ECCOMAS Multibody Dynamics 2013
1-4 July, 2013, University of Zagreb, Croatia

Computational Kinematics of Multibody Systems: The Advantages of a Topological Method Based on its Kinematic Structure

Mariano Saura[1], Javier Cuadrado [2], Daniel Dopico[2], Ana I. Celdran[1]

[1] Department of Mechanical Engineering. Universidad Politecnica de Cartagena. Doctor Fleming, s/n, 30202 Cartagena, Spain: msaura.sanchez@upct.es

[2] Mechanical Engineering Laboratory. University of A Coruña. Mendizabal, s/n, 15403 Ferrol, Spain
javicuad@cdf.udc.es; ddopico@udc.es

Abstract

In this work a topological formulation that automatically models and solves the kinematics of multibody systems taking advantage of their kinematic structure is presented. The kinematic structure of a multibody system specifies the set and the order of the kinematic chains (structural groups) into which it can be divided, and allows the automatic definition of dependent and independent sets of coordinates needed to solve the kinematics of each one of its structural groups. A systematic method to solve the kinematics of any structural group using two different types of coordinates and formulations is presented. These solutions can be programmed in specific subroutines and called from a main program in the order specified by its kinematic structure so as to analyse the whole multibody system. These formulations have been implemented in the MATLAB programming environment, and their efficiency compared against other two formulations - global and global sparse -, making use of a scalable four-bar linkage whose number of constraint equations can be controlled by adding a different number of dyads. The computing time distribution among the main mathematical operations is compared and the main advantages of the topological methods are discussed.

Keywords: *Topological kinematic formulation, multibody kinematic structure, structural group kinematics*

1 Introduction

Computational kinematic analysis plays a fundamental role in the study of mechanical systems. It is not only necessary in multibody dynamics formulations, frequently is employed as a first stage in the design of mechanical systems (dimensional and/or kinematic synthesis) and, sometimes, the interest in the multibody system is purely kinematic (position analysis, range of movement, transmission angle, etc.).

Due to the broad field of computational kinematics applications, a permanent interest in the scientific community exists in offering methods that facilitate both the automatic modeling of multibody systems and the reduction of the computing time needed to solve it. To that end, two different families of formulations are normally used in the analysis of multibody systems: global and topological. The selected formulation and the appropriate choice of the corresponding type of coordinates determine the size of the problem, the complexity of the set of differential equations that describes its motion (ODE or DAE) and the ease of deriving those equations systematically. Three formulations are briefly described next.

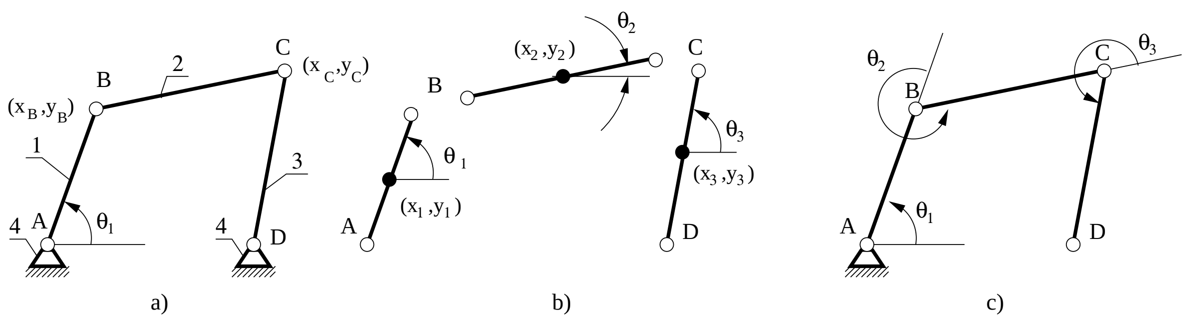


Figure 1: a) Mixed coordinates. b) Reference point coordinates. c) Tree-like structure and relative coordinates.

Global formulations. The configuration of the multibody system is described using a set of dependent coordinates: natural or mixed $[x_B\ y_B\ x_C\ y_C\ \theta_1]$ in Figure 1.a or reference point $[x_1\ y_1\ \theta_1\ x_2\ y_2\ \theta_2\ x_3\ y_3\ \theta_3]$ in Figure 1.b. Then, a simple body joint inspection is enough to identify the degrees of freedom that are constrained by each type of kinematic pair allowing the analyst to relate those dependent coordinates through the corresponding constraint equations due to rigid-body and kinematic-pair conditions.

Topological formulations based on closed loops opening. These methods use relative coordinates and are oriented to take profit of the topology of the multibody system. The closed loops have to be opened so as to yield a tree-like structure of the mechanism, and then, the kinematic relations among bodies due to the joints connecting them can be defined, along with the loop-closure equations which relate the system dependent $[\theta_2\ \theta_3]$ and independent $[\theta_1]$ coordinates (Figure 1.c).

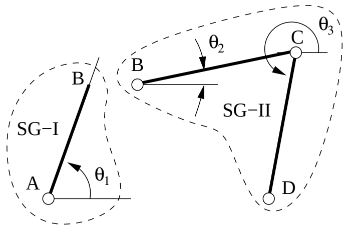
Topological formulations based on kinematic structure. This topological approach is based on the theory of structural analysis which divides a multibody system in a set of kinematic chains called structural groups (SG) by applying certain topological criteria [1]. Figure 2.a shows the division of a four-bar linkage (Figure 1.a) into two structural groups SG-I and SG-II. From the adjacency matrix of the multibody system, its kinematic structure can be obtained by computational structural analysis methods [2] and expressed in a matrix form ($KinStr$ in Figure 2.b), which will be useful for the automatic modelling of the multibody system (section 3.4).

The automatic modeling of multibody systems avoids the need for high qualified analysts and is considered safer than specific purpose solutions. For this reason, some companies have developed high performance commercial software (ADAMS, SIMPACK) but they show some known disadvantages (i.e. high-cost for research groups or industry, source code is not available) that encourages other groups of researchers to develop their own general purpose software. The use of the three formulations cited above has been extensively treated in the literature: global [3, 4], topological based in closed loops opening [5, 6, 7] and topological based on kinematic structure [8, 9], for the automatic modelling and analysis of multibody systems.

Because of its interest, the main objective of this paper is to introduce and evaluate the efficiency of a systematic methodology for the automatic modelling and solution of the kinematic analysis of planar multibody systems based on their kinematic structure. To that end we introduce in section 2 a general methodology to formulate the kinematics of any kind of SG. Two kinematic formulations frequently used in dynamic analysis are described. Applying this method one can develop specific subroutines which will be called from a main program to solve, at each calculation step, all the SG that compose the kinematic structure of the mechanism. The algorithms for the computational kinematic analysis with both global and topological formulations are introduced in section 3. In section 4 the kinematics of a four-bar scalable linkage with an increasing number of constraint equations and coordinates is performed using four different approaches and their efficiency is compared. Finally, the main conclusions of the present work are drawn in section 5.

2 Kinematic formulation of a Structural Group

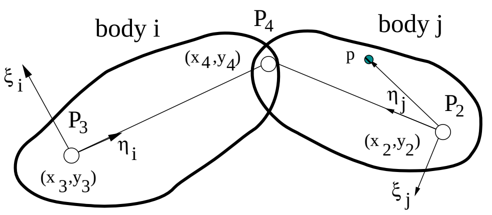
The kinematic structure of a multibody system defines what SG it is composed of and the specific order in which they have to be solved. Then, the solution of the multibody system can be carried out by solving, at each calculation step, the kinematics of all its SG, which can be programed in specific separated subroutines, whose writing can be standardized following a few steps as we introduce here. Two different kinematic formulations are considered: in the first one the system Jacobian matrix is derived with respect to time; in the second one it is derived with respect to the independent coordinates of the whole system giving rise to a third order tensor.



a)

$$KinStr = \begin{pmatrix} 0 & 1 & 2 & 2 \\ 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

b)



c)

Figure 2: a) Four-bar linkage split into structural groups: SG-I, SG-II and coordinates. b) The computational kinematic structure is represented by *KinStr* matrix [2]. c) Local coordinate systems and group coordinates in a 3R structural group.

2.1 First approach: time derivatives

The kinematics of any SG can be solved if the appropriate set of group coordinates $\mathbf{q}_G$ are selected and the corresponding constraint equations $\mathbf{\Phi}$ defined. The specific subroutine can be programmed accordingly with the following steps.

Identify the group coordinates and parameters: A local coordinate system attached to each body is defined and the appropriate set of coordinates (of any kind) that defines the kinematic chain is selected. We introduce two subsets of group coordinates: dependent $\boldsymbol{\varphi}$ and independent $\mathbf{h}$. The later are different from the independent coordinates of the whole system (referred to as $\mathbf{z}$ in many recursive formulations). Other parameters which will depend on the specific SG to solve have to be identified from the geometry of the problem and the results of the computational structural analysis i.e. reference points needed to identify the $\mathbf{h}$ coordinates, dimensions of the bodies, and so on. As an example, a 3R Assur SG is shown in Figure 2.c. The local systems $(\{\eta_i\,\xi_i\}, \{\eta_j\,\xi_j\})$ are attached to the bodies $(i,\,j)$ and the dependent $\boldsymbol{\varphi} = [x_4,\,y_4]$, and independent coordinates $\mathbf{h} = [x_3,\,y_3,\,x_2,\,y_2]$ are defined.

Solve the position problem for the SG: To solve the position problem of any SG, the corresponding constraint equations, in accordance to the selected type of coordinates, have to be defined (1). For the given set of constraint equations, the terms of the Jacobian matrix $\mathbf{\Phi}_{\boldsymbol{\varphi}}$ can be analytically obtained and the Newton-Raphson iterative method applied to obtain the values of the dependent group coordinates (1). Depending on the geometry of the SG, explicit solution of the position problem might be possible and should be taken into account to reduce the computation time.

$$\mathbf{\Phi} = 0 \qquad \rightarrow \qquad \boldsymbol{\varphi}_k = \boldsymbol{\varphi}_{k-1} - (\mathbf{\Phi}_{\boldsymbol{\varphi}})^{-1}_{k-1} \cdot \mathbf{\Phi}_{k-1} \tag{1}$$

Solve the velocity problem: As the values of the independent group velocities $\dot{\mathbf{h}}$ are known, the velocity problem can be formulated by deriving the constraint equations with respect to time and solved for the dependent ones (2). Not only the Jacobian matrix $\mathbf{\Phi}_{\mathbf{h}}$, but the whole expression $-\left(\mathbf{\Phi}_{\boldsymbol{\varphi}}\right)^{-1}\mathbf{\Phi}_{\mathbf{h}}$ can be analytically obtained in most cases for planar SG.

$$\dot{\mathbf{\Phi}}(\mathbf{q}, t) = 0 \qquad \rightarrow \qquad \dot{\boldsymbol{\varphi}} = -\left(\mathbf{\Phi}_{\boldsymbol{\varphi}}\right)^{-1}\left[\mathbf{\Phi}_{\mathbf{h}}\dot{\mathbf{h}}\right] \tag{2}$$

Solve the acceleration problem: The acceleration problem for the dependent group coordinates can be solved by deriving the velocity constraint equations with respect to time (3). Again, due to the reduced dimensions of the matrices involved, most of the calculations can be analytically performed and included into each SG subroutine. In more complicated cases, as in 3D SG, the general formulation presented here will be necessary.

$$\mathbf{\Phi}_{\boldsymbol{\varphi}}\ddot{\boldsymbol{\varphi}} + \dot{\mathbf{\Phi}}_{\mathbf{q}_G}\dot{\mathbf{q}_G} = 0 \qquad \rightarrow \qquad \ddot{\boldsymbol{\varphi}} = -\left(\mathbf{\Phi}_{\boldsymbol{\varphi}}\right)^{-1}\left[\mathbf{\Phi}_{\mathbf{h}}\ddot{\mathbf{h}} + \dot{\mathbf{\Phi}}_{\mathbf{q}_G}\dot{\mathbf{q}}_G\right] \tag{3}$$

Solve the kinematics of other POIs: Apart from the dependent coordinates, the results of other points of interest (POIs) might be necessary (i.e. center of mass, or reference points for other SG). The position, velocity and acceleration of a POI that belongs to any body ($p \in j$, Figure 2.c) is easily obtained as its local coordinates are known $\bar{\mathbf{r}}_p = [\bar{x}_p\,\bar{y}_p]$ (4). For that purpose the rotation matrix and its time derivatives have to be evaluated (5).

$$\mathbf{r}_p = \mathbf{r}_{ref} + \mathbf{A}\bar{\mathbf{r}}_p;\ \dot{\mathbf{r}}_p = \dot{\mathbf{r}}_{ref} + \dot{\mathbf{A}}\bar{\mathbf{r}}_p;\ \ddot{\mathbf{r}}_p = \ddot{\mathbf{r}}_{ref} + \ddot{\mathbf{A}}\bar{\mathbf{r}}_p \tag{4}$$

$$\mathbf{A} = \begin{bmatrix} x_j - x_i & y_i - y_j \\ y_j - y_i & x_j - x_i \end{bmatrix};\ \dot{\mathbf{A}} = \begin{bmatrix} \dot{x}_j - \dot{x}_i & \dot{y}_i - \dot{y}_j \\ \dot{y}_j - \dot{y}_i & \dot{x}_j - \dot{x}_i \end{bmatrix};\ \ddot{\mathbf{A}} = \begin{bmatrix} \ddot{x}_j - \ddot{x}_i & \ddot{y}_i - \ddot{y}_j \\ \ddot{y}_j - \ddot{y}_i & \ddot{x}_j - \ddot{x}_i \end{bmatrix} \tag{5}$$

2.2 Second approach: the three order tensor $\mathbf{\Phi}_{qq}$

In some kinematic and dynamic formulations it is preferable to express the velocities and accelerations of the dependent group coordinates with respect to the independent group coordinates of the whole system (usually known as $\mathbf{z}$). This formulation can also be used with the help of the SG decomposition. The two first steps are similar to the previous option. The velocity and acceleration problems are solved by introducing the $\mathbf{R}$ and $\mathbf{S}$ matrices (6), which are extensively used in efficient recursive formulations as an alternative method to Baumgarte's constraint stabilization [10].

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}^d \\ \mathbf{S}^i \end{bmatrix} = \begin{bmatrix} (\mathbf{\Phi}_{\boldsymbol{\varphi}})^{-1} \\ 0_{f\times m} \end{bmatrix}, \qquad \mathbf{R} = \begin{bmatrix} \mathbf{R}^d \\ \mathbf{R}^i \end{bmatrix} = \begin{bmatrix} -(\mathbf{\Phi}_{\boldsymbol{\varphi}})^{-1}\,\mathbf{\Phi}^i_{\mathbf{q}_G} \\ I_f \end{bmatrix}, \qquad \mathbf{K}_{\boldsymbol{\varphi}\mathbf{h}} = \mathbf{R}^d \tag{6}$$

Solve the velocity problem: The elements in matrix $\mathbf{K}_{\boldsymbol{\varphi}\mathbf{h}}$ correspond to velocity coefficients and show that the dependent group velocities $\dot{\boldsymbol{\varphi}}$ can be expressed as a linear combination of the independent ones $\dot{\mathbf{h}}$ (7).

$$\mathbf{K}_{\varphi\mathbf{h}} = -\left(\boldsymbol{\Phi}_{\varphi}\right)^{-1}\boldsymbol{\Phi}_{\mathbf{h}} \rightarrow \dot{\varphi} = \mathbf{K}_{\varphi\mathbf{h}}\dot{\mathbf{h}} \tag{7}$$

The velocity coefficients $\mathbf{K}_{\mathbf{hz}}$ that define the velocity of the independent group coordinates as a linear combination of the independent velocities of the whole system $\dot{\mathbf{z}}$ are available for the coordinates of the previous SG. Then it is straightforward to express the velocity of the dependent group coordinates in terms of the independent ones of the whole system by means of their respective velocity coefficients $\mathbf{K}_{\varphi\mathbf{z}}$, which are very useful in dynamic formulations, and to calculate the dependent velocities (8).

$$\mathbf{K}_{\varphi\mathbf{z}} = \mathbf{K}_{\varphi\mathbf{h}}\mathbf{K}_{\mathbf{hz}} \qquad \rightarrow \qquad \dot{\varphi} = \mathbf{K}_{\varphi\mathbf{z}}\,\dot{\mathbf{z}} \tag{8}$$

Solve the acceleration problem: To solve the acceleration problem in terms of the system independent coordinates, the time derivative of (8) has to be performed. Considering the chain differentiation rule, the system in (9) is obtained. In this approach, instead of differentiating the velocity coefficients with respect to time, as in the previous formulation, they have been differentiated with respect to the **z** coordinates.

$$\ddot{\varphi} = \mathbf{K}_{\varphi\mathbf{z}}\,\ddot{\mathbf{z}} + \sum_{i=1}^{f}\dot{\mathbf{z}}_i \cdot \left(\mathbf{L}_{\varphi\mathbf{z}}\right)_{z_i} \cdot \dot{\mathbf{z}} \tag{9}$$

The terms $\left(\mathbf{L}_{\varphi\mathbf{z}}\right)_{z_i}$ in the summatory are defined as velocity coefficient derivatives; that is, the derivative of the velocity coefficient matrix $\mathbf{K}_{\varphi\mathbf{z}}$ with respect to a given independent coordinate z_i.

$$\left(\mathbf{L}_{\varphi\mathbf{z}}\right)_{z_i} = \frac{d\,\mathbf{K}_{\varphi\mathbf{z}}}{d\,z_i}$$

Calculating the velocity coefficient derivatives is the most complicated and time consuming part of this approach. However, an analytic expression can be introduced by comparing the acceleration of the dependent group coordinates expressed in terms of both the independent group system coordinates **z** (9) and the independent group ones **h** (10) respectively.

$$\ddot{\varphi} = \mathbf{K}_{\varphi\mathbf{h}} \cdot \ddot{\mathbf{h}} + \sum_{j=1}^{r}\dot{h}_j \cdot \left(\mathbf{L}_{\varphi\mathbf{h}}\right)_{h_j} \cdot \dot{\mathbf{h}} \tag{10}$$

The accelerations $\ddot{\mathbf{h}}$ can also be expressed in terms of the independent system coordinates z_i (11).

$$\ddot{\mathbf{h}} = \mathbf{K}_{\mathbf{hz}} \cdot \ddot{\mathbf{z}} + \sum_{i=1}^{f}\dot{z}_i \cdot \left(\mathbf{L}_{\mathbf{hz}}\right)_{z_i} \cdot \dot{\mathbf{z}} \tag{11}$$

And the independent group velocities, in terms of the independent system ones (12):

$$\dot{\mathbf{h}} = \mathbf{K}_{\mathbf{hz}} \cdot \dot{\mathbf{z}} \tag{12}$$

Substituting (11) and (12) in (10) we obtain:

$$\ddot{\varphi} = \mathbf{K}_{\varphi\mathbf{h}}\mathbf{K}_{\mathbf{hz}} \cdot \ddot{\mathbf{z}} + \sum_{i=1}^{f}\dot{z}_i \cdot \left(\mathbf{K}_{\varphi\mathbf{h}} \cdot \left(\mathbf{L}_{\mathbf{hz}}\right)_{z_i} + \sum_{j=1}^{r}\mathbf{K}_{h_j z_i} \cdot \left(\mathbf{L}_{\varphi\mathbf{h}}\right)_{h_j} \cdot \mathbf{K}_{\mathbf{hz}}\right) \cdot \dot{\mathbf{z}} \tag{13}$$

Comparing (13) and (9), it can be seen that the coefficients of the independent accelerations $\ddot{\mathbf{z}}$ correspond to those already introduced in (8), and that the coeficients of the centripetal and Coriolis terms, $\mathbf{L}_{\varphi\mathbf{z}}$, which we are looking for, correspond to the expression into brackets in (13). Then we can write (14) to calculate their values.

$$\left(\mathbf{L}_{\varphi\mathbf{z}}\right)_{z_i} = \left(\mathbf{K}_{\varphi\mathbf{h}} \cdot \left(\mathbf{L}_{\mathbf{hz}}\right)_{z_i} + \sum_{j=1}^{r}\mathbf{K}_{h_j z_i} \cdot \left(\mathbf{L}_{\varphi\mathbf{h}}\right)_{h_j} \cdot \mathbf{K}_{\mathbf{hz}}\right) \tag{14}$$

In (14) the velocity coefficient derivatives $\left(\mathbf{L}_{\mathbf{hz}}\right)_{z_i}$ are known from POIs that belongs to previous SG. The velocity coefficients derivatives $\left(\mathbf{L}_{\varphi\mathbf{h}}\right)_{h_j}$ of the dependent group coordinates φ with respect to the **h** coordinates can be obtained both analytically and numerically by using (15).

$$(\mathbf{L}_{\varphi\mathbf{h}})_{h_j} = \mathbf{S}\cdot\left(\frac{d\,\mathbf{\Phi}_{\mathbf{h}}}{d\,h_j} - \frac{d\,\mathbf{\Phi}_{\varphi}}{dh_j}\cdot\mathbf{K}_{\varphi}\right) \tag{15}$$

The two topological formulations introduced in this section have been implemented in the MATLAB programming environment together with a global formulation (section 3) and their efficiency is compared in section 4.

3 Computational kinematics: Global and SG Topological methods

In this section we present two algorithms for the kinematic analysis of planar multibody systems (MBS). The first one corresponds to a global formulation, and the second one to the specific development introduced in this work, which also considers the computational structural analysis and the automatic modelling and solution of the MBS.

Algorithm 1: Kinem: Global solution

```
1.1:  MBDatos                 /* Read data MBS */
1.2:  for $t = t_0 : timeStep : t_f$ do
1.3:      $\mathbf{z} = \mathbf{z} + \Delta\mathbf{z}$     /* set indep.  values */
1.4:      %% Position problem %
1.5:      evaluate $\mathbf{\Phi}$ $\rightarrow$ mFi
1.6:      $error = norm(\mathbf{\Phi})$
1.7:      while $error > tolerance$ do
1.8:          evaluate $\mathbf{\Phi_q}$ $\rightarrow$ Jacob
1.9:          extract $\mathbf{\Phi_q^d}$
1.10:         solve $\mathbf{q}_k^d = \mathbf{q}_{k-1}^d - \left(\mathbf{\Phi_q^d}\right)_{k-1}^{-1} \cdot \mathbf{\Phi}_{k-1}$
1.11:         evaluate $\mathbf{\Phi}$ $\rightarrow$ mFi
1.12:         $error = norm(\mathbf{\Phi})$
1.13:     end
1.14:     %% Velocity problem %%%
1.15:     evaluate $\mathbf{\Phi_q}$ $\rightarrow$ Jacob
1.16:     extract $\mathbf{\Phi_q^d}$
1.17:     extract $\mathbf{\Phi_q^i}$
1.18:     solve $\dot{\mathbf{q}}^d = -\left(\mathbf{\Phi_q^d}\right)^{-1} \mathbf{\Phi_q^i}\dot{\mathbf{q}}^i$
1.19:     %% Acceleration problem %%%%
1.20:     evaluate $\dot{\mathbf{\Phi}}_\mathbf{q}\dot{\mathbf{q}}$ $\rightarrow$ Fiqpqp
1.21:     evaluate $-\left[\mathbf{\Phi_q^i}\ddot{\mathbf{q}}^i + \dot{\mathbf{\Phi}}_\mathbf{q}\dot{\mathbf{q}}\right]$
1.22:     solve $\ddot{\mathbf{q}}^d = -\left(\mathbf{\Phi_q^d}\right)^{-1}\left[\mathbf{\Phi_q^i}\ddot{\mathbf{q}}^i + \dot{\mathbf{\Phi}}_\mathbf{q}\dot{\mathbf{q}}\right]$
1.23:     $\mathbf{z} = \mathbf{z} + \Delta\mathbf{z}$
1.24: end
```

Algorithm 2: Kinem: Topol. SG solution

```
2.1:  MBDatos;                /* Read data MBS */
2.2:  %% Computational Struct. Analysis
2.3:  $Kinstr$ $\rightarrow$ CompSG  /* Kin.Struct */
2.4:  %% MBS automatic modelling
2.5:  $MGroups; POI$ $\rightarrow$ MBSModelling
2.6:  %% MBS Kin. analysis
2.7:  for $t = t_0 : timeStep : t_f$ do
2.8:      $\mathbf{z} = \mathbf{z} + \Delta\mathbf{z}$ /* set indep.  values */
2.9:      for $ng = 2 : length(MGroups)$ do
              /* solve each SG                */
2.10:         switch $MGroups(ng).kind$ do
2.11:             case MGroups(ng).kind == 1RSG
2.12:                 POI $\rightarrow$ Solve_1RSG
2.13:             case MGroups(ng).kind == 3RSG
2.14:                 POI $\rightarrow$ Solve_3RSG
2.15:
2.16:         endsw
2.17:     end
2.18: end
```

Algorithm 3: Kinem: Global Sparse opt

```
      /* calls to Sparse function          */
3.1:  ...
3.2:  evaluate $\mathbf{\Phi_q}$ $\rightarrow$ Jacob
3.3:  extract $\mathbf{\Phi_q^d}$
3.4:  $\underline{\mathbf{\Phi}}_\mathbf{q}^d$ $\rightarrow$ Sparse($\mathbf{\Phi_q^d}$) ...
3.5:  solve $\mathbf{q}_k^d = \mathbf{q}_{k-1}^d - \left(\underline{\mathbf{\Phi}}_\mathbf{q}^d\right)_{k-1}^{-1} \cdot \mathbf{\Phi}_{k-1}$
3.6:  ...
```

3.1 Typical algorithm for a global method

In Algorithm 1, a global formulation for the kinematic analysis of MBS is presented. A coordinate partition has been used to evaluate the velocities and accelerations of the dependent coordinates $\mathbf{q}_d$ of the system, once the independent $\mathbf{q}_i$ are known. To that end, the position problem is solved using a Newton-Raphson method (lines 1.5 - 1.13). Both the constraint equations $\mathbf{\Phi}$ and the Jacobian matrix $\mathbf{\Phi}_q^d$ are evaluated in separated subroutines, *mFi* and *Jacob* respectively. The velocity problem is solved in lines (1.14 - 1.18) and the acceleration problem in lines (1.19 - 1.22). Only $\dot{\mathbf{\Phi}}_q\dot{\mathbf{q}}$ has to be evaluated in a separated subroutine *Fiqpqp*. This solution loops until a final value of time is reached (or other stopping criterion is defined).

3.2 Proposed algorithm for the SG method

In Algorithm 2 a schematic representation of the computational method developed in this work for the kinematics of multibody systems based on its kinematic structure is presented. It is mainly divided into three steps that correspond to the computational structural analysis (subsection 3.3), the automatic modelling of the MBS (subsection 3.4) and, finally, the computational kinematic analysis of the system (subsection 3.5).

3.3 Computational structural analysis

The application of the SG decomposition with both graph-analytical and computational methods based on Assur theory [11, 12, 13] and, more recently, based on combinatorial processes [14], has been extensively treated in the literature for planar mechanisms with limitation to lower kinematic pairs, Assur Groups and one degree of freedom.

The method we describe in this work applies a previously developed algorithm for the computational structural analysis which uses an extended concept of Structural Group, making it useful not only to planar mechanisms with Assur Groups and one degree of freedom, but also to any kind of MBS.

The solution starts, as shown in Algorithm 2, reading detailed information of the MBS from a separated file ($MBDatos$). This information includes the adjacency matrix, which contains the system topology and is enough to perform the computational structural analysis (line 2.3) and obtain the kinematic structure, $KinStruc$, of the multibody system by accessing a separated algorithm which has been developed and explained in a previous work [2].

3.4 Automatic modeling of the multibody system

This procedure can be divided into three steps. The first one stores the identification number of the bodies that pertain to each SG in the data structure $MGroups.link$. In addition, analyzing the rows of the $AdjacencyMatrix$, the coordinates $\mathbf{q}_G$ for each SG are stored on $MGroups.coordinates$, distinguishing between dependent and independent ones.

In a second step, the $AdjacencyMatrix$ is again accessed to determine the type of each SG, which will be saved on $MGroups.kind$. This result is used in the for-end loop (2.9 - 2.17) to perform the complete kinematic analysis by accessing specific subroutines (switch-case, in 2.10 - 2.16) in accordance to each one of the SG that have been obtained.

In the third and last step, the algorithm allows the user to introduce additional information regarding each of the obtained SG by two methods: using a Graphical User Interface (GUI) or introducing the name of the file that contains the corresponding information.

3.5 Computational kinematic analysis

Although there exists much literature that applies global and topological methods for the kinematic and dynamic analysis of multibody systems, few works have been found about solutions based on SG decomposition. In [15], an algorithm for the computational kinematic analysis of linkages formed by any number of 3R Assur Groups (two bodies and three rotation kinematic pairs) is presented. In [8], an efficient kinematic solution for planar, one-degree-of-freedom multibody system is introduced by reordering the elements of its Jacobian matrix. Varbanov [9] introduces the integration of kinematic synthesis and analysis of planar mechanisms formed by one body with one degree of freedom, combined with different forms of dyads (AG of Class I). There are few works that automatically perform the computational structural analysis of mechanism with any number of degrees of freedom, planar or spatial, and that use their results for the automatic modelling and kinematic analysis of a given multibody system.

Once the MBS has been modelled, a kinematic analysis based on the formulations presented in sections (2.1) or (2.2) can be automatically performed by calling specific subroutines (Algorithm 2, lines 2.9-2.17). These subroutines might follow a similar scheme as the one introduced in lines (1.3-1.21, Algorithm 1), but adapted to each specific SG.

4 Global versus Topological formulation based on SG decomposition

In this section we compare the efficiency of four different methods for the computational kinematic analysis of MBS: I) Global formulation with mixed coordinates, II) Topological formulation based on SG with mixed coordinates and time derivatives (sec.2.1), III) Topological formulation with reference point coordinates and third order tensor (sec.2.2). Additionally, due to the sparsity of the matrices in the global solution (I), we introduce a fourth solution: IV) Global sparse formulation, which is the same as (I), but the linear and nonlinear systems of equations are solved with the MATLAB sparse option.

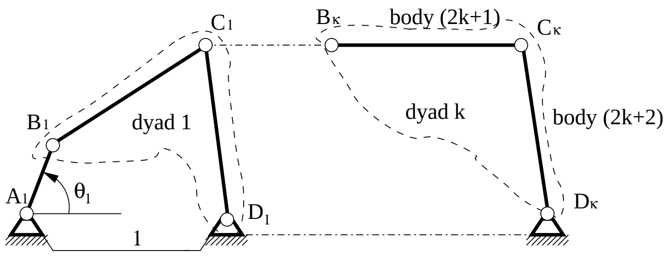


Figure 3: Scalable four-bar linkage. A number (k) of dyads, as many as required, can be added to the crank $\overline{A_1B_1}$

To evaluate the efficiency of the four methods, we use an scalable four-bar linkage consisting of a crank (body 2) and an increasing number k of dyads: (bodies $(2k+1)$, $(2k+2)$, joined with rotation joint, in Figure 3). This method allows us to control the number of constraint equations and to evaluate how it influences the calculation time of the four methods.

Both the global (I, IV) and the topological (II) solutions define the multibody system with the same number of constraint equations $m = 2 \times k + 2$ and number of coordinates $n = 2 \times k + 3$. In (III) these are reduced in two units each $m = 2 \times k$ and $n = 2 \times k + 1$.

To carry the kinematic analysis out, the input movement is defined as the crank rotation at a constant velocity $\dot{\theta}_1 = 1$ r/s. All lengths are set equal to 6, except $\overline{A_1B_1} = 2$. In order to let the results be comparable for a reduced number of dyads, the calculations will last until the crank completes 500 rotations.

I) Global solution: In this formulation the $n = 2 \times k + 3$ coordinates correspond to the rotation angle θ_1 and the two Cartesian coordinates of point B_1 and the k points C_k. The Algorithm 1 shows the sequence of the main program in which three external functions are called, *mFi*, *Jacob* and *Fiqpqp*, which automatically form the corresponding vectors and matrices $\mathbf{\Phi}$, $\mathbf{\Phi_q}$, $\dot{\mathbf{\Phi}}_\mathbf{q}\dot{\mathbf{q}}$ whose size depends on the number of dyads that have been considered.

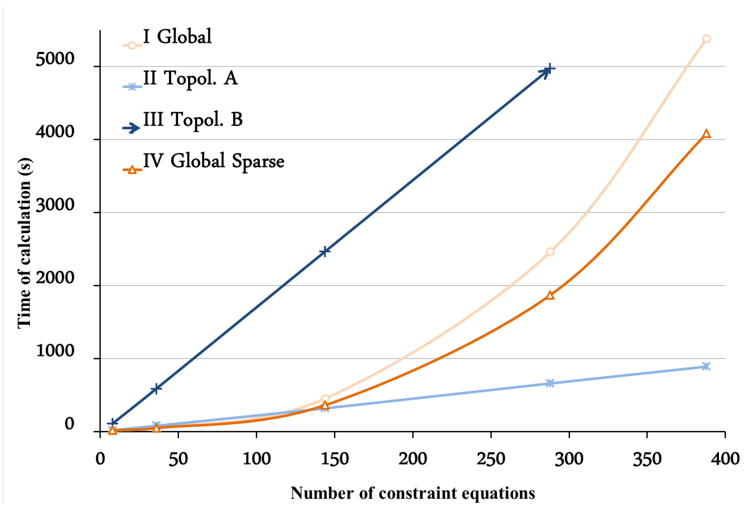
II) Topological solution: This solution uses the same set of coordinates as the previous one. The main steps needed to execute this solution have been introduced in Algorithm 2, which is the same for both topological options, (II and III), based on SG decomposition. The differences between the two formulations reside in the kinematic formulation used to solve any kind of structural group. As it has been shown, for every calculation step each SG that forms the kinematic structure of the multibody system is analyzed (Algorithm 2). Depending on the kind of SG, the corresponding specific subroutine is called: (*1R_SG*) to calculate $\boldsymbol{\varphi}_I = (x_B\, y_B)$ for a given value of $\mathbf{h}_I = z$, or *(*3R_SG) to calculate $\boldsymbol{\varphi}_k = (x_{C_k}\, y_{C_k})$ given the values of $\mathbf{h}_k = (x_{C_{k-1}}\, y_{C_{k-1}}\, x_{D_{k-1}}\, y_{D_{k-1}})$. In option II, any of these subroutines follows the formulation introduced in section 2.1 based in a coordinate partitioning method, which is similar to the one shown in Algorithm 1 (lines 4-22) for the global approach. However, the sizes of the matrices involved are now considerably reduced.

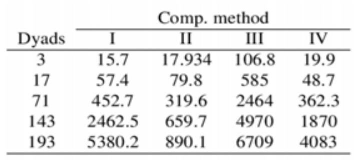
III) Topological solution: In order to implement this method, a set of rotation coordinates θ_{nb} of the nb bodies have been chosen together with the Cartesian coordinates of the B_1 and C_k points. The main reason for it is not to compare the efficiency of different types of coordinates, but to demonstrate that any kind of coordinates can be selected if the appropriate constraint equations are defined. The main program has the same structure than method II (Algorithm 2), but the code needed to solve each kind of SG is, in this case, cumbersome.

IV) Global sparse solution: Due to the geometry of the scalable four-bar linkage, the constraint equations of any k dyad depend on the Cartesian coordinates of points C_k and C_{k-1} only. This makes the Jacobian matrix $\mathbf{\Phi}_q$ sparse and special techniques have to be taken into consideration to improve the efficiency of the calculations. The MATLAB programming environment allows to easily consider this option which has been implemented as shown in Algorithm 3.

4.1 Calculation time for different approaches

Figure 4 shows the calculation time in seconds for each computational method (right) and their graphical evolution (left) as a function of the number of constraint equations. While the global methods (I, IV) evolve exponentially, the topological ones (II, III) does it linearly. Apart from option III, which show a very low efficiency, the other three methods show a very similar performance in a range of 3-17 dyads (8-144 constraint equations). More specifically, the global method I is faster for small number of constraint equations and, only for a number close to 36, the *sparse* function shows some advantage.





Dyads	Comp. method			
	I	II	III	IV
3	15.7	17.934	106.8	19.9
17	57.4	79.8	585	48.7
71	452.7	319.6	2464	362.3
143	2462.5	659.7	4970	1870
193	5380.2	890.1	6709	4083

Figure 4: Evolution of the calculation time required by the four kinematic solutions for the scalable four-bar linkage depending on a) the number of constraint equations, b) the number of dyads.

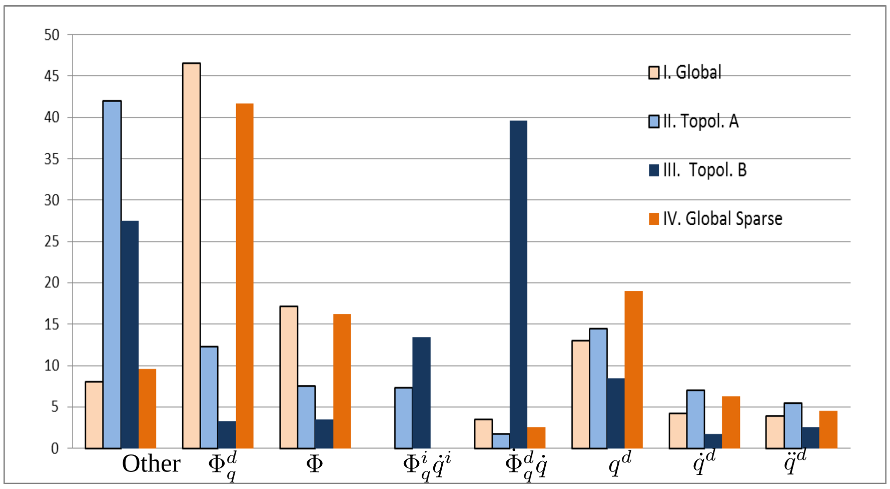


Figure 5: Calculation time distribution (%) for the different kinematic solutions among their main operations in the kinematic analysis of the four-bar scalable linkage with 17 dyads (144 constraint equations).

Close to a number of 71 dyads (144 constraint equations) and above, the linear behavior of method II turns it into the most efficient of the four ones that have been considered.

In order to study more in depth the efficiency of these methods, we have used the MATLAB profile tool to watch how they distribute their calculation time among all the mathematical operations. Figure 5 shows the percentage of the calculation time that the four methods employ in their most important operations: calculate the Jacobian matrix $\mathbf{\Phi}_q^d$, the vector of constraint equations $\mathbf{\Phi}$, the independent term for the velocities $\mathbf{\Phi}_{\mathbf{q}}^i \dot{\mathbf{q}}^i$, one of the terms involved in the calculation of the accelerations $\dot{\mathbf{\Phi}}_q \dot{\mathbf{q}}$, and the solution of the nonlinear system of equations for the positions $\mathbf{q}^d$, and the linear systems for the velocities $\dot{\mathbf{q}}^d$ and accelerations $\ddot{\mathbf{q}}^d$. The 'Other' item states for all the other code lines in which each method performs other mathematical operations that are time consuming. The topological method III does not calculate $\mathbf{\Phi}_{\mathbf{q}}^i \dot{\mathbf{q}}^i$ nor $\dot{\mathbf{\Phi}}_q \dot{\mathbf{q}}$, so in their place we represent the $\mathbf{K}$ and $\mathbf{L}$ calculations as defined in (8) and (14) respectively.

There are several interesting results from this analysis. The first one is that, while the global methods spend near a 45% in evaluating the terms of the Jacobian matrix, the topological method II spends only a 12%, and, where the first ones spend a 17% in evaluating the vector of constraint equations, the topological method II spends a 7%. However, while global methods spend only a 7% in 'other' operations, method II dedicates about a 42% of its time accessing to data of

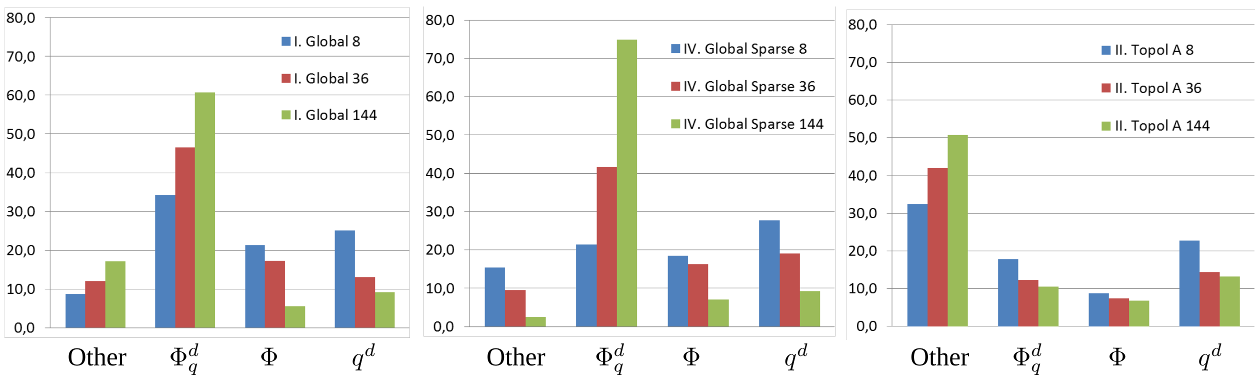


Figure 6: Time percentage distribution among the most relevant items depending on the number of constraint equations for three formulations: I. Global (left), IV. Global Sparse (center) and II. Topological (right)

SG that are being or have been analyzed already. Another interesting result is that method III spends almost a 70% of its time evaluating $\mathbf{L}$ and 'other' items. Those results confirm the inefficiency of method III, reveal that global methods efficiency is conditioned because of the need to construct the Jacobian matrix, and allow us to expect an improvement of the II topological method as soon as we are capable of reducing the time that is spent in performing 'other' calculations.

All the methods (except III) spend a low percentage of their time in solving the velocities and accelerations; that allows us to study more in depth how the time that is employed in their main functions is distributed depending on the number of coordinates (Figure 6).

Although the three methods are influenced by the number of coordinates, the topological one shows a similar reduction in the percentage of time devoted to perform all operations different from the 'other' item. However, this method is the one that shows a major increase of time spent in 'other' operations when the number of constraints is increased. The global methods, on the other hand, show a strong sensitivity to the number of constraint equations, which is more relevant for the global sparse (IV). As the number of constraint equations increase, the percentage of time spent in evaluating the Jacobian of the multibody systems grows exponentially. It can be seen that global methods could improve by considering sparse matrix techniques if the number of constraint equations is relevant (above 36), and if efficient methods are used to only store the elements of the Jacobian matrix different from zero.

5 Conclusions

In this work a topological method based on the kinematic structure of a multibody system that automatically models and solves its kinematic analysis is presented. This topological method allows the use of any kind of coordinates $\mathbf{q}$ though the appropiate set of constraint equations $\mathbf{\Phi}$ has to be defined. Two formulations has been presented for the topological method: one in which the Jacobian matrix of the constraint equations ($\mathbf{\Phi}_{\mathbf{q}}$) is differentiated with respect to the time (method II) and other in which the Jacobian matrix is differentiated with respect to the independent coordinates of the whole system, $\mathbf{z}$, (method III). The efficiency of these formulations has been compared against other two formulations - global (method I) and global sparse (method IV)- making use of a scalable four-bar linkage that allows us to control the number of constraint equations. The results that have been obtained allow us to list some advantages that the topological methods based on the kinematic structure (II and III) introduce with respect to the global methods (I and IV) used for comparison purposes:

- The kinematic structure can be obtained computationally in matrix form and can be used to automatically model and solve the kinematics of the multibody system. This considerably streamlines the modelling phase and avoids making mistakes in this first stage of the analysis.
- These methods are not limited to a specific kind of coordinates: (II) and (III) use natural and mixed coordinates respectively. It is only necessary to program the specific subroutines that solve the different kinds of SG using the selected type of coordinates.
- Two different formulations for the kinematic analysis of a general SG have been introduced in a systematic form that might be useful as a guide to implement new specific subroutines for other kinds of SG.

- Topological method III is much more complicated to program in a subroutine than topological method II. This is not because of the type of selected coordinates, but because of the decision to evaluate all the velocity coefficients **K** and velocity coefficient derivatives **L**. Although matrix **K** and the three order tensor **L** might be useful for dynamic purposes, sensitivity analysis or optimization problems, the time needed to evaluate them makes this solution highly inefficient even for a system with only one degree of freedom.
- Topological methods can be extended to the kinematics and dynamics of spatial multibody systems as global methods do.
- The computing time grows linearly in topological methods II and III, and exponentially in global methods I and IV. The main reason for these results is the size of the matrices in each method: the global methods deal with $Dim(\mathbf{\Phi_q}) = m \times n$, while the topological methods deal with $Dim(\mathbf{\Phi_q}) = 2 \times 2$, considering the SG that have been used in this study.
- When sparse matrix techniques are used in global formulation (IV), the method is slightly faster than (I) for problems under 100 constraints. However, from that point on, the advantage of topological method II linearly increases.

References

[1] M. Z. Kolovsky, A. N. Evgrafov, Yu. A. Semenov, and A. V. Slousch. *Advanced theory of mechanism and machines*. Springer, 2000.

[2] M. Saura, A.I. Celdran, D. Dopico, and J. Cuadrado. Computational structural analysis of planar multibody systems with lower and higher kinematic pairs. In *Proceedings of the 2^{nd} Joint International Conference on Multibody System Dynamics*, Stuttgart, Germany, 2012.

[3] E. J. Haug. *Computer-Aided kinematics and dynamics of mechanical systems*. Allyn and Bacon, Boston, MA, 1989.

[4] A. A. Shabana. *Dynamics of Multibody systems*. Cambridge Univ Press, 2nd edition, 1998.

[5] A. Kecskeméthy, T. Krupp, and M. Hiller. Symbolic processing of multiloop mechanism dynamics using closed-form kinematics solution. *Multibody System Dynamics*, 1(1):23–45, 1997.

[6] M. Leger and J. McPhee. Selection of modeling coordinates for forward dynamics multibody simulations. *Multibody System Dynamics*, 18(18):277–297, 2007.

[7] J. Cuadrado, D. Dopico, M. Gonzalez, and M. A. Naya. A combined penalty and recursive real-time formulation for multibody dynamics. *Journal of Mechanical Design*, 126(4):602–608, 2004.

[8] J. Buskiewicz. A method of optimization of solving a kinematic problem with the use of structural analysis algorithm (SAM). *Mechanism and Machine Theory*, 41:823–837, 2006.

[9] H. Varbanov, T. Yankova, K. Kulev, and S. Lilov. S&A - Expert system for planar mechanism design. *Expert systems with applications*, 31(31):558–569, 2006.

[10] J. G. Jalon. *Kinematic and dynamic simulation of multibody systems*. Springer-Verlag, NY, 1993.

[11] T. S. Mruthyunjaya and M.R. Raghavan. Computer aided analysis of the structure of kinematic chains. *Mechanism and Machine Theory*, 19(19):357–368, 1984.

[12] H.Ding, Z.Huang, and D.Mu. Computer-aided structure decomposition theory of kinematic chains and its applications. *Mechanism and Machine Theory*, 43(12):1596–1609, 2008.

[13] I.Popescu and D.B.Marghitu. Structural design of planar mechanisms with dyads. *Multibody System Dynamics*, (19):407–425, 2008.

[14] M.Terushkin and S.Offer. Applying rigidity theory methods for topological decompositions and synthesis of gear trains systems. In *Proceedings of the ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 1–10, Chicago, IL, USA, 2012.

[15] M. Saura. Utilidad del analisis estructural en el analisis cinematico de mecanismos articulados planos. In *Proceedings of the Congreso IberoAmericano de Ingenieria Mecanica*, pages 1–11, La Habana, Cuba, 2005.