

ADAMS Theory in a Nutshell
for
Class ME543
Department of Mechanical Engineering
The University of Michigan
Ann Arbor
March 22, 2001

Dan Negrut
dnegr@adams.com

Brett Harris
bharris@adams.com

Definitions, Notations, Conventions

1. Generalized Coordinates used in ADAMS

In ADAMS, the position of a rigid body is defined by 3 Cartesian coordinates x , y , and z .

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (1)$$

The orientation of a rigid body is defined by a set of 3 Euler angles that correspond to the 3-1-3 sequence rotation: \mathbf{y} , \mathbf{q} , and \mathbf{f} , respectively. These 3 angles are stored in an array (not vector) in the following form:

$$\mathbf{e} = \begin{bmatrix} \mathbf{y} \\ \mathbf{f} \\ \mathbf{q} \end{bmatrix} \quad (2)$$

The set of generalized coordinates associated with rigid body i in ADAMS is denoted in what follows by

$$\mathbf{q}_i = \begin{bmatrix} \mathbf{p}_i \\ \mathbf{e}_i \end{bmatrix} \quad (3)$$

Based on this choice of generalized coordinates, the body longitudinal and angular velocity are obtained as

$$\mathbf{u} = \mathbf{p}\dot{\mathbf{z}} \quad (4)$$

$$\bar{\mathbf{w}} = \mathbf{B}\dot{\mathbf{z}} = \mathbf{B}\dot{\mathbf{z}} \quad (5)$$

where

$$\mathbf{B} = \begin{bmatrix} \sin \mathbf{f} \sin \mathbf{q} & 0 & \cos \mathbf{f} \\ \cos \mathbf{f} \sin \mathbf{q} & 0 & -\sin \mathbf{f} \\ \cos \mathbf{q} & 1 & 0 \end{bmatrix} \quad (6)$$

and $\bar{\mathbf{w}}$ is the body angular velocity expressed in body local coordinate system. Equation (6) is important as it defines the relationship between the angular velocity of the body; i.e., an intrinsic characteristic of the body, and our choice of generalized coordinates.

Finally, note the relationship between the time derivative of the body orientation matrix \mathbf{A} and angular velocity $\bar{\mathbf{w}}$:

$$\dot{\mathbf{A}} \sim \mathbf{A} \bar{\mathbf{w}} \quad (7)$$

where a \sim represents the skew-symmetric operator.

For an entire mechanical system models containing nb bodies, the array

$$\mathbf{q} = [\mathbf{q}_1^T \quad \mathbf{q}_2^T \quad \mathbf{K} \quad \mathbf{q}_{nb}^T]^T = [q_1 \quad q_2 \quad \mathbf{K} \quad q_n]^T \quad (8)$$

with $n = 6 \cdot nb$ will describe at a given time the position and orientation of each body in the system.

2. Joints in ADAMS

Joints in ADAMS are regarded as constraints that act among some of the coordinates q_1 through q_n of Eq.(8). From a mathematical perspective, such a constraint assumes the expression

$$\Phi(\mathbf{q}) = 0 \quad (9)$$

For example, a revolute joint acting between 2 bodies would induce a set of 5 constraints like the one in Eq.(9) to allow one degree of freedom between the two bodies connected by this joint.

The collection of all constraints induced by the joints present in the model is denoted by \mathbf{F} :

$$\mathbf{F}(\mathbf{q}) = [\mathbf{F}_1^T(\mathbf{q}) \quad \mathbf{F}_2^T(\mathbf{q}) \quad \mathbf{K} \quad \mathbf{F}_{nj}^T(\mathbf{q})]^T = [\Phi_1(\mathbf{q}) \quad \Phi_2(\mathbf{q}) \quad \mathbf{K} \quad \Phi_m(\mathbf{q})]^T \quad (10)$$

where nj is the number of joints in the model, and m is the sum of the number of constraints induced by each joint. Note that $\mathbf{q} \in \mathbf{R}^n$, while $\mathbf{F} \in \mathbf{R}^m$. Typically, $m < n$; i.e., the number of generalized coordinates is larger than the number of constraints they must satisfy.

By taking one time derivative of the position kinematic constraint equations of Eq.(10), the velocity kinematic constraint equations are obtained as

$$F_q \dot{\mathbf{q}} = \mathbf{0} \quad (11)$$

By taking yet another time derivative of Eq.(11), the acceleration kinematic constraint equations are obtained as

$$F_q \ddot{\mathbf{q}} - (F_{qq} \dot{\mathbf{q}}) \dot{\mathbf{q}} = \mathbf{0} \quad (12)$$

Equations (10) through (12) can be seen as conditions that the generalized coordinates array \mathbf{q} along with its first and second time derivatives must satisfy. This is to ensure that the evolution of the mechanical system makes sense; i.e., the mechanism is assembled, and the parts move such that the constraints imposed by joints are obeyed at every time.

3. Motions in ADAMS

From a mathematical perspective, motions indicate that a generalized coordinate of the system, or an expression depending on generalized coordinates explicitly depends on time. As an example, consider a simple pendulum connected to ground through a revolute joint. A motion might impose that the angle associated with its unique degree of freedom will change in time like $\mathbf{a} = \sin(10\mathbf{p} \cdot t)$.

Generally, a motion is represented as a time dependent constraint equation:

$$\Phi(\mathbf{q}, t) = 0 \quad (13)$$

Revisiting the definition of the position, velocity, and acceleration kinematic constraint equations, for constraint equations induced by either *joints* or *motions* in the most general case the following equations must be satisfied at any time t .

$$F(\mathbf{q}, t) = 0 \quad (14)$$

$$F_q(\mathbf{q}, t) \cdot \dot{\mathbf{q}} = -F_t(\mathbf{q}, t) \quad (15)$$

$$F_q(\mathbf{q}, t) \cdot \ddot{\mathbf{q}} - (F_{qq} \dot{\mathbf{q}}) \dot{\mathbf{q}} = 2F_{qt} \dot{\mathbf{q}} - F_{tt}(\mathbf{q}, t) \quad (16)$$

Equations (15) and (16) are obtained by taking one and respectively two time derivatives of the position kinematic constraint equation of Eq.(14). Finally, a set of generalized coordinates is said to be consistent, if it satisfies the position kinematic constraint equations. Likewise, a set of generalized velocities is considered consistent provided for a consistent position configuration, \mathcal{Q} satisfies the velocity kinematic constraint equations of Eq.(15).

Finally, in this document a vector quantity marked with an over-bar indicates that the vector is expressed in a local body reference frame. Note in this context the over-bar for the angular velocity in Eq.(5).

INITIAL CONDITION ANALYSIS

Initial Condition (IC) Analysis is concerned with determining a consistent configuration of the mechanical system model at the beginning of the simulation at time t_0 . During IC analysis, the mechanism must be *assembled* and the velocities of the parts in the mechanism must be *consistent*.

To be assembled, the generalized coordinates \mathbf{q} must satisfy all constraint equations; i. e.,

$$\mathbf{F}(\mathbf{q}, t_0) = \mathbf{0} \quad (17)$$

while for the generalized velocities to be consistent, they must satisfy the velocity kinematic constraint equation

$$\mathbf{F}_q(\mathbf{q}, t) \cdot \dot{\mathbf{q}} = -\mathbf{F}_t(\mathbf{q}, t) \quad (18)$$

1. Position Initial Condition Analysis

During IC analysis, the user might want sometimes for certain reasons to fix the value of some of the generalized coordinates q_{i_1} , q_{i_2} , etc., of the generalized coordinate array in Eq.(8). In other words, if the user prescribes the position of body 3 in the system to be $q_{13} = 0.9$, $q_{14} = -1.0$, $q_{16} = 0$, the solver should assemble the mechanism and in the same time do its best to satisfy the prescribed conditions. This IC analysis is solved in ADAMS via an optimization approach. The constrained optimization problem solved minimizes the cost function

$$f(q_1, \mathbf{K}, q_n) = \frac{1}{2} w_1 (q_1 - q_1^0)^2 + \mathbf{K} + \frac{1}{2} w_n (q_n - q_n^0)^2 \quad (19)$$

subject to the constraint equations $\mathbf{F}(\mathbf{q}, t_0) = \mathbf{0}$. In Eq.(19), the values w_i are weight factors, while q_i^0 can be regarded as generalized coordinates inducing an initial configuration of the system. Notice that this configuration $\mathbf{q}^0 = [q_1^0 \quad q_2^0 \quad \mathbf{K} \quad q_n^0]^T$ need not be consistent.

The user may prescribe that some of the entries in the \mathbf{q}^0 array; i.e., $q_{i_1}^0$, $q_{i_2}^0$, etc., are to be regarded "exact". As it will be justified shortly, the corresponding weights w_{i_1} ,

w_{i_2} , etc., will be given large values; i.e., values like 10^{10} . The remaining weights, namely the ones corresponding to generalized coordinates q_i^0 that the user did not specify are given values of 1. With this, the constrained optimization problem will manage in its solution sequence to keep the user imposed IC values almost unchanged, while focusing on adjusting the "free-to-change" generalized coordinates to find the solution of the problem. Notice that "the solution of the problem" means minimizing the cost function while *satisfying* the constraints.

In matrix notation, the constrained optimization problem reads for $\mathbf{q} \in \mathbf{R}^n$, minimize

$$f(\mathbf{q}) = \frac{1}{2}(\mathbf{q} - \mathbf{q}^0)^T \mathbf{W}(\mathbf{q} - \mathbf{q}^0) \quad (20)$$

subject to

$$\mathbf{F}(\mathbf{q}, t_0) = \mathbf{0} \quad (21)$$

In Eq.(20), \mathbf{W} is a diagonal matrix of weights,

$$\mathbf{W} = \text{diag}(w_1, w_2, \mathbf{K}, w_n) \quad (22)$$

while the constraints of Eq.(21) that must be satisfied in the optimization problem are exactly the position kinematic constraints of Eq.(14).

ADAMS approximates the non-convex optimization problem of Eqs.(20) and (21) by a succession of convex problem that are guaranteed to have a solution, which can be found in one iteration. In this context, the set of non-linear constraint equations of Eq.(21) is linearized in the vicinity of \mathbf{q}^0

$$\mathbf{F}(\mathbf{q}, t_0) = \mathbf{F}(\mathbf{q}^0, t_0) + \mathbf{F}_q(\mathbf{q}^0, t_0)(\mathbf{q} - \mathbf{q}^0) \quad (23)$$

Equation (23) replaces Eq.(21) and with the notation $\mathbf{d} \equiv \mathbf{q} - \mathbf{q}^0$ the now convex optimization problem reads

$$\text{Minimize} \quad f(\mathbf{d}) = \mathbf{d}^T \mathbf{W} \mathbf{d} \quad (24)$$

$$\text{Subject to} \quad \mathbf{F}(\mathbf{q}^0, t_0) + \mathbf{F}_q(\mathbf{q}^0, t_0) \mathbf{d} = \mathbf{0} \quad (25)$$

To solve the convex constrained optimization problem of Eqs.(24) and (25) define the Lagrangian

$$F(\mathbf{d}, \mathbf{l}) = f(\mathbf{d}) + \mathbf{l}^T (\mathbf{F}(\mathbf{q}^0) + \mathbf{F}_q(\mathbf{q}^0)\mathbf{d}) \quad (26)$$

The optimality conditions for this problem are

$$\begin{aligned} \left(\frac{\partial F}{\partial \mathbf{d}} \right)^T &= \mathbf{0} \\ \left(\frac{\partial F}{\partial \mathbf{l}} \right)^T &= \mathbf{0} \end{aligned} \quad (27)$$

which lead to the following linear system of equations:

$$\begin{bmatrix} \mathbf{W} & \mathbf{F}_q^T(\mathbf{q}^0) \\ \mathbf{F}(\mathbf{q}^0) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{d} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -\mathbf{F}(\mathbf{q}^0) \end{bmatrix} \quad (28)$$

Based on Eq.(28), ADAMS computes the value of \mathbf{d} , and given \mathbf{q}^0 computes the solution of the convex optimization problem as

$$\mathbf{q} = \mathbf{q}^0 + \mathbf{d} \quad (29)$$

The configuration induced by the new set of generalized coordinates obtained as in Eq.(29) corresponds to the linearized problem of Eqs.(24) and (25). Therefore, while the solution satisfies the conditions of Eq.(25), it might be that it does not satisfy the original non-linear system constraint equations induced by the system's joints as in Eq.(21). If this is the case, then the configuration \mathbf{q} just obtained is set to be the new \mathbf{q}^0 , and another iteration starting with the linearization of Eq.(23) is carried out.

Typically, after a couple of iterations the solution of the linear convex optimization problem will satisfy the non-linear constraint equations induced by the joints of the mechanical system. The approach fails when the only assumption made by the numerical method involved ceases to hold; i.e., when the linearization in Eq.(23) is far from approximating the non-linear manifold induced by the original constraint equations. It is worth noting here that even when the manifold is highly non-linear, the solution sequence will converge provided the starting point \mathbf{q}^0 is close enough to the final solution. This is the reason for which it is essential for the algorithm to have a good starting point.

Finally, a word on how the weights w_i of Eq.(19) factor in to keep the user defined initial conditions to their prescribed value. To better see how these weights kick in, consider a case with only one constraint that needs to be satisfied. Likewise, assume that the system has 2 generalized coordinates x and y , and that the constraint equation they must satisfy is $\Phi(x, y) = x^2 - y = 0$. Obviously this constraint equation is satisfied by an infinite number of pairs like (2,4), (2.5, 6.25), etc., but in this simple case the user would like to specify that the value of x is $x^0 = 1$, while the value y^0 is free to change. As the value for y^0 is not prescribed, assume our initial guess takes $y^0 = 6$. Since the user prescribed a value for x , the weight associated with this generalized coordinates is large; i.e., $w_1 = 10^{10}$. On the other hand, since there is no condition imposed on the second generalized coordinate, $w_2 = 1$. Notice that “prescribed” in the discussion above refers to coordinates specified to be *exact* in the ADAMS modeling language (the adm files).

With this, the matrix of Eq.(28) assumes the form

$$\begin{bmatrix} 10^{10} & 0 & 2 \\ 0 & 1 & -1 \\ 2 & -1 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \mathbf{I} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

Then $d_1 = 10^{-9}$, $d_2 = -5$, $\mathbf{I} = -5$, and

$$\begin{aligned} x_{IC} &= 1 + 10^{-9} \approx 1 \\ y_{IC} &= 6 - 5 = 1 \end{aligned}$$

As it can be easily verified, $\Phi(x_{IC}, y_{IC}) = 2 \cdot 10^{-9} + 10^{-18}$, that is, the non-linear constraint equation is very well satisfied, and the correction applied in the user prescribed initial condition x is of the order 10^{-9} which for most practical purposes is negligible. Thus, the weights w_i is the means through which the algorithm is determined to be biased in finding a solution by changing a certain subset of generalized coordinates rather than another set whose values were prescribed by the user.

On a final note, it is worth noting that during position IC analysis ADAMS checks for the sanity of the constraint equations induced by the joints and motions present in the

model. During redundant constraint analysis some of the constraint equations might turn out to be redundant. In a benign situation a redundant constraint is consistent. A quick example of such a case is when ADAMS is presented with one constraint equation that looks like

$$x^2 - y = 0$$

and then with a different constraint equation that reads like

$$2x^2 - 2y = 0$$

The latter clearly does not add anything to the picture, since when the first equation is satisfied the second one is automatically satisfied too. Thus, the second equation is redundant, but it is consistent and throughout the simulation ADAMS will monitor this equation to make sure that the redundant constraint continues to be consistent.

The malign case is when the second equation would read like

$$2x^2 - 2y = 1$$

If this is the case, the two constraint equations can not be simultaneously satisfied no matter what values the generalized coordinates x and y assume. This is the case of incompatible redundant constraint equations. ADAMS will do a LU factorization with full pivoting and inform the user about encountering this situation. ADAMS will stop simulation upon finding incompatible redundant constraints because from a modeling perspective there is something qualitatively wrong with the system being simulated.

Note that redundant constraints are mostly encountered when too many joints are used to model the mechanical system and therefore the number of constraint equations induced by these joints exceeds the number of generalized coordinates of the model.

In what follows, two or more constraint equations will be called independent if they are not redundant.

2. *Velocity Initial Condition Analysis*

The velocity IC analysis is a direct and simple application of the algorithm employed for the position IC analysis. It is direct because it is applied exactly as presented before, and it is simple because the constraint equations that need to be satisfied are already in a linear form. Therefore, there is no need to linearize them as was

the case with the position constraint equations (see Eq.(23)) and the solution is guaranteed to be found in one iteration. Thus, the convex constrained optimization problem solved to retrieve the initial velocities $\dot{\mathbf{q}}_0$ minimizes the cost function

$$f(\dot{\mathbf{q}}_0, \mathbf{K}) = \frac{1}{2} (\dot{\mathbf{q}}_0 - \dot{\mathbf{q}}_0^T)^T \mathbf{W} (\dot{\mathbf{q}}_0 - \dot{\mathbf{q}}_0^T) \quad (30)$$

subject to the always linear velocity kinematic constraint equations of Eq.(15):

$$\mathbf{F}_q(\mathbf{q}, t_0) \cdot \dot{\mathbf{q}}_0 + \mathbf{F}_t(\mathbf{q}, t_0) = \mathbf{0} \quad (31)$$

As for IC analysis, the weight diagonal matrix \mathbf{W} has some very large positive entries that ensure this time around that the corresponding user prescribed initial velocities are not changed by the optimization algorithm. From here on, the same procedure previously used for position IC analysis is employed. Note that because of the linearity of the velocity kinematic constraint equations the algorithm is guaranteed to converge in one iteration.

3. Force and Acceleration Initial Condition Analysis

In the absence of friction forces, the acceleration IC analysis requires the solution of the linear system assembled from the equations of motion (EOM) and the acceleration kinematic constraint equations of Eq.(12). The resulting system has the form

$$\begin{bmatrix} \mathbf{M} & \mathbf{F}_q^T(\mathbf{q}^0) \\ \mathbf{F}_q(\mathbf{q}^0) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_0 \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \mathbf{t} \end{bmatrix} \quad (32)$$

Note that this is a linear system, and the iterative process involved typically converges in one iteration. The user does not directly prescribe any initial acceleration for the force/acceleration IC analysis. The reaction force and initial accelerations are evaluated based on the computed initial position, initial velocity, and the applied force acting on the system at the initial time. For a more detailed explanation of how the EOM are obtained (the first row in the Eq.(32)), see the Section on Dynamic Analysis in ADAMS.

Besides $\ddot{\mathbf{q}}_0$, the solution of the linear system above also provides the Lagrange multipliers \mathbf{l} . The constraint reaction force and torque induced by joint j that connects body i to ground or other part in the system are computed as

$$\mathbf{F}^C = - \left(\frac{\partial \mathbf{F}^{(j)}}{\mathbf{v}_i} \right)^T \mathbf{1}^{(j)} \quad (33)$$

$$\mathbf{T}^C = - \left(\frac{\partial \mathbf{F}^{(j)}}{\mathbf{w}_i} \right)^T \mathbf{1}^{(j)} \quad (34)$$

In Eqs.(33) and (34) the superscript C indicates that the quantities are expressed in a Cartesian coordinate system; \mathbf{v}_i is the Cartesian velocity of body i ; \mathbf{w}_i is the global angular velocity; $\mathbf{F}^{(j)}$ represent the set of constraint equations induced by joint j .

KINEMATIC ANALYSIS

Typically, for a kinematic analysis to be carried out a number of independent constraint equations equal to the number of generalized coordinates in the model must be prescribed. For the mechanism to actually change its configuration in time, some of these constraints must be motions; i.e., they should depend on time.

1. *Position Level Kinematic Analysis*

Given the position of the system at time t_0 , the problem here is to determine the position at time $t_1 > t_0$. Because of the non-linear nature of the constraint equations in Eq.(14), a Newton-Raphson iterative method is used in ADAMS to compute \mathbf{q}_1 at time t_1 . To understand how this method works and what its limitations are first note that it is obtained from a Taylor expansion based linearization of the non-linear constraint equations:

$$\mathbf{F}(\mathbf{q}_1, t_1) = \mathbf{F}(\mathbf{q}_0, t_1) + \mathbf{F}_q(\mathbf{q}_0, t_1)(\mathbf{q}_1 - \mathbf{q}_0) \quad (35)$$

Since the number of constraints is equal to the number of generalized coordinates, first note that the matrix $\mathbf{F}_q(\mathbf{q}_0, t_1)$ is square. As the constraint equations were assumed to be independent this matrix is also invertible. Based on an explicit integrator an initial starting configuration $\mathbf{q}_1^{(0)}$ is determined, and the iterative algorithm proceeds at each iteration $j \geq 0$ by finding the correction $\mathbf{D}^{(j)}$

$$\mathbf{F}_q(\mathbf{q}_0, t_1)\mathbf{D}^{(j)} = -\mathbf{F}(\mathbf{q}_1^{(j)}, t_1) \quad (36)$$

Then, $\mathbf{q}_1^{(j+1)} = \mathbf{q}_1^{(j)} + \mathbf{D}^{(j)}$, with the iterative process being stopped when the correction $\mathbf{D}^{(j)}$ and/or the residual $\mathbf{F}(\mathbf{q}_1^{(j)}, t_1)$ become small enough.

As for the position IC analysis, ADAMS can fail to find \mathbf{q}_1 if while with an initial guess far enough from the consistent solution \mathbf{q}_1 , the linearization turns out to be a poor approximation of the non-linear manifold. In these situations, the remedy lies in

decreasing the simulation step-size, and thus causing \mathbf{q}_1 to lie closer on the manifold to the last consistent configuration \mathbf{q}_0 .

2. *Velocity Level Kinematic Analysis*

Velocity kinematic analysis is straightforward as the velocity kinematic constraint equations are linear in velocity. With \mathbf{q}_1 already available after the position kinematic analysis, the non-singular matrix $\mathbf{F}_q(\mathbf{q}_1, t_1)$ is evaluated and the linear system of Eq.(15) is solved for the new velocity $\dot{\mathbf{q}}_1$.

3. *Acceleration Level Kinematic Analysis*

Acceleration kinematic analysis is immediate, as at time t_1 it is found as the solution of the linear system of Eq.(16). Notice that the same matrix that is factored for velocity kinematic analysis is used for a forward/backward substitution sequence to solve for the generalized accelerations $\ddot{\mathbf{q}}$.

Once $\ddot{\mathbf{q}}$ is available, the Lagrange multipliers associated with the set of constraints acting on the system are computed as the solution of the linear system

$$\mathbf{F}_q^T \mathbf{l} = \mathbf{F} - \mathbf{M} \ddot{\mathbf{q}} \quad (37)$$

This equation is identical to the first row of the linear system of Eq.(32) and in fact represents precisely the equations of motion. More information on how Eq.(37) is obtained is provided in the next Section.

DYNAMIC ANALYSIS

1. *Nomenclature, Conventions, Definitions.*

In addition to the definitions and notations introduced at the beginning of this document, the following quantities will be used in formulating the rigid body equations of motion.

\mathbf{M} - generalized mass matrix

$\bar{\mathbf{J}}$ - generalized inertia matrix expressed about the principal local reference frame

K - kinetic energy, defined as

$$K = \frac{1}{2} \mathbf{u}^T \mathbf{M} \mathbf{u} + \frac{1}{2} \bar{\mathbf{w}}^T \bar{\mathbf{J}} \bar{\mathbf{w}} \quad (38)$$

$\mathbf{l} \in \mathbf{R}^m$ - array of Lagrange multipliers. The number m of Lagrange multipliers is given by the number of constraint equations induced by joints connecting a body to other bodies in the system.

$\mathbf{F}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} \mathbf{f} \\ \bar{\mathbf{n}} \end{bmatrix} \in \mathbf{R}^6$ - the vector of applied forces

$\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbf{R}^6$ - the generalized force acting on the body. Obtained by projecting the applied force \mathbf{F} upon the generalized coordinates. Typically,

$$\mathbf{Q} = \begin{bmatrix} (\mathbf{P}^P)^T \mathbf{f} \\ (\mathbf{P}^R)^T \bar{\mathbf{n}} \end{bmatrix} \quad (39)$$

where with \mathbf{v}^P being the velocity of the point of application \mathbf{P} of the external force \mathbf{F} , the projection operators are computed like

$$\mathbf{P}^P = \frac{\partial \mathbf{v}^P}{\partial \mathbf{u}} \quad (40)$$

$$\mathbf{P}^R = \frac{\partial \bar{\mathbf{w}}}{\partial \mathbf{z}} \quad (41)$$

2. Formulation of Equation Of Motion (EOM) in ADAMS

Provided in any Dynamics book, the Lagrange formulation of the equations of motion leads to the following second order differential equations

$$\frac{d}{dt} \left[\left(\frac{\partial K}{\partial \dot{\mathbf{q}}} \right)^T \right] - \left(\frac{\partial K}{\partial \mathbf{q}} \right)^T + \mathbf{F}_q^T \mathbf{l} = \mathbf{Q} \quad (42)$$

Considering the choice of generalized coordinates in ADAMS; i.e., the definition of \mathbf{q} as in Eq.(3), Eq.(42) is rewritten for a rigid body as

$$\frac{d}{dt} \begin{bmatrix} \left(\frac{\partial K}{\partial \mathbf{u}} \right)^T \\ \left(\frac{\partial K}{\partial \mathbf{z}} \right)^T \end{bmatrix} - \begin{bmatrix} \left(\frac{\partial K}{\partial \mathbf{p}} \right)^T \\ \left(\frac{\partial K}{\partial \mathbf{e}} \right)^T \end{bmatrix} + \begin{bmatrix} \mathbf{F}_p^T \mathbf{l} \\ \mathbf{F}_e^T \mathbf{l} \end{bmatrix} = \begin{bmatrix} (\mathbf{P}^P)^T \mathbf{f} \\ (\mathbf{P}^R)^T \bar{\mathbf{n}} \end{bmatrix} \quad (43)$$

It is worth pointing out that when dealing with a full system of rigid bodies connected through joints, the system Equation Of Motions (EOM) are obtained by simply stacking together the EOM for the bodies in the system.

Since

$$\frac{d}{dt} \left(\frac{\partial K}{\partial \mathbf{u}} \right)^T = \mathbf{M} \dot{\mathbf{u}} \quad (44)$$

$$\left(\frac{\partial K}{\partial \mathbf{p}} \right)^T = \mathbf{0} \quad (45)$$

with the angular momenta defined as

$$\mathbf{G} \equiv \frac{\partial K}{\partial \mathbf{z}} = \mathbf{B}^T \bar{\mathbf{J}} \mathbf{B} \mathbf{z} \quad (46)$$

the EOM of Eq.(43) are reformulated in ADAMS as

$$\begin{aligned} \mathbf{M} \dot{\mathbf{u}} + \mathbf{F}_p^T \mathbf{l} &= (\mathbf{P}^P)^T \mathbf{f} \\ \mathbf{G} \frac{\partial K}{\partial \mathbf{e}} + \mathbf{F}_e^T \mathbf{l} &= (\mathbf{P}^R)^T \bar{\mathbf{n}} \end{aligned} \quad (47)$$

The first order differential equations above are called in what follows kinetic differential equations, and they indicate how external forces determine the time variation of the translational and angular momenta.

Finally, the time variation of the generalized coordinates is related to the translational and angular momenta by means of the kinematic differential equations. By assembling the kinetic and kinematic differential equations ADAMS generates a set of 15 equations that provide the information necessary to find a numerical solution for the dynamic analysis of a mechanical system. These equations are as follows:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{F}_p^T \mathbf{1} - (\mathbf{P}^P)^T \mathbf{f} = \mathbf{0} \quad (48)$$

$$\mathbf{G} - \mathbf{B}^T \mathbf{J} \mathbf{B} \mathbf{z} = \mathbf{0} \quad (49)$$

$$\mathbf{e} \frac{\partial K}{\partial \mathbf{e}} + \mathbf{F}_e^T \mathbf{1} - (\mathbf{P}^R)^T \bar{\mathbf{n}} = \mathbf{0} \quad (50)$$

$$\mathbf{p} \mathbf{z} - \mathbf{u} = \mathbf{0} \quad (51)$$

$$\mathbf{e} \mathbf{z} = \mathbf{0} \quad (52)$$

3. Numerical Solution for Dynamic Analysis. Jacobian Information Computation.

Equations (48) through (52) indicate how relevant variables change in time. What is missing in this picture is the fact that the solution of this system of differential equations must also satisfy the kinematic constraint equations of Eqs.(14) through (16). From a numerical standpoint, this is what makes the dynamic analysis of a mechanical system the most difficult type of simulation.

There is a multitude of methods for solving the assembly differential+constraint equations. This document is not concerned with these methods and it only provides a glimpse at what ADAMS does to address this problem. In this context it is worth mentioning that this assembly: differential and constraint equations forms what is called a set of Differential-Algebraic Equations (DAE). A DAE has an index associated with it, and rule goes that the higher the index, the more challenging the numerical solution of the DAE becomes. In particular, the DAE induced by the dynamic analysis problem in mechanical system simulation has index 3, which is considered high.

In the ADAMS Fortran solver there are two more reliable methods for solution. The most common one is a direct index 3 DAE solver, in which associated to the differential equations induced by (48) through (52) are the position kinematic constraint

equations of Eq.(14). In this approach the velocity and acceleration level kinematic constraint equations are periodically enforced. This is how the solver GSTIFF in ADAMS works.

A second, more refined algorithm reduces the original index 3 problem to an analytically but yet numerically different index 2 DAE problem. Thus, instead of considering the position, the velocity level kinematic constraint equations of Eq.(15) are solved for along with the kinematic differential equations. In the Fortran solver, this algorithm is called SI2, and while typically slower than the index 3 approach it turns out to be more accurate and robust.

In what follows the index 3 approach is presented in a reasonable amount of detail. In an attempt to keep the presentation simple, the index 3 DAE will be integrated via an order 1 implicit integration formula. This formula is the backward Euler formula - an one step, A stable algorithm that qualitatively captures all the relevant details characteristic to higher order methods. Backward Euler integration formula replaces the derivative $\dot{\mathbf{y}}$ at time t_1 with

$$\dot{\mathbf{y}} = \frac{1}{h} \mathbf{y}_1 - \frac{1}{h} \mathbf{y}_0 \quad (53)$$

Based on Eq.(53), an Initial Value Problem (IVP) $\dot{\mathbf{y}} = \mathbf{g}(\mathbf{y}, t)$, $\mathbf{y}(t_0) = \mathbf{y}_0$ is solved by finding $\mathbf{y}(t_1)$ at time $t_1 > t_0$ as the solution \mathbf{y}_1 of the discretization algebraic non-linear system

$$\frac{1}{h} \mathbf{y}_1 - \frac{1}{h} \mathbf{y}_0 - \mathbf{g}(t_1, \mathbf{y}_1) = \mathbf{0} \quad (54)$$

The system of equations in Eq.(54) is called a “discretization system” since the derivative in the original IVP problem was “discretized” using the integration formula of Eq.(53). Since almost always the function \mathbf{g} is non-linear, a non-linear algebraic system needs to be solved to retrieve \mathbf{y}_1 . This is done in ADAMS by using a Newton-Raphson type iterative algorithm.

Based on the implicit Euler discretization formula introduced above, all the first order time derivatives that appear in the equations of motion in Eqs.(48) through (52) are discretized to produce a set of algebraic non-linear equations. In the index 3 approach,

the position kinematic constraint equations are appended to these equations, along with the force function definition \mathbf{F} and \mathbf{T} . This appending of the force functions is done with the sole purpose of increasing the number of unknowns and thus inducing a larger but yet sparser Jacobian matrix. Thus, after the implicit Euler based discretization Eqs.(14), and (48) through (52) along with the force/torque definition equations assume the following form:

$$\begin{aligned}
\frac{1}{h}\mathbf{M}\mathbf{u} - \frac{1}{h}\mathbf{M}\mathbf{u}_0 + \mathbf{F}_p^T \mathbf{l} - (\mathbf{A}^P)^T \mathbf{f} &= \mathbf{0} \\
\mathbf{G} - \mathbf{B}^T \bar{\mathbf{J}} \mathbf{B} \mathbf{z} &= \mathbf{0} \\
\frac{1}{h}\mathbf{G} - \frac{1}{h}\mathbf{G}_0 - \left(\frac{\partial K}{\partial \mathbf{e}}\right)^T + \mathbf{F}_e^T \mathbf{l} - (\mathbf{A}^R)^T \bar{\mathbf{n}} &= \mathbf{0} \\
\frac{1}{h}\mathbf{p} - \frac{1}{h}\mathbf{p}_0 - \mathbf{u} &= \mathbf{0} \\
\frac{1}{h}\mathbf{e} - \frac{1}{h}\mathbf{e}_0 - \mathbf{z} &= \mathbf{0} \\
\mathbf{F}(\mathbf{p}, \mathbf{e}, t_1) &= \mathbf{0} \\
\mathbf{f} - \mathbf{F}(\mathbf{u}, \mathbf{z}, \mathbf{p}, \mathbf{e}, \mathbf{f}, \bar{\mathbf{n}}, t_1) &= \mathbf{0} \\
\bar{\mathbf{n}} - \mathbf{T}(\mathbf{u}, \mathbf{z}, \mathbf{p}, \mathbf{e}, \mathbf{f}, \bar{\mathbf{n}}, t_1) &= \mathbf{0}
\end{aligned} \tag{55}$$

The unknowns in this non-linear system are \mathbf{u} , \mathbf{G} , \mathbf{z} , \mathbf{p} , \mathbf{e} , \mathbf{l} , \mathbf{f} , $\bar{\mathbf{n}}$. The subscript 1 indicating the time step was dropped for convenience.

Introducing the array

$$\mathbf{y} = \begin{bmatrix} \mathbf{u} \\ \mathbf{G} \\ \mathbf{z} \\ \mathbf{p} \\ \mathbf{e} \\ \mathbf{l} \\ \mathbf{f} \\ \bar{\mathbf{n}} \end{bmatrix} \tag{56}$$

the non-linear system of Eq.(55) is rewritten as

$$\mathbf{Y}(\mathbf{y}) = \mathbf{0} \tag{57}$$

A Newton-Raphson type algorithm finds the solution of this system. Thus, first a prediction $\mathbf{y}^{(0)}$ of the solution is provided, typically by using a predictor constructed around an explicit integrator. Once an initial guess of the solution is provided, iterations

$$\begin{aligned} \mathbf{Y}_y(\mathbf{y}_0) \mathbf{D}^{(j)} &= -\mathbf{Y}(\mathbf{y}^{(j)}) \\ \mathbf{y}^{(j+1)} &= \mathbf{y}^{(j)} + \mathbf{D}^{(j)} \end{aligned} \tag{58}$$

are carried out until the residual $\mathbf{Y}(\mathbf{y}^{(j)})$ and/or the correction $\mathbf{D}^{(j)}$ are small enough.

The last piece in the puzzle is what turns out to be also the most costly one; i.e., the computation of the Jacobian $\mathbf{Y}_y(\mathbf{y}_0)$, which is obtained from Eq.(55). With the notation introduced in Eq.(56), the expression of the Jacobian $\mathbf{Y}_y(\mathbf{y}_0)$ is provided in the Appendix.

The same remarks made in conjunction with the iterative Newton-Raphson algorithm used for IC analysis and for Kinematic analysis are applicable here. Thus, if the initial guess; i.e., the predicted value of $\mathbf{y}^{(0)}$ is too far away from the solution, the iterative process might fail to converge. This is more likely to happen with dynamic analysis than with other types of analysis, as it is clear that the system that needs to be solved at each integration step is highly non-linear.

If the iterative process fails, the integration step-size is decreased and another step is attempted. ADAMS users are familiar in this context with messages informing them that the step-size was decreased too much, and yet the convergence was not attained. Getting such a message is a bad omen, as typically the user will have to revisit the model, make modifications in the simulation defining parameters, or to try a different integrator like SI2 for example.

While talking about the integration Jacobian, it is the right place to mention yet another piece of information that the solver supplies to the user, namely the need for refactorization. As can be seen in the Appendix, the Jacobian has a time-invariant sparsity pattern. When solving for the corrections $\mathbf{D}^{(j)}$ the Jacobian needs to be factored. The pivots in the LU factorization are chosen at the beginning of the simulation and they are re-used upon a new call for the solution of this system. Especially for long simulations, it turns out that after some time the initial pivot sequence results in a singular

matrix. This requires a refactorization and possibly a decrease of the integration step-size. The user is flagged when the solver runs into such a scenario.

Finally, although the discretization formula used to convey the dynamic analysis solution message was backward Euler it conceptually captures the essence of the ADAMS solution sequence. ADAMS typically uses higher order integrators whenever the signals sent over by the problem being solved suggest that this would improve performance. The expression of the integration Jacobian is qualitatively the same, with very minor and insignificant changes – for example the denominator of the fraction $1/h$ would become $1/(hb)$, where \mathbf{b} is an integration formula specific coefficient. What is important to remember here is that the use of a more sophisticated integration formula serves in the end the same purpose, namely to replace a first order time derivative with a linear combination of future and past values of the unknown, which is what backward Euler formula does in a very basic way through Eq.(54).

STATICS AND QUASI-STATICS ANALYSIS

Statics and Quasi-Statics analysis in ADAMS is merely a simplified case of Dynamic Analysis. This is because what the solver currently does is to set the coefficient $1/h$ (or $1/(h\mathbf{b})$ for more complex multi-step methods) to zero. This effectively implies that the value of the time derivative in Eq.(53) becomes zero. This is how equilibrium is perceived – there is no change in time of any of the unknowns, and therefore their time derivatives are all zero.

As a parenthesis here, in ADAMS if they wish the users can set the coefficient above to some very small but yet non-zero value. This is to allow the algorithm to handle neutral equilibrium configurations without running into singular Jacobian matrices.