3).type='e';
data(2,3).Z1=25*exp(j*(2*alpha-beta-pi));
data(2,3).Z2=10*exp(j*(alpha-beta));
```

Here is given MATLAB session to analyse this mechanism.

```
>> %% Collecting the data
data_exemple

>> % Computing the adjacency matrix
M=kinemod(data)

M =

  10.6066 +10.6066i        0              9.6593 - 2.5882i -20.2659 - 8.0184i
        0            -6.4705 -24.1481i   9.6593 + 2.5882i  -3.1888 +21.5600i

>> % Number of degree of freedom
size(M,2)-rank(M)

ans =

     2

>> % Computing the speed ratio matrix
Rep=null(M)
```

```
Rep =

  -0.1944 + 0.2990i   0.7267 - 0.0512i
   0.3186 - 0.2606i   0.3580 + 0.2626i
   0.8211 + 0.1035i   0.0687 - 0.0073i
   0.1262 - 0.0508i   0.4963 + 0.1449i

>> % Computing the Speed Ratio Matrix and imposing the link cart speed to 0
>> null([M;0 0 0 1])

ans =

   0.4700 - 0.2370i
  -0.2820 + 0.1422i
  -0.6603 - 0.4328i
        0

>> % Normalizing with respect to link 1 Speed
>> W=null([M;0 0 0 1])/ans(1)

W =

   1.0000
  -0.6000 + 0.0000i
  -0.7500 - 1.2990i
        0

>> % Computing all the contact defaults of the mechanism
>> def=default(data,W)

def =

6x1 struct array with fields:
    rate
    N
    joint
    type
    doc
```

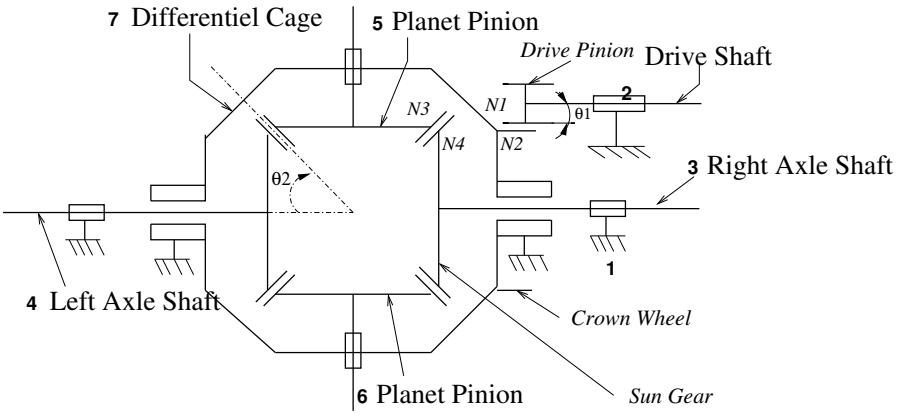

Figure 2: Car differential

```
>> % Displaying the second default
>> def(2)

ans =

     rate: 0.5263
        N: 15.0000
    joint: [1 3]
     type: 'e'
      doc: 'default on a tooth of body # 1'
```

3.2 Example 2 : the car differential

The figure 2 gives the representation of the car differential. The reader could find the associated data file in the file `data_differentiel.m`

The main file `main.m` highlighting the various kinematic behaviors of this system is:

```
%% Collecting the data
data_differential

%% Kinematic analysis
% Computing the table of links and joints of the system
```

```
T=jointtable(differentiel);

% Computing the adjacency matrix
M=kinemod(differentiel);


% Number of degree of freedom
Ndof=size(M,2)-rank(M);

% Computing the speed ratio matrix
Rep=null(M);

%% Imposing the car frame speed to 0 and normalizing with respect to two
% wheels.
aux=null(M(:,2:end));
W=aux*inv(aux(2:3,1:2));

%% Nominal behaviour : driving straight ahead
% The carter is fixed to 0 and the two wheels turn at the same speed
% The speed ratio vector is normalized with respect to the transmission
aux=null([M(:,2:end);0 1 -1 0 0 0]);
W=aux*inv(aux(1,1));
W=[0;aux];
% Inventaire des dfauts:
def=default(differentiel,W);

%% The car is jacked up
% It means the transmission speed is set to 0. Then the Speed ratio vector
% is normalized with respect to the wheels.
aux=null(M(:,3:end));
W=[0 ; 0 ; aux*inv(aux(1,1))];


%% Numerical application
aux=null(M(:,2:end));
aux=aux*inv(aux(2:3,1:2));
W=aux*[26;30];
W=[0;W];
```

```
def=default(differentiel,W);
```

3.3 Example 3 : the Main Gear Box of an Alouette

The figure 3 gives the representation of the main gear box of an Alouette III. There is no data for the bearings so the type field of the turning elements of `data` in the fiel `data_alouette.m` will be 'l'.

A possible main file is :

```
%% Collecting the data
data_alouette

%% Kinematic analysis
% Computing the table of links and joints of the system
T=jointtable(data);

% Computing the adjacency matrix
M=kinemod(data);


% Number of degree of freedom
size(M,2)-rank(M)

% Computing the speed ratio matrix
Rep=null(M);

%% Nominal behaviour :
% The MGB frame (link 1) speed is set to 0 and the speed ratio
% vector is normalized with respect to the link 2

aux=null(M(:,2:end));
W=[0;aux*inv(aux(1,1))];

def=default(data,W);
```

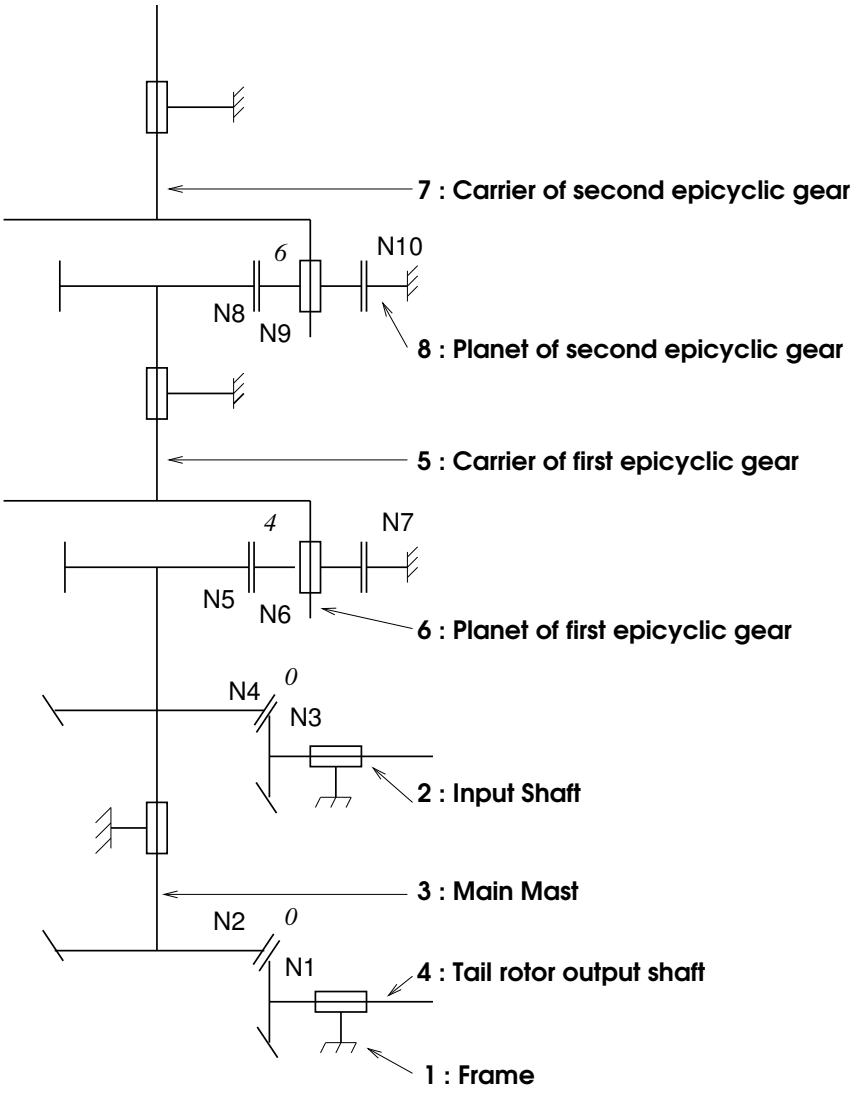

Figure 3: Main Gear Box of an helicopter

4 Appendix

In this appendix, the different `help` section of the functions of the system can be found.

Function bodyref

```
% KK=BOODYREF(DATA,II,JJ) find the reference body KK for
%    the gear pair (II,JJ) in the mechanism
%    described by DATA  (NxN structured array where N stands
%    for the number of bodies).
```

Function multiturning

```
% [MULTI,TAB]=multiturning(DATA) returns multi-turning pairs
% (shaded polygons) for mechanism DATA:
%     * MULTI(I,:) gives all the links in shaded polygon # I,
%     * TAB(J,:) gives all shaded polygons including link # J.
```

Function jointtable

```
% T=JOINTTABLE(DATA) summarizes in the matrix T the joint architecture
%  for the mechanism described in DATA (NxN structured array where N
%  stands for the number of bodies):
%   T(i,j)=0    ==> no joint between body i and body j
%   T(i,j)=Inf  ==> pivot joint between body i and body j
%   T(i,j)      ==> complex tooth number on body i in the gear joint
%                   between body i and body j
```

Function kinemod

```
% M=KINEMOD(DATA) computes the adjacency matrix of the system, i.e. the
% kinematic model of the mechanism described in DATA
% (NxN structured array where N stands for the number of bodies): that
% is the linear relationship between the N angular rates of bodies.
```

Function default

```
% DEF=DEFAULT(DATA,W) provides, in the structured variable DEF,
%    the list of all contact defauts in the mechanism described
%    by DATA (NxN structured array where N stands for the number
%    of bodies) where the N bodies angular rates are given by W
%    (a vector of length N).
%     DEF is a variable with the following structure:
%         DEF.rate   : the angular rate of the defaut,
%         DEF.N      : number of default sources,
%         DEF.joint  : a vector of two integers :[I J] ==> the
%                      default is the joint between body I and body J,
%         DEF.type   : 'e' or 'p' : type of the joint (gear or pivot),
%         DEF.doc    : string, example:
%                       * 'gear frequency',
%                       * 'default in body I',
%                       * 'default in the inside ring of bearing # 2',
%                       * 'default in the outside ring of bearing # 1',
%                       * 'default in the ball of bearing # 1'.
```