Forward Dynamics, Elimination Methods, and Formulation Stiffness in Robot Simulation

Uri M. Ascher*, Dinesh K. Pai†and Benoit P. Cloutier‡
Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1Z4, Canada

May 27, 1996

Abstract

The numerical simulation problem of tree-structured multibody systems, such as robot manipulators, is usually treated as two separate problems: (i) the forward dynamics problem for computing system accelerations, and (ii) the numerical integration problem for advancing the state in time. The interaction of these two problems can be important and has led to new conclusions about the overall efficiency of multibody simulation algorithms (Cloutier *et al.*, 1995). In particular, the fastest forward dynamics methods are not necessarily the most numerically stable, and in ill-conditioned cases may slow down popular adaptive step-size integration methods. This phenomenon is called "formulation stiffness."

In this paper, we first unify the derivation of both the composite rigid body method (Walker & Orin, 1982) and the articulated-body method (Featherstone, 1983; Featherstone, 1987) as two elimination methods to solve the same linear system, with the articulated body method taking advantage of sparsity. Then the numerical instability phenomenon for the composite rigid body method is explained as a cancellation error that can be avoided, or at least minimized, when using an appropriate version of the articulated body method. Specifically, we show that a variant of the articulated-body method is better suited to deal with certain types of ill-conditioning than the composite rigid body method. The unified derivation also clarifies the underlying linear algebra of forward dynamics algorithms and is therefore of interest in its own accord.

*Supported in part by an NSERC Research Grant.

†Supported in part by grants from NSERC, B.C. ASI, IRIS, and a CSA Stear8 contract through MPB Technologies.

‡Supported in part by NSERC and FCAR fellowships. Currently with MPB Technologies Inc., Pointe Claire, Canada.

1 Introduction

Simulation of robot dynamics has been used in off-line tasks such as robot design testing, robot programming, and controller validation. Recently, the need for real-time performance has become important for on-line applications such as virtual environments for operator training, predictive displays for time-delayed teleoperation, and development of advanced robot control schemes. Hence it has become more important to understand and compare the actual performance of simulation algorithms.

This paper focuses on algorithms for simulating the motion of robot manipulators without closed loops (for example robots whose motion is not constrained by contacts with the environment). For the simulation of such robots, the elimination method leads to an ODE formulation of the equations of motion:

$$\tau = \mathcal{M}(q)\ddot{q} + c(q, \dot{q})$$

where:

- τ is the $N \times 1$ vector of torques (forces) applied by the joint actuators,
- q is the $N \times 1$ vector of joint variables ($\dot{q}$ and $\ddot{q}$ are the joint velocities and accelerations),
- $\mathcal{M}$ is the $N \times N$ joint-space inertia matrix (JSIM) or generalized inertia matrix,
- c is the $N \times 1$ bias vector representing the torques (forces) due to gravity, centrifugal and Coriolis accelerations, and any external moments and forces acting on the end effector of the manipulator.

Here, N is the number of degrees of freedom.

The simulation problem requires that the joint trajectory of a robot be determined given prior knowledge of the torques and forces applied by the actuators and the initial state of the robot. The simulation problem is usually divided into two subproblems.

- *The forward dynamics problem*: computing the joint accelerations given the actuator forces and torques.
- *The motion integration problem*: computing the joint trajectory (joint positions and velocities) given the joint accelerations and initial conditions.

In recent years, many different algorithms have been proposed for solving the forward dynamics problem, ranging in computational complexity from $O(N^3)$ (e.g. the composite rigid body method (CRBM) (Walker & Orin, 1982)) to $O(N)$ (e.g. the articulated-body method (ABM) (Featherstone, 1983; Featherstone, 1987)). Operation counts of forward dynamics algorithms have become one of the most commonly used means of comparing the performance of different algorithms, i.e. the two parts of the simulation problem are typically treated as completely separate problems having no effect on one another.

However, the computational complexity of forward dynamics is only half the story, which cannot always be considered separately from the following motion integration. For one thing, some special motion integration algorithms (e.g. (Bock & von Schwerin, 1993; Ascher & Lin, 1995)) may affect the comparison between forward dynamics algorithms. Moreover, different numerical stability characteristics of forward dynamics algorithms may affect the performance of popular adaptive step-size integration methods. The latter is the focal point of the present article.

The difference between the stability characteristics of various algorithms becomes important when they are applied to systems which have ill-conditioned joint space inertia matrices. Most industrial robot arms do not have such an ill-conditioning, and this perhaps explains why the problem has not been noticed before. However, ill-conditioned systems do occur in several important situations. For example, the Stanford Arm is known to exhibit condition numbers as high as 11934 (Angeles & Ma, 1988). Noticeable ill-conditioning may arise when simulating long chains; when simulating a biological system such as a walking person, where a small toe in contact with the ground is attached to a much larger foot, followed by a large leg and an even larger body; or when simulating a light weight, long reach manipulator in a microgravity environment with a large payload at its end. We also note that if double precision is used in the simulation, the instability phenomenon will be observed at larger condition numbers (we make this more precise below); however this added security is at the expense of longer computation time on most computers today.

Example 1 *A simple 2DOF two-link serial manipulator with parallel revolute joints is used to illustrate the effect of a poorly conditioned JSIM on individual computations of the forward dynamics. Ill-conditioning is obtained by increasing the length* (l_1) *and mass* (m_1) *of the link closer to the tip while keeping the length and mass of the base-connected link constant*[1].

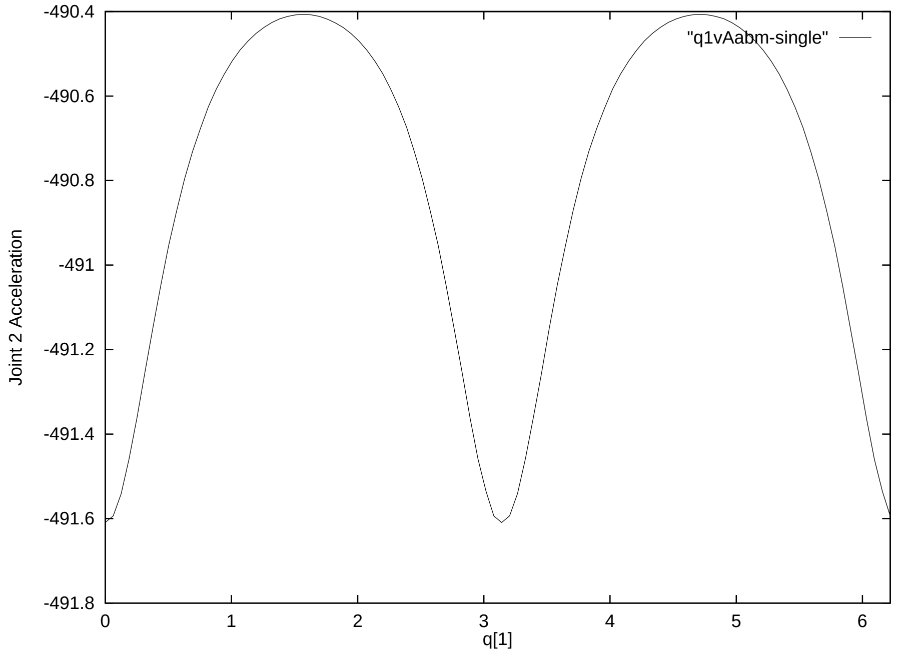
To illustrate the effect of joint configuration on the condition number, we perform a sweep of the possible configurations of each two link chain by varying the angle of the tip end joint (q_1) *from 0 to* 2π. *For each configuration of the chain, we compute the joint accelerations using the articulated-body method (ABM) and the composite rigid body method (CRBM), using single precision floating point artihmetic*[2]. *Cholesky decomposition was used to solve the linear system in the CRBM*[3]. *The computed accelerations of the first joint using the two forward dynamics algorithms is depicted in Fig. 1.*

Specifically, Fig. 1 depicts the forward dynamics computed under the following conditions: zero initial joint angles and velocities, gravity $9.81 m/s^2$ *acting orthogonal*

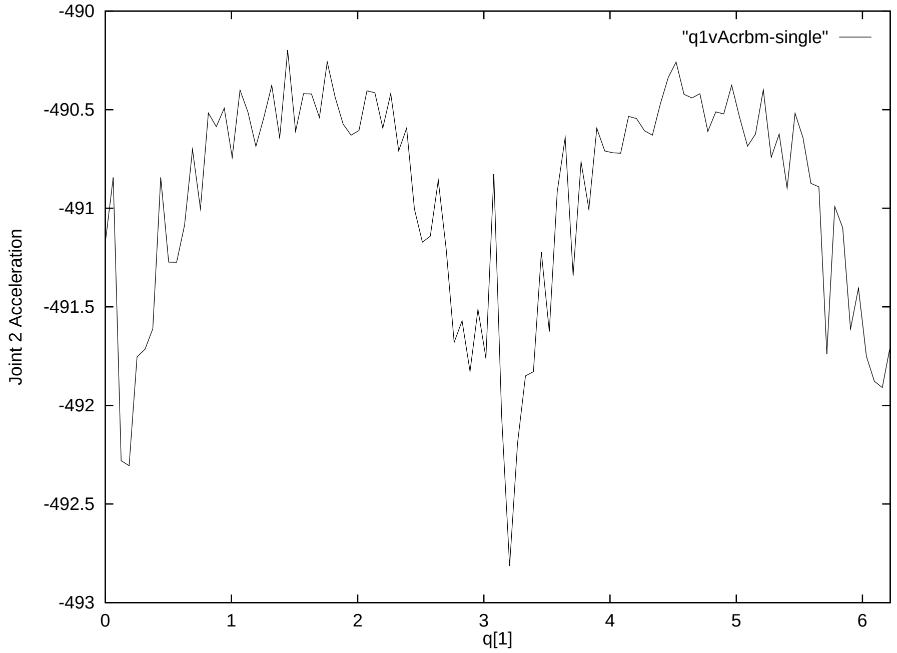
[1]It turns out to be convenient later on to number chain bodies from tip to base, as e.g., in (Jain, 1991), so we adopt this convention already in this example.

[2]In (Cloutier *et al.*, 1995) we have considered the double precision case in more detail.

[3]Similar results were obtained using LU decomposition with pivoting.



Articulated-body Method



Composite Rigid-Body Method

Figure 1: Comparison of calculated joint 2 (base joint) accelerations. Note the smoothness of the ABM solution.

r	n_{ABM}	n_{CRBM}	$O(\kappa_{max})$
0.1000	3169	3041	10^2
0.0500	4623	5109	10^3
0.0100	8189	8082	10^4
0.0050	10534	10420	10^5
0.0040	11170	11835	10^5
0.0035	12375	13792	10^5
0.0030	11990	18986	10^5
0.0025	12170	35406	10^5
0.0020	12779	44294	10^5
0.0018	12418	86656	10^5
0.0015	12860	(fail)	10^5

Table 1: Number of forward dynamics evaluations (n_{ABM} and n_{CRBM}) required to simulate 10 seconds of motion for various 2 link chains of increasing condition number. The table displays the link length ratio ($r = l_2/(l_1 + l_2)$), the number of forward dynamics function evaluations for each method (n_{ABM}, n_{CRBM}), and the order of magnitude of the maximum condition number (κ_{max}) of $\mathcal{M}$. For these simulations, a fourth order Runge-Kutta adaptive step size integrator was used, with step size accuracy of 1e-4.

to the length of the arm, no friction, no applied actuator torques, uniform link widths (= 0.02m), first (distal) link of length $l_1 = 2.0m$ and mass $m_1 = 10.0kg$, and second (proximal) link of length $l_2 = 0.02m$ and mass $m_2 = 0.1kg$. Note that the links are numbered starting from the free end following (Jain, 1991) (see §2).

Observe that the AB method yields a much smoother acceleration than the CRB method. When this is fed to an adaptive ODE motion integrator the result is that much smaller integration steps are needed with the CRBM to maintain a given accuracy, because of the roughness of the calculated accelerations; hence the ABM yields an overall faster algorithm, even though at each point t the CRBM is faster (note that $N = 2$ is very small).

Table 1 illustrates this with the results of 10 second simulations of various two link chains. Each system starts at rest from the configuration $q_1 = q_2 = 0.0$. Accelerations are computed under similar conditions to those described above, with the exception that, in this case, the ratio of the link lengths and masses is varied while keeping the same total mass and length (this ensures that each system has the same total energy). The links are rectangular with link lengths $l_1 + l_2 = 1.0m$, widths $h_1 = h_2 = 0.1m$, and masses $m_1 + m_2 = 1.0kg$, with $l_1 = m_1$ and $l_2 = m_2$. It is clear from Table 1 that for ill-conditioned systems the ABM becomes an overall more effective method than the CRBM, even for small N.

$\square$

More details on this example are given in (Cloutier *et al.*, 1995). The validity of the simulation results was confirmed by comparing our results to those of a commercially available simulation package (SD/FAST[4]). Our purpose here is to explain the observed results.

For this purpose we must first give these methods a unified, algebraic description. This is done in §2, following (Jain, 1991) and (Lubich *et al.*, 1992). We show that, using spatial representation, the different forward dynamics algorithms can in fact be viewed as different Gaussian elimination methods to solve the same system of algebraic (linear) equations. The ABM simply uses the sparsity structure of the system more effectively when N is large, whereas it requires more overhead than CRBM when N is small (say $N < 7$).

In related work, (Ellis *et al.*, 1992; Ellis & Ricker, 1994) considered the numerical stability of articulated body algorithms. They showed that the ABM can be considered as a modified Gaussian elimination performed on a dense "inertial supermatrix," and therefore stable as long as the positive definiteness is maintained. However, they did not observe any numerical superiority of the ABM over other methods such as CRBM and it is not obvious to us how to apply the analysis there to explain formulation stiffness. Our derivation is not related to the inertial supermatrix but rather is performed on a sparse "Kuhn-Tucker"-type matrix derived from the dynamics of individual bodies and their constraints (e.g. (Lubich *et al.*, 1992)). This corresponds to formulating the problem as a system of differential-algebraic equations (DAEs) rather than ODEs. In particular, once the matrix is formed, we formally derive both the CRBM and ABM from linear algebra considerations only.

Armed with the algorithm descriptions of §2 we then explain the source of numerical instability in the CRBM in §3. The superiority of the ABM in this respect arises from a better order in which the pivotal elements in the elimination matrix are formed, which allows for less cancellation error than for the CRBM. With the ABM, too, attention must be paid to the order in which operations are performed. The usual method (e.g. (Featherstone, 1987)) performs well in ill-conditioned cases only if local coordinate systems are essentially aligned with the principal directions of motion. For the general case a variant of ABM called *modified ABM* is developed in §4.

[4]SD/FAST is a trademark of Symbolic Dynamics, Inc., 561 Bush Street, Mountain View, CA 94041 USA.

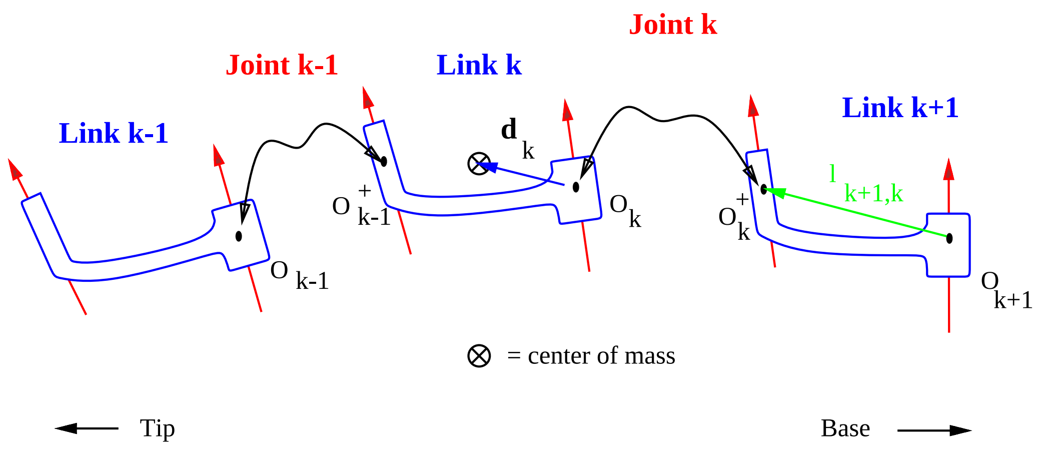


Figure 2: Link k of chain

2 Unified problem formulation and solution methods

We will use essentially the notation developed by Rodriguez, Jain, and Kreutz-Delgado (Rodriguez *et al.*, 1991; Jain, 1991), which in turn uses a variant of the spatial vector notation introduced by (Featherstone, 1987). Consider, for the sake of simplicity, a chain of n links numbered from tip to base. Concentrating on the kth link and kth joint (see figure 2), we define

$$M_k = \begin{pmatrix} \mathcal{J}_k - m_k\tilde{d}_k^{\,2} & m_k\tilde{d}_k \\ -m_k\tilde{d}_k & m_k I \end{pmatrix}$$

- m_k – mass of kth body
- $\mathcal{J}_k$ – moment of inertia about center of mass
- O_k – reference location of kth joint on kth link
- d_k is the displacement from O_k to the center of mass; $\tilde{d}_k$ is the skew-symmetric matrix of the cross-product.
- M_k – 6×6 spatial inertia of kth link about O_k.
- $\Phi^*_{k,k-1}$ – composite body transformation operator which transforms spatial velocities and other contravariant quantities from O_k to O_{k-1} (a 6×6 matrix, which depends on the displacement vector between these two points). The "$*$" denotes adjoint. Therefore $\Phi_{k,k-1}$ transforms spatial forces from O_{k-1} to O_k.
- H^*_k – joint matrix for kth joint (for a 1 DOF joint, H^*_k is a 6-vector, otherwise it's a matrix; if all joints have 1 DOF then $N = n$.)

- $\dot{p}_k$ – spatial velocity of kth link at O_k ($\dot{p}_k$ is a 6-vector). We note that the spatial velocity $\dot{p}_k$ should not be interpreted as the derivative of a globally defined set of coordinates p_k due to well known properties of $SE(3)$.

So, the relative spatial velocity across the kth joint is $H_k^* \dot{q}_k$, and

$$\begin{aligned} \dot{p}_k &= \Phi_{k+1,k}^* \dot{p}_{k+1} + H_k^* \dot{q}_k \\ \ddot{p}_k &= \Phi_{k+1,k}^* \ddot{p}_{k+1} + H_k^* \ddot{q}_k + a_k \end{aligned} \tag{1}$$

for $k = n, n-1, \ldots, 1$, ($\dot{p}_{n+1} = \ddot{p}_{n+1} = 0$), where a_k is Coriolis and centrifugal acceleration.

The equations of motion of the kth link about O_k are

$$\begin{aligned} f_k &= \Phi_{k,k-1} f_{k-1} + M_k \ddot{p}_k + b_k \\ \tau_k &= H_k f_k \end{aligned} \tag{2}$$

for $k = 1, 2, \ldots, n$ ($f_0 = 0$), where

- τ_k – joint force at kth joint (scalar for a 1 DOF joint)
- b_k – gyroscopic force

Defining

$$\varepsilon_\phi = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ \Phi_{2,1} & 0 & \ldots & 0 & 0 \\ 0 & \Phi_{3,2} & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & \Phi_{n,n-1} & 0 \end{pmatrix}$$

we can put the recursion relations (1), (2) in matrix form,

$$\Phi = (I - \varepsilon_\phi)^{-1} = I + \varepsilon_\phi + \ldots + \varepsilon_\phi^{n-1}$$

$$M = \operatorname{diag}\{M_1, \ldots, M_n\}$$

$$H = \operatorname{diag}\{H_1, \ldots, H_n\}$$

This results in the equations

$$\dot{p} = \Phi^* H^* \dot{q}$$

$$\ddot{p} = \Phi^* (H^* \ddot{q} + a)$$

$$f = \Phi (M \ddot{p} + b)$$

$$\tau = Hf = [H\Phi M\Phi^* H^*]\ddot{q} + H\Phi(M\Phi^* a + b)$$

(the various vectors $\ddot{q}, \ddot{p}, f, a, b, \tau$ have an obvious componentwise notation in (1), (2)), or

$$\mathcal{M}\ddot{q} = \tau - c \tag{3}$$

The CRBM solves this latter linear system of equations for $\ddot{q}$. This takes $O(N^3)$ operations using a direct Gaussian elimination method. On the other hand, $O(N)$ methods like ABM consist in propagating the recursions with $\ddot{p}$, $\ddot{q}$, rather than forming equations for $\ddot{q}$ first. This idea has been rediscovered a few times in the literature, see, e.g. (Armstrong, 1979; Bae & Haug, 1987; Brandl *et al.*, 1986; Featherstone, 1983; Rodriguez, 1987; Vereshchagin, 1974).

To understand this better, we write the equations involving $\ddot{q}, \ddot{p}$ and f as one algebraic system, following (Lubich *et al.*, 1992),

$$\begin{pmatrix} M & 0 & \Phi^{-1} \\ 0 & 0 & H \\ (\Phi^{-1})^* & H^* & 0 \end{pmatrix} \begin{pmatrix} -\ddot{p} \\ \ddot{q} \\ f \end{pmatrix} = \begin{pmatrix} b \\ \tau \\ -a \end{pmatrix} \tag{4}$$

and consider various strategies of Gaussian elimination for this system. [5]

1. Straightforward block-row elimination of the second block-row in (4) using the first and then the third block-rows gives

$$\begin{pmatrix} M & 0 & \Phi^{-1} \\ 0 & \mathcal{M} & 0 \\ (\Phi^{-1})^* & H^* & 0 \end{pmatrix} \begin{pmatrix} -\ddot{p} \\ \ddot{q} \\ f \end{pmatrix} = \begin{pmatrix} b \\ \tau - c \\ -a \end{pmatrix}$$

   where $\mathcal{M} = H\Phi M\Phi^* H^*$ and $c = H\Phi(M\Phi^* a + b)$. The second block-row is now decoupled from the rest, and yields the system (3). This essentially gives the CRB method [6].

2. Instead, we can attempt to take advantage of the *sparsity* of (4). We write the system in block form

$$\begin{pmatrix} M_k & 0 & I \\ 0 & 0 & H_k \\ I & H_k^* & 0 \end{pmatrix} \begin{pmatrix} -\ddot{p}_k \\ \ddot{q}_k \\ f_k \end{pmatrix} = \begin{pmatrix} \tilde{b}_k \\ \tau_k \\ -\tilde{a}_k \end{pmatrix} \tag{5}$$

[5]Note that (4) corresponds to an embedding of the ODE system (3) in a larger DAE system. This, however, is just a convenient representation of the same model, and we do not advocate use of DAE methods as such for simple multibody chains. In both algorithms described here, the solution process does not include finding the algebraic variables.

[6]We note that the CRBM actually involves one more feature that is essential for the efficiency of the method: joint space inertia matrix $\mathcal{M}$ is computed as $\mathcal{M} = H(R + \tilde{\Phi}R + R\tilde{\Phi}^*)H^*$ where R is a block diagonal matrix of composite rigid body inertias and $\tilde{\Phi} = \Phi - I$. However, our experimental results (Cloutier *et al.*, 1995) suggest that the instability is primarily a consequence of forming the poorly conditioned joint space inertia matrix $\mathcal{M}$.

where

$$\tilde{a}_k = a_k + \Phi^*_{k+1,k}\ddot{p}_{k+1}$$

$$\tilde{b}_k = b_k + \Phi_{k,k-1}f_{k-1}$$

and perform the elimination process as much as possible within each block (5). For this purpose, the blocks k must be decoupled (using $f_0 = 0$, $\ddot{p}_{n+1} = 0$). This elimination strategy yields the ABM.

Specifically, for the block-oriented approach (ABM) we obtain a permutation of the system (4). For instance, $n = 3$ gives the system $\boldsymbol{Mx} = \boldsymbol{b}$, where

$$\boldsymbol{M} = \begin{pmatrix} M_1 & 0 & I & & & & & & \\ 0 & 0 & H_1 & & & & & & \\ I & H_1^* & 0 & -\Phi^*_{21} & & & & & \\ & & -\Phi_{21} & M_2 & 0 & I & & & \\ & & & 0 & 0 & H_2 & & & \\ & & & I & H_2^* & 0 & -\Phi^*_{32} & & \\ & & & & & -\Phi_{32} & M_3 & 0 & I \\ & & & & & & 0 & 0 & H_3 \\ & & & & & & I & H_3^* & 0 \end{pmatrix} \tag{6}$$

$$\boldsymbol{x} = \begin{pmatrix} -\ddot{p}_1 \\ \ddot{q}_1 \\ f_1 \\ -\ddot{p}_2 \\ \ddot{q}_2 \\ f_2 \\ -\ddot{p}_3 \\ \ddot{q}_3 \\ f_3 \end{pmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} b_1 \\ \tau_1 \\ -a_1 \\ b_2 \\ \tau_2 \\ -a_2 \\ b_3 \\ \tau_3 \\ -a_3 \end{pmatrix}$$

The elimination therefore proceeds as follows: for $k = 1, 2, \ldots, n$, eliminate the middle rows in the k^{th} block precisely as in the CRBM case (4), and *also* (for $k < n$) use the k^{th} block block-rows to eliminate $-\Phi_{k+1,k}$ in order to decouple the next block. The elimination of $-\Phi_{k+1,k}$ using block k in the form corresponding to (4) produces a decoupled block $k + 1$ where only M_{k+1} needs to be updated to

$$\hat{M}_{k+1} = M_{k+1} + \Phi_{k+1,k}\hat{M}_k{\Phi_{k+1,k}}^* - \Phi_{k+1,k}\hat{M}_k H_k^*(H_k\hat{M}_k H_k^*)^{-1}H_k\hat{M}_k{\Phi_{k+1,k}}^* \tag{7}$$

($\hat{M}_1 = M_1$), and a corresponding update is applied to the right hand side:

$$\tau_k - c_k = \tau_k - H_k\hat{b}_k - H_k\hat{M}_k a_k$$

replaces τ_k and

$$\hat{b}_{k+1} = b_{k+1} + \Phi_{k+1,k}(\hat{b}_k + \hat{M}_k a_k) + \Phi_{k+1,k}\hat{M}_k H_k^*(H_k\hat{M}_k H_k^*)^{-1}(\tau_k - c_k)$$

replaces b_{k+1}. We note that $\hat{M}_k$ is the articulated body inertia at link k (denoted $P(k)$ in (Jain, 1991)); the bias force $z(k)$ in (Jain, 1991) is our $\hat{b}_k + \hat{M}_k a_k$.

For instance, in the $n = 3$ case we obtain the 0-structure

$$\begin{pmatrix}
\times & 0 & \times & & & & & & \\
0 & \times & 0 & \times & & & & & \\
\times & \times & 0 & \times & & & & & \\
 & & & \times & 0 & \times & & & \\
 & & & 0 & \times & 0 & \times & & \\
 & & & \times & \times & 0 & \times & & \\
 & & & & & & \times & 0 & \times \\
 & & & & & & 0 & \times & 0 \\
 & & & & & & \times & \times & 0
\end{pmatrix}$$

Here, $\times$ stands for a possibly nonzero block. This is precisely the update in (Featherstone, 1983; Featherstone, 1987; Jain, 1991). Following the forward elimination process we obtain a system for *both* $\ddot{q}_k$ and $\ddot{p}_k$ coupled together via a permuted block upper triangular matrix but decoupled from f. A backward elimination step then completes the ABM algorithm, yielding the relative joint accelerations $\ddot{q}$ (and also $\ddot{p}$, but the latter is not really necessary anymore). The pivotal blocks used in this backward elimination step are, in this order, $(3k-1, 3k-1)$, $(3k, 3k-2)$ for $k = n, n-1, \ldots, 1$. For the $n = 3$ instance, they are circled below. Recall that the pivotal blocks are of full rank: the $(3k, 3k-2)$ blocks are identities and the $(3k-1, 3k-1)$ blocks are the joint inertias $D_k = H_k\hat{M}_k H_k^*$ which are invertible.

$$\begin{pmatrix}
\times & 0 & \times & & & & & & \\
0 & \otimes & 0 & \times & & & & & \\
\otimes & \times & 0 & \times & & & & & \\
 & & & \times & 0 & \times & & & \\
 & & & 0 & \otimes & 0 & \times & & \\
 & & & \otimes & \times & 0 & \times & & \\
 & & & & & & \times & 0 & \times \\
 & & & & & & 0 & \otimes & 0 \\
 & & & & & & \otimes & \times & 0
\end{pmatrix}$$

The advantage of the block algorithm ABM is that all operations are done blockwise and therefore the operation count is clearly $O(N)$. This algorithm is therefore superior to CRBM when N is large enough. The disadvantage here is that more work needs to be done when N is small, e.g. $N < 7$, because not only the relative coordinates but also the absolute ones are calculated. This gives the edge for small systems to the CRBM, so long as the two algorithms produce essentially the same numerical results, which is the case in well-conditioned multibody systems. But Example 1 shows that a different situation arises for ill-conditioned systems.

It is simple to extend this algorithm description to tree structured systems and even to systems with closed loops (Lubich *et al.*, 1992), even though the latter lead to differential-algebraic equations which cannot be simply reduced to ODEs as in (3) and the $O(N)$ complexity is achieved only for a fixed number of closed loops. Here, however, we proceed to explain the stability of these algorithms, which may already be observed for simple multibody chains.

3 Explaining formulation stiffness

Since we have shown that the CRBM and ABM solve precisely the same linear system of algebraic equations at each time instance, one may suspect that the difference in roundoff error effects is due either to insufficient pivoting in the Gaussian elimination process of the CRBM or to an excessive mixing of contributions from different blocks which results in an unfortunate cancellation error when using floating point arithmetic. We now consider a two-link chain with the link connected to the base being much lighter (and shorter) than the other link, as in Example 1, and show that poor cancellation is the source of the observed roundoff error effects.

Consider the ABM first. Using the first and third block-rows to eliminate the second results in the equations

$$(H_1 M_1 H_1^*)\ddot{q}_1 + (H_1 M_1 \Phi_{21}^*)\ddot{p}_2 = \tau_1 - (H_1 b_1 + H_1 M_1 a_1) =: \tau_1 - c_1$$

Using the same block-rows plus the above equation to decouple the second block from the first results in the equations for the second block

$$\begin{pmatrix} \hat{M}_2 & 0 & I \\ 0 & 0 & H_2 \\ I & H_2^* & 0 \end{pmatrix} \begin{pmatrix} -\ddot{p}_2 \\ \ddot{q}_2 \\ f_2 \end{pmatrix} = \begin{pmatrix} \hat{b}_2 \\ \tau_2 \\ -a_2 \end{pmatrix} \tag{8}$$

where $\hat{M}_2 = M_2 + \Phi_{21} M_1 \Phi_{21}^* - \Phi_{21} M_1 H_1^* (H_1 M_1 H_1^*)^{-1} H_1 M_1 \Phi_{21}^*$, and $\hat{b}_2 = b_2 + \Phi_{21}^* b_1 + \Phi_{21}^* M_1 a_1 + \Phi_{21}^* M_1 H_1^* (H_1 M_1 H_1^*)^{-1} (\tau_1 - c_1)$. Next using the first and third block-rows in (8) to eliminate the second results in the equations

$$(H_2 \hat{M}_2 H_2^*)\ddot{q}_2 = \tau_2 - (H_2 \hat{b}_2 + H_2 \hat{M}_2 a_2) =: \tau_2 - c_2$$

This completes the forward pass. Now we solve backwards for $\ddot{q}_2$, then $\ddot{p}_2 = a_2 + H_2^* \ddot{q}_2$, and finally we solve for $\ddot{q}_1$.

Next consider the CRBM. We have

$$\mathcal{M} = \begin{pmatrix} H_1 M_1 H_1^* & H_1 M_1 \Phi_{21}^* H_2^* \\ H_2 \Phi_{21} M_1 H_1^* & H_2 (M_2 + \Phi_{21} M_1 \Phi_{21}^*) H_2^* \end{pmatrix}. \tag{9}$$

To solve using the 2×2 matrix $\mathcal{M}$, we eliminate the lower left element by a block-row operation. This gives in the lower right (diagonal) position precisely $H_2 \hat{M}_2 H_2^*$.

Therefore, *there is no poor pivoting strategy involved here*, even when $m_1 \gg m_2$. Rather, the only difference is in the order in which $\hat{M}_2$ is being formed! In the CRBM we must, while forming $\mathcal{M}$, add the small M_2 to the large $\Phi_{21} M_1 \Phi_{21}^*$. Then the other large, rank deficient term $\Phi_{21} M_1 H_1^* (H_1 M_1 H_1^*)^{-1} H_1 M_1 \Phi_{21}^*$ is subtracted during the solution process. This may cause cancellation error which appears as a random function of time t, even though all the quantities in (3) are smooth functions of t. When such noise becomes relatively large a nonsmooth solution profile results which would slow down an unsuspecting adaptive ODE dynamics solver.

The ABM can be easily formulated to perform as badly as the CRBM! But with the ABM we can also do better, by first forming the projected inertia matrix $\bar{M}_1 = M_1 - M_1 H_1^* (H_1 M_1 H_1^*)^{-1} H_1 M_1$. Then forming $\Phi_{21} \bar{M}_1 \Phi_{21}^*$ gives a rank deficient matrix with possibly large terms, and only then M_2 is added to form $\hat{M}_2$.

Example 2 *In order to illustrate this last point, consider the case where* $H_1 = (0,0,0,0,0,1)$, $\Phi_{21} = I$ *and* $H_2 = H_1$. *This choice corresponds to collinear prismatic joints; the choice of* Φ_{21} *is a good approximation to the actual transformation when the base link is short; and the choice of parallel* H_2 *gives a worst case scenario. Further, to simplify notation assume that the difference between the masses* m_1 *and* m_2 *is so large that in floating point arithmetic* $fl(m_1 + m_2) = fl(m_1)$. *Then, when using CRBM one forms* $M_2 + \Phi_{21} M_1 \Phi_{21}^*$, *the contribution of* m_2 *is entirely lost, and the system (3) becomes singular (see (9), where all entries become equal, and Table 2). In particular, one obtains* $H_2 \hat{M}_2 H_2^* = 0$. *On the other hand, it is easy to see that* $\bar{M}_1 = M_1(I - H_1^* H_1)$, *so the* $(6,6)-$*entry of* $\bar{M}_1$ *is zero while the rest are like in* M_1. *Forming* $M_2 + \Phi_{21} \bar{M}_1 \Phi_{21}^*$ *now gives* m_2 *in the lower right corner, so* $H_2 \hat{M}_2 H_2^* = m_2$ *correctly.*

We remark that such bad cases can occur in practice; for instance, if there are three intersecting revolute joints at the shoulder, two of these joints can become collinear at a shoulder singularity.

□

The importance of M_2 is therefore in its contribution to the nullspace of $\tilde{M}_2 = \Phi_{21} \bar{M}_1 \Phi_{21}^*$, particularly what effect remains after the quadratic form with H_2 is taken. More generally, we cannot assume that $\Phi_{21} = I$ or that $H_2 = H_1$. Let $\mathcal{S}_2 = range\{(\Phi_{21}^{-1})^* H_1^*\}$ and project H_2^* into this subspace,

$$H_2^* = E_2^* + F_2^*$$

where $F_2^* \in \mathcal{S}_2$ and $E_2^* \perp \mathcal{S}_2$. Since

$$\bar{M}_1 H_1^* = 0,$$

we have that

$$H_2\hat{M}_2H_2^* = H_2M_2H_2^* + E_2\tilde{M}_2E_2^* \tag{10}$$

so the term involving M_2 does not get swallowed (in finite precision arithmetic) if the other term on the right hand side of (10) is not much larger, and this in turn occurs when E_2 is sufficiently small in norm, which is the only case when the contribution of $H_2M_2H_2^*$ matters. [7]

In (10) we consider the effect of adding $H_2M_2H_2^*$ to $H_2\tilde{M}_2H_2^*$. However, the order of operations using the ABM with the standard implementation is to form $M_2 + \tilde{M}_2$ first, and only then calculate the quadratic form with H_2. This procedure can be ruined by a poor choice of coordinates: $\tilde{M}_2 = \Phi_{21}\bar{M}_1\Phi_{21}^*$ is rank deficient, but could have all large entries in general. (For an example, repeat Example 2 with everything the same except $H_2 = H_1 = \frac{1}{\sqrt{2}}(0,0,0,0,1,1)$.) In this case the ABM could perform as badly as the CRBM. We observe this phenomenon with the SD/FAST package as well. For instance, in Example 1, if the coordinates used for defining the moments of inertia are rotated by 45 degrees, the integrator using the ABM option breaks down at roughly the same condition number as the CRBM. While such difficulties can sometimes be addressed by a careful choice of coordinates, this is not always convenient or possible due to the constraints placed by modeling software. In §4 below, we describe a modified ABM that addresses this problem.

It can be verified in general (i.e. also for $n > 2$) that when using the CRBM the block pivot elements obtained by the Gaussian elimination process[8] are $H_k\hat{M}_kH_k^*$, i.e. they are the same as in the ABM described in (Featherstone, 1983; Jain, 1991). Thus, observations and explanations made here regarding ill-conditioned robotics systems extend to the more general case. In fact, adding more links at the tip end of the ill-conditioned two-body system makes the conditioning worse. This has also been verified by experimentation with the SD/FAST package.

4 Modified ABM

The above deliberations highlight the particular importance of calculating $D_k = H_k\hat{M}_kH_k^*$ as accurately as possible. In fact, using ABM the calculation of $\hat{M}_k$ may be significantly more polluted by floating point cancellation error effects than that of D_k if local coordinate systems are not aligned with the principal directions of motion. This then suggests to reorder the calculations in the ABM so as not to use $\hat{M}_k$ for calculating D_k: Starting with $D_1 = H_1M_1H_1^*$,

$$\bar{M}_k \quad = \quad \hat{M}_k - \hat{M}_kH_k^*D_k^{-1}H_k\hat{M}_k$$

[7]We note that in finite precision arithmetic, the relationship (10) also holds only approximately; but this does not alter the essence of the argument.

[8]using no row permutations and our system of link enumeration.

		CRBM		*ABM*		*Modified ABM*	
m_1	*case*	D_2	$\ddot{q}_1$	D_2	$\ddot{q}_1$	D_2	$\ddot{q}_1$
1.e+7	*1*	*1.0*	*971.19*	*1.0*	*971.19*	*1.0*	*971.19*
1.e+8	*1*	*0*	–	*1.0*	*971.19*	*1.0*	*971.19*
1.e+9	*1*	*0*	–	*1.0*	*971.19*	*1.0*	*971.19*
1.e+6	*2*	*1.03*	*941.76*	*1.03*	*941.76*	*1.0*	*971.19*
1.e+7	*2*	*1.5*	*647.46*	*1.5*	*647.46*	*1.0*	*971.19*
1.e+8	*2*	*0*	–	*0*	–	*1.0*	*971.19*
1.e+9	*2*	*0*	–	*0*	–	*1.0*	*971.19*

Table 2: Comparison of modified ABM with CRBM and ABM.

$$\begin{aligned}
\tilde{M}_{k+1} &= \Phi_{k+1,k}\bar{M}_k\Phi^*_{k+1,k} \\
D_{k+1} &= H_{k+1}M_{k+1}H^*_{k+1} + H_{k+1}\tilde{M}_{k+1}H^*_{k+1} \\
\hat{M}_{k+1} &= M_{k+1} + \tilde{M}_{k+1}
\end{aligned} \tag{11}$$

In words, we compute and maintain D_k separately from $\hat{M}_k$, $k = 1, 2, \ldots, n$. This has the small extra expense of computing the quadratic form with H_{k+1} twice at each step instead of once, but it has the added stability in case of a general local coordinate system. As before, note that in fact,

$$H_{k+1}\tilde{M}_{k+1}H^*_{k+1} = E_{k+1}\tilde{M}_{k+1}E^*_{k+1}$$

where E^*_{k+1} is the distance matrix (vector for a 1 DOF joint) from H^*_{k+1} to the subspace $\mathcal{S}_{k+1} = range\{(\Phi^{-1}_{k+1,k})^* H^*_k\}$ which is the nullspace of $\tilde{M}_{k+1}$. The contribution of $H_{k+1}M_{k+1}H^*_{k+1}$ to D_{k+1} is important, even when $\|M_{k+1}\| \ll \|\tilde{M}_{k+1}\|$, precisely when $\|E_{k+1}\|$ is small.

Example 3 *In Table 2 we list computational results obtained in single precision for the extended Example 2. We take* $m_2 = 1$ *and list* m_1. *The lengths of the links are proportional to their masses. We choose* $a_k = b_k = 0, \tau_1 = 981, \tau_2 = 9.81$, *and list results for the two cases: (1)* $H = (0,0,0,0,0,1)^T$ *and (2)* $H = \frac{1}{\sqrt{2}}(0,0,0,0,1,1)$. *It is easy to see that in exact arithmetic* $D_2 = 1$ *regardless of* m_1. *We list it and the acceleration of the tip joint. Note that the modified ABM given in (11) gives better results than the CRBM. In case (2), where the local coordinate systems are not aligned with the principal directions of motion, the results are also better than the usual ABM. The usual ABM does not improve over the CRBM in the latter case.*

$\square$

5 Conclusions

While there has been considerable work in reducing the computational complexity of algorithms for computing the forward dynamics of robot manipulators, it is only recently that the numerical properties of these algorithms have been considered carefully. This is important since forward dynamics algorithms are typically used with adaptive step-size numerical integrators which could interpret the poor numerical performance of the forward dynamics as requiring much smaller step sizes. We have shown how some of these numerical properties, in particular formulation stiffness, can be analyzed.

We presented a unified formulation of two important methods for computing forward dynamics, viz., the composite rigid body method and the articulated body method, as elimination methods for solving a large, sparse, linear system. The articulated body method is shown to obtain its linear time complexity by exploiting the sparsity of the system. We analyzed the formulation stiffness of these algorithms and showed that the poor behavior of the composite rigid body method in certain ill-conditioned systems is due to certain cancellation errors, and this problem can be avoided in the articulated body method. Our analysis also reveals pitfalls for implementors of the articulated body method and other forward dynamics algorithms.

A modified articulated body method with superior stability properties for general, ill-conditioned multibody problems was proposed and demonstrated.

Acknowledgement: We wish to thank Prof. J. Varah for fruitful discussions.

References

Angeles, J., & Ma, O. 1988. Dynamic Simulation of n-Axis Serial Robotic Manipulators Using a Natural Orthogonal Complement. *International Journal of Robotics Research*, **7**(5), 32–47.

Armstrong, W.W. 1979. Recursive Solution to the Equations of Motion of an n-link Manipulator. *Pages 1343–1346 of: Proceedings of the 5th World Congress on Theory of Machines and Mechanisms*, vol. 2.

Ascher, U., & Lin, P. 1995. *Sequential Regularization Methods for Nonlinear Higher Index DAEs.* Tech. report 95-14, Comp. Sci, UBC, Vancouver.

Bae, D.S., & Haug, E.J. 1987. A Recursive Formulation for Constrained Mechanical System Dynamics: Part I. Open Loop Systems. *Mechanical Structures & Machines*, **15**(3), 359–382.

Bock, G., & von Schwerin, R. 1993. *An inverse dynamics Adams-method for constrained multibody systems.* Tech. report 93-27, IWR, Univ. Heidelberg.

Brandl, H., Johanni, R., & Otter, M. 1986 (December). A Very Efficient Algorithm for the Simulation of Robots and Similar Multibody Systems Without Inversion of the Mass Matrix. *In: Proceedings of IFAC/IFIP/IMACS International Symposium on the Theory of Robots.*

Cloutier, B., Pai, D. K., & Ascher, U. M. 1995. The formulation stiffness of forward dynamics algorithms and implications for robot simulation. *In: Proceedings of the IEEE Conference on Robotics and Automation.*

Ellis, R.E., & Ricker, S.L. 1994. Two Numerical Issues in Stimulating Constrained Robot Dynamics. *IEEE Transactions on Systems, Man, and Cybernetics*, **24**(1), 19–27.

Ellis, R.E., Ismaeil, O.M., & Carmichael, I.H. 1992. Numerical Stability of Forward-Dynamics Algorithms. *In: Proceedings of the IEEE Conference on Robotics and Automation.*

Featherstone, R. 1983. The Calculation of Robot Dynamics Using Articulated-Body Inertias. *International Journal of Robotics Research*, **2**(1), 13–30.

Featherstone, R. 1987. *Robot Dynamics Algorithms.* Norwell, MA: Kluwer Academic Publishers.

Jain, A. 1991. Unified Formulation of Dynamics for Serial Rigid Multibody Systems. *Journal of Guidance, Control, and Dynamics*, **14**(3), 531–542.

Lubich, C., Nowak, U., Pohle, U., & Engstler, Ch. 1992. *MEXX – Numerical software for the integration of constrained mechanical multibody systems.* Tech. report SC92-12, Konrad-Zuse-Zentrum für Informationstechnik, Berlin.

Rodriguez, G. 1987. Kalman Filtering, Smoothing and Recursive Robot Arm Forward and Inverse Dynamics. *IEEE Journal of Robotics and Automation*, **3**(6), 624–639.

Rodriguez, G., Jain, A., & Kreutz-Delgado, K. 1991. A Spatial Operator Algebra for Manipulator Modeling and Control. *The International Journal of Robotics Research*, **10**(4), 371–381.

Vereshchagin, A.F. 1974. Computer Simulation of the Dynamics of Complicated Mechanisms of Robot-Manipulators. *Engineering Cybernetics*, **6**, 65–70.

Walker, M.W., & Orin, D.E. 1982. Efficient Dynamic Computer Simulation of Robotic Mechanisms. *Journal of Dynamic Systems, Measurement, and Control*, **104**, 205–211.