Compensating Mass Matrix Potential for Constrained Molecular Dynamics

Abhinandan Jain

Jet Propulsion Laboratory/California Institute of Technology, 4800 Oak Grove Drive, Pasadena, California 91109

Received January 1997

1. INTRODUCTION

Rigid internal constraints are used in molecular models to speed up molecular dynamics (MD) simulations. It is well recognized that statistical averages from such constrained MD simulations differ by a metric tensor-dependent term from similar averages computed using conventional unconstrained MD simulations. Fixman proposed augmenting the standard potential with a compensating term which depends on the metric tensor to nullify the effects of this bias term. However, in the absence of tractable algorithms to compute this compensating tensor potential and its gradient its use has been impractical. This paper derives a new algorithm for computing this compensating tensor potential and its gradient for tree topology molecular systems. The algorithm is quite straightforward and is an extension of the spatial operators based \( O(2) \) algorithm that has been recently proposed for constrained dynamics. Indeed, the compensating potential is closely related and computed from the articulated body inertia quantities available from this \( O(2) \) algorithm. © 1997 Academic Press

2. ENSEMBLE AVERAGES IN CONSTRAINED DYNAMICS SIMULATIONS

The partition function \( \mathcal{Z}(\mathcal{T}) \) for an \( n \) degree of freedom unconstrained Cartesian molecular model is given by the expression

\[
\mathcal{Z}(\mathcal{T}) = \frac{1}{h^n} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\gamma_1} \cdots \int_{-\infty}^{\gamma_n} \exp \left[ -\frac{1}{2} p \cdot \mathcal{M}^{-1} p + \gamma \right] / k \mathcal{T} \, dp_1 \cdots dp_n dq_1 \cdots dq_n
\]

where \( \mathcal{T} \) denotes the temperature, and \( q_i \) and \( p_i \) are the configuration and momentum coordinates, \( \gamma \) is the standard potential energy, \( \mathcal{M} \in \mathbb{R}^{n \times n} \) denotes the system mass matrix (or metric tensor). The system kinetic energy is given by \( \frac{1}{2} p \cdot \mathcal{M}^{-1} p \), and the limits of integration \( \alpha_i \) and \( \gamma_i \) are determined by the geometry of the problem [12]. For conventional unconstrained Cartesian dynamics, the mass matrix \( \mathcal{M} \) is constant (and diagonal) and does not depend upon the system configuration. As a consequence, the ensemble average of a function \( f(q) \) is given by

\[
\langle f(q) \rangle = \frac{2\pi k \mathcal{T}^{n/2}}{h^n} \int_{-\alpha_1}^{\gamma_1} \cdots \int_{-\alpha_n}^{\gamma_n} f(q) \exp[-\gamma/k \mathcal{T}] \, dq_1 \cdots dq_n \, \mathcal{T}(\mathcal{T}^{-1})
\]

has been a major hurdle on the use of these compensating potentials more generally in constrained MD simulations.

This paper analyzes the structure of the compensating potential and its gradient and develops substantially simpler expressions for them for tree topology molecular models. These expressions are used to derive computational algorithms for use in constrained MD simulations. Indeed, the compensating potential and its gradient are readily computable from the articulated body inertia quantities available from this \( O(2) \) algorithm.
where the momentum variables have been integrated over and eliminated.

In constrained dynamics models, the molecular system is modeled as a collection of rigid clusters coupled together by articulable hinges [2]. For such constrained dynamics models, the mass matrix $\mathcal{M}(\theta) \in \mathbb{R}^{n \times n}$ is a function of the internal configuration coordinates vector $\theta$ and $n$ the number of degrees of freedom for the constrained model. The kinetic energy $K.E.$ is now given by the expression

$$K.E. = \frac{1}{2} \beta^* \mathcal{M}(\theta) \beta = \frac{1}{2} p^* \mathcal{M}^{-1}(\theta) p,$$  

(2.3)

where $\beta$ denotes the internal velocity coordinates vector, and where the conjugate momenta vector $p$ is given by the expression

$$p = \mathcal{M}(\theta) \beta.$$  

(2.4)

The partition function $\mathcal{Z}(\mathcal{F})$ for the system is given by the expression

$$\mathcal{Z}(\mathcal{F}) = \frac{1}{\mathcal{F}^{\frac{n}{2}}} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \exp \left\{ -\frac{1}{2} p^* \mathcal{M}^{-1}(\theta) p + \mathcal{F} \right\} / k\mathcal{F} \times dp_1 \cdots dp_n d\theta_1 \cdots d\theta_n.$$  

(2.5)

Using a diagonalizing transformation on the momentum coordinates of the form

$$p' = \mathcal{M}^{-1/2}(\theta)p$$  

(2.6)

we can integrate Eq. (2.5) over the new momentum coordinates to get the following expression for the partition function

$$\mathcal{Z}(\mathcal{F}) = \frac{2\pi k^{\frac{n}{2}}}{\mathcal{F}^{\frac{n}{2}}} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \det \{ \mathcal{M}^{1/2}(\theta) \} \exp \left\{ -\mathcal{F} / k\mathcal{F} \right\} \times d\theta_1 \cdots d\theta_n.$$  

(2.7)

This implies that the ensemble average of a function $f(\theta)$ over the configuration space defined by the internal coordinates is given by

$$\langle f(\theta) \rangle = \frac{2\pi k^{\frac{n}{2}}}{\mathcal{F}^{\frac{n}{2}}} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \det \{ \mathcal{M}^{1/2}(\theta) \} f(\theta) \exp \left\{ -\mathcal{F} / k\mathcal{F} \right\} \times d\theta_1 \cdots d\theta_n | \mathcal{Z}'.$$  

(2.8)

In contrast with the unconstrained Cartesian dynamics expression in Eq. (2.2), the integrand in the partition function integral in Eq. (2.5) involves the determinant of the mass matrix. This additional term introduces a bias in statistical averages computed using constrained dynamics models versus those obtained using Cartesian models [3].

A method for bridging this gap in statistical estimates was proposed by Fixman [3]. He suggested replacing the potential function $\gamma(\theta)$ by the modified potential function $\gamma'(\theta)$

$$\gamma'(\theta) \equiv \gamma(\theta) + \gamma_c(\theta), \quad \text{where} \quad \gamma_c(\theta) \equiv \frac{1}{2} \ln \det \{ \mathcal{M}(\theta) \}$$  

(2.9)

in constrained MD simulations. It is easy to see that replacing $\gamma'(\theta)$ in Eq. (2.7) and Eq. (2.8) by $\gamma'(\theta)$ eliminates the metric tensor from the partition functions and the expression for the ensemble average. The extra potential term $\gamma_c(\theta)$ is referred to as the compensating mass matrix potential or the metric-tensor potential [7], since it effectively compensates for the mass matrix determinant term in Eq. (2.8).

With the use of $\gamma'(\theta)$, the remaining differences between statistical averages computed using constrained and unconstrained MD simulations are due to the coarser sampling of the conformational space during the averaging process in constrained MD simulations. The appropriate use of constraints is important for ensuring that the loss in fidelity is within acceptable limits for the simulation experiment at hand. The use of the compensating mass matrix potential helps ensure that at least the systematic bias term from the metric tensor does not contribute to the averaging errors.

The implication of using $\gamma'(\theta)$ is that now its gradient must be used for computing the forces during constrained MD simulations. The overall hinge torque vector $T'$ is defined as the gradient of $\gamma'(\theta)$ and is given by

$$T' = \nabla_\theta \gamma'(\theta) = T + T_c,$$  

(2.10)

where

$$T \equiv \nabla_\theta \gamma(\theta), \quad T_c \equiv \nabla_\theta \gamma_c(\theta).$$

$T_c$ represents the compensating hinge torque arising from the compensating mass-matrix potential $\gamma_c(\theta)$ and must be used in addition to the standard torque term $T$ during constrained MD simulations. Its $k$th element, $T_c(k)$, for the $k$th hinge is given by

$$T_c(k) = \frac{\partial \gamma_c(\theta)}{\partial \theta(k)} = \frac{1}{2} \frac{\partial \ln \det \{ \mathcal{M}(\theta) \} }{\partial \theta(k)}.$$  

(2.11)
The need for the use of the compensating potential has been verified via computer simulations for several small systems [7, 8, 10, 11, 13, 14]. There is general consensus that the compensating potential is less important for rigid constraints on the bond stretching degrees of freedom while they are a significant factor for rigid constraints involving bond angles. It was found [7, 8] that the use of the compensating potential in constrained MD simulations for \( n \)-butane other systems effectively bridged the gap in the number of dihedral transitions using constrained and unconstrained MD simulations. Go and Scheraga [5, 6] examined the relative merits of Cartesian dynamics models and constrained dynamics models and concluded that the former was the more correct model for molecular dynamics simulations. However, Refs. [8, 9] argued that the constrained model was the more correct model for molecular dynamics simulations. Go and Scheraga [5, 6] examined the relative merits of Cartesian dynamics models and constrained dynamics models and concluded that the former was the more correct model for molecular dynamics simulations. However, Refs. [8, 9] argued that the constrained MD simulations produce similar statistical estimates as the Cartesian case if the compensating potential is included.

Despite the accepted importance of using the compensating potential during constrained dynamics simulations, it is rarely used in practice. The primary factor has been the lack of a tractable method for computing the compensating torque, \( T_c \), for all but simple molecular systems [13]. A clever method for computing \( T_c \) was proposed by Fixman [4]. However, this method is also limited to moderatively sized molecular systems and does not include a procedure for computing the compensating torque \( T_c \).

In the following sections we take a closer look at the expression for \( T_c \) in Eq. (2.11) and derive simpler expressions for it using techniques from the spatial operator algebra [2]. These expressions lead to a simple method for computing this compensating torque for arbitrary tree-topology molecular systems. This method is an extension of the spatial operators based algorithm for constrained dynamics simulations [2]. This algorithm provides a highly efficient \( O(V) \), recursive method for solving the equations of motion for constrained systems without using iterative procedures such as in the SHAKE algorithm [15]. The equations of motion for the system are solved exactly and the complexity of the algorithm is \( O(V) \), i.e., the computational cost grows only linearly with the number of unconstrained degrees of freedom in the system. The algorithm—which has been implemented as the NEIMO (Newton–Euler inverse mass operator method) software package [2, 16, 17]—is based upon closed-form spatial operator expressions for the factorization and inversion of the mass matrix.

**Expression for the Compensating Torque \( T_c(k) \)**

In general, if \( g(X) \) is a scalar function of a matrix \( X \in \mathbb{R}^{m \times n} \), then its derivative with respect to a variable \( y \) is given by

\[
\frac{\partial g(X)}{\partial y} = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{\partial g(X)}{\partial X(i,j)} \frac{\partial X(i,j)}{\partial y} = \text{Trace} \left( \frac{\partial g}{\partial X} \frac{\partial X}{\partial y} \right),
\]

(2.12)

where \( \frac{\partial g}{\partial X}(i,j) \) and \( \frac{\partial X(i,j)}{\partial y} \) are \( m \times n \) matrices whose elements are defined as

\[
\frac{\partial g}{\partial X}(i,j) = \frac{\partial g(X)}{\partial X(i,j)}, \quad \frac{\partial X(i,j)}{\partial y} = \frac{\partial X(i,j)}{\partial y}.
\]

For the scalar function \( g(X) = \ln \det \{X\} \), it is a well established fact [18] that

\[
\frac{\partial g(X)}{\partial X} = \frac{\partial \ln \det \{X\}}{\partial X} = \{X^*\}^{-1}.
\]

(2.13)

Using Eq. (2.13) and Eq. (2.12) in Eq. (2.11) leads to the following expression for \( T_c(k) \):

\[
T_c(k) = \frac{1}{2} \text{Trace} \left \{ \mathcal{M}^{-1}(\theta) \frac{\partial \mathcal{M}(\theta)}{\partial \theta(k)} \right \} = \frac{1}{2} \text{Trace} \left \{ \mathcal{M}^{-1}(\theta) \mathcal{M}_{(k)}(\theta) \right \}.
\]

(2.14)

We have used the notational shorthand \( \mathcal{M}_{(k)}(\theta) \) in place of \( \mathcal{M}(\theta)/\partial \theta(k) \) in Eq. (2.14). In the following sections we use spatial operator expressions for \( \mathcal{M}(\theta) \) to further simplify Eq. (2.14).

3. **Spatial Operator Form of Mass Matrix**

Now we briefly review the derivation of the spatial operator equations of motion and refer the reader to Ref. [2] for more detailed discussion on the notation and the concepts. For notational simplicity we limit our initial discussion to an \( n \)-cluster system with \textit{serial chain} structure and single degree of freedom rotational hinges between clusters. The number of degrees of freedom for this system is \( n+5 \). Later we discuss the steps involved in extending the derivations and algorithms to general tree-topology systems with multiple degrees of freedom hinges.

As shown in Ref. [2], the Newton–Euler recursive equations of motion for the whole system have the form:
In particular, the equations of motion have the form:

\[
V(n+1) = 0, \alpha(n+1) = 0
\]

for \( k = n \cdots 1 \)

\[
V(k) = \phi^*(k+1,k)V(k+1) + H^*(k)\theta(k)
\]

\[
\alpha(k) = \phi^*(k+1,k)\alpha(k+1) + H^*(k)\dot{\theta}(k) + a(k)
\]

end loop

\[
f(0) = 0
\]

for \( k = 1 \cdots n \)

\[
f(k) = \phi(k,k-1)f(k-1) + M(k)\alpha(k) + b(k) + \ddot{f}_c(k)
\]

\[
T(k) = H(k)f(k)
\]

end loop

The introduction of spatial operators allows the expression of the equations of motion in the following more concise form:

\[
V = \phi^*H\dot{\theta}
\]

\[
\alpha = \phi^*(H\dot{\theta} + a)
\]

\[
f = \phi(M\alpha + b + \dot{f}_c)
\]

\[
= \phi M\phi^*H\dot{\theta} + \phi(M\phi^*a + b + \dot{f}_c)
\]

\[
T = Hf = H\phi M\phi^*H\dot{\theta} + H\phi(M\phi^*a + b + \dot{f}_c).
\]

In particular, the equations of motion have the form

\[
T = \mathcal{M}(\theta)\dot{\theta} + \mathcal{E}(\theta, \dot{\theta}), \quad (3.3)
\]

where

\[
\mathcal{M}(\theta) \triangleq H\phi M\phi^*H \in \mathbb{R}^{r \times r}
\]

\[
\mathcal{E}(\theta, \dot{\theta}) \triangleq H\phi(M\phi^*a + b + \dot{f}_c) \in \mathbb{R}^r. \quad (3.4a)
\]

The new square factorization described in Eq. (3.5a) is also referred to as the Innovations Operator Factorization of the mass matrix and is an alternative to the factorization in Eq. (3.4). The factor \([I + H\phi K] \in \mathbb{R}^{r \times r}\) is square, block lower triangular, and nonsingular, while \(D\) is a block diagonal matrix. This factorization provides a closed form operator expression for the block \(LDL^*\) decomposition of \(\mathcal{M}\). The following lemma gives the closed form operator expression for the inverse of the factor \([I + H\phi K]\). Once again, the factor \([I + H\phi K]\) is square, block lower triangular, and nonsingular and so Eq. (3.5c) provides a closed form expression for the block \(LDL^*\) decomposition of \(\mathcal{M}^{-1}\). The spatial operators \(\phi, K,\) and \(D\) embedded in these factorizations are based on spatially recursive filtering and smoothing algorithms [19, 21, 22]. The following Riccati equation for the articulated body inertia \(P\) is a key element of these filtering and smoothing algorithms.

\[\text{Algorithm 3.1. The articulated body inertia quantities}\ P(\cdot), D(\cdot), G(\cdot), K(\cdot), \tau(\cdot), \tau^*(\cdot), P^*(\cdot), \text{and } \psi(\cdot, \cdot) \text{ are computed by the following recursive procedure:}\
\]
Using Eq. (2.14) as a starting point, we now develop an algorithm for solving the equations of motion. Let 

\[ P^+(0) = 0 \]

for \( k = 1 \cdots N \)

\[ P(k) = \phi(k, k-1)P^+(k-1) + M(k) \]

\[ D(k) = H(k)P(k)H^*(k) \]

\[ G(k) = P(k)H^*(k)D^{-1}(k) \]

\[ K(k+1, k) = \phi(k+1, k)G(k) \quad (3.6) \]

\[ \tau(k) = G(k)H(k) \]

\[ \bar{\tau}(k) = I - \tau(k) \]

\[ P^+(k) = \bar{\tau}(k)P(k) \]

\[ \psi(k+1, k) = \phi(k+1, k)\bar{\tau}(k) \]

end loop

Algorithm 3.1 is by now the classical [19, 23] Riccati equation of Kalman filtering. Its solution \( P(k) \) is the articulated body inertia [19, 24] of the part of the system outboard of hinge \( k \). The operator \( P \) is a block-diagonal \( 6n \times 6n \) matrix with its \( k \)th diagonal element being \( P(k) \in \mathbb{R}^{6 \times 6} \). Define also

\[ D = HPH^* \in \mathbb{R}^{6 \times 6} \]

\[ G = PH^*D^{-1} \in \mathbb{R}^{6n \times 6} \]

\[ K = \psi^*G \in \mathbb{R}^{6n \times 6} \]

\[ \bar{\tau} = I - GH \in \mathbb{R}^{6n \times 6} \]

\[ \psi = (I - \bar{\psi})^{-1} \in \mathbb{R}^{6n \times 6} \]

The operators \( D, G, \) and \( \bar{\tau} \) are all block diagonal. The operators \( K \) and \( \psi \) are not block-diagonal, but their only nonzero block elements are \( K(k, k-1)'s \) and \( \psi(k, k-1)'s \), respectively, along the first subdiagonal. The following lemma describes the operator expression for the generalized accelerations \( \dot{\theta} \) in terms of the hinge forces \( T \) and Cartesian spatial forces \( \tilde{f}_c \).

**Lemma 3.2.**

\[ \dot{\theta} = [I - H\psi K^*D^{-1}][T - H\psi(KT + Pa + b + \tilde{f}_c)] - K^*\psi^*a \quad (3.8) \]

**Proof.** See Ref. [2].

\[ O(\cdot, \cdot) \text{ Algorithm for Solving the Equations of Motion} \]

The recursive implementation of Eq. (3.8) leads to the following \( O(\cdot, \cdot) \) computational algorithm for the accelerations, \( \dot{\theta} \):

\[ \dot{z}^+(0) = 0 \]

for \( k = 1 \cdots n \)

\[ z(k) = \phi(k, k-1)z^+(k-1) + P(k)a(k) + b(k) + \tilde{f}_c(k) \quad (3.9) \]

\[ \epsilon(k) = T(k) - H(k)z(k) \]

\[ \nu(k) = D^{-1}(k)\epsilon(k) \]

\[ z^+(k) = z(k) + G(k)\epsilon(k) \]

end loop

\[ \alpha(n + 1) = 0 \]

for \( k = n \cdots 1 \)

\[ \alpha^c(k) = \phi^*(k+1, k)\alpha(k+1) \quad (3.10) \]

\[ \dot{\theta}(k) = \nu(k) - G^*(k)\alpha^c(k) \]

\[ \alpha(k) = \alpha^c(k) + H^*(k)\dot{\theta}(k) + a(k) \]

end loop

This algorithm does not require either the explicit computation of the mass matrix or, nor the numerical solution of the matrix equation (3.3). The steps in the above algorithm can be summarized as follows:

1. The first step is a recursion from the base to the tip to compute the orientation, location, and spatial velocities, \( V(k) \), and the Coriolis and gyroscopic terms \( a(k) \) and \( b(k) \) for each of the clusters using the first base-to-tip recursion in Eq. (3.1).

2. Next follows a recursion from the tip towards the base as defined by Eq. (3.6) to compute the \( P(k)'s \) etc.

3. The recursion in Eq. (3.9) from the tip to the base is used next to compute the residual forces \( z(k) \) etc. This recursion can be combined with the tip to base recursion in the previous step to obtain a single tip to base recursion sequence.

4. Finally, the base to tip recursion described by Eq. (3.10) computes the \( \dot{\theta}(k) \) accelerations for all the clusters.

The computational cost of this algorithm depends only linearly on the number of clusters. The structure of this algorithm closely resembles those found in Kalman filtering and smoothing theory [21, 25].

4. COMPENSATING MASS MATRIX TORQUE \( T_c(i) \)

Using Eq. (2.14) as a starting point, we now develop an expression for the compensating torque \( T_c(i) \) that is simple to compute. While we have already seen an expression for the mass matrix inverse, \( \mathbb{M}^{-1}(\theta) \), we need an expression for the derivative of the mass matrix with respect to hinge coordinates.
Spatial Operator Expression for $\mathcal{M}_H$

**Lemma 4.1.**

\[
\mathcal{M}_H = H\phi [H^\alpha \phi M - M\phi \alpha H^\alpha] \phi H^*. \tag{4.11}
\]

**Proof.** See Ref. [26].

The matrix $H^\alpha$ is a new quantity in this result. $H^\alpha$ is the $6n \times 6n$ matrix whose elements are all zero, except for a single $6 \times 6$ block $H(i)$ at the $i$th location on the diagonal. The index $i$ corresponds to the joint-angle $\theta_i$ with respect to which the sensitivity $\mathcal{M}_H$ is being taken. The nonzero block-diagonal element $H(i) \in \mathbb{R}^{6 \times 6}$ is obtained as follows from the hinge rotational axis unit vector $h(i)$:

\[
H(i) = \left( \begin{array}{cc}
\bar{h}(i) & 0 \\
0 & \bar{h}(i)
\end{array} \right). \tag{4.12}
\]

The notation $\bar{v}$ above denotes the $3 \times 3$ cross-product tensor matrix associated with a 3-vector $v = \left( \begin{array}{c} x \\ y \\ z \end{array} \right)$ and is defined as

\[
\bar{v} = \left( \begin{array}{ccc}
0 & -z & y \\
z & 0 & -x \\
-y & x & 0
\end{array} \right).
\]

The formula in Eq. (4.11) is closed-form, in the sense that it contains an explicit analytical expression for the mass matrix sensitivity in terms of the operators $\phi$, $M$, and $H$ appearing in the mass matrix itself. That the formula is closed-form is of extreme importance, because it implies that the mass matrix derivatives can be easily computed using operations and spatially recursive algorithms similar to those used to compute the mass matrix itself.

Spatial Operator Expressions for $T_c(i)$

The following lemma uses Lemma 4.11 to develop a new expression for $T_c(i)$.

**Lemma 4.2.**

\[
T_c(i) = \text{Trace} \{ P\Omega H^\alpha \phi M \}, \tag{4.13}
\]

where $\Omega \triangleq \psi \phi^* H^\alpha D^{-1} H\psi \in \mathbb{R}^{6n \times 6n}$.

**Proof.** Two spatial operator identities that we need for the proof are given by the following equations. Their derivation can be found in the appendix of Ref. [27]:

\[
[I - H\psi K]H\phi = H\psi \tag{4.14a}
\]

\[
\phi M\Omega = (\phi - \psi) + P\Omega. \tag{4.14b}
\]

The derivation of Eq. (4.13) goes as follows:

\[
T_c(i) \overset{2.11}{=} \frac{1}{2} \text{Trace} \{ \mathcal{M}_H^{-1}(\theta_i) \cdot [\mathcal{M}_H(\theta_i)] \}
\]

\[
\overset{3.5 \rightarrow 4.11}{=} \frac{1}{2} \text{Trace} \{ [I - H\psi K]^*D^{-1}[I - H\psi K] \}
\]

\[
= \text{Trace} \{ (H\phi[I H^\alpha \phi M - M\phi \alpha H^\alpha] \phi H^*) \}
\]

\[
\overset{4.14a}{=} \text{Trace} \{ (I - H\psi K) \phi H^\alpha \phi M \}
\]

\[
= \text{Trace} \{ \phi^* H^\alpha D^{-1} H\psi H^\alpha \phi M \}
\]

\[
\overset{4.14b}{=} \text{Trace} \{ \phi^* H^\alpha D^{-1} H\psi H^\alpha \phi M \}
\]

\[
\overset{4.13}{=} \text{Trace} \{ \Omega^* H\phi M \}
\]

\[
= \text{Trace} \{ \phi^* M\Omega \}
\]

\[
\overset{4.14b}{=} \text{Trace} \{ (\phi - \psi) + P\Omega \}
\]

In the above steps, we have used the fact that $\text{Trace} \{ AB \} = \text{Trace} \{ BA \}$. Also, the last step used the fact that $\text{Trace} \{ (\phi - \psi) H^\alpha \} = 0$. This is true because $(\phi - \psi)$ is strictly lower triangular and $H^\alpha$ is block diagonal.

It has been shown in Ref. [19] that $\Omega$ can be decomposed as

\[
\Omega = Y + \bar{\psi}^* Y + Y \bar{\psi}, \tag{4.15}
\]

where $\bar{\psi} \triangleq \psi - I$, and the diagonal elements $Y(k, k) \in \mathbb{R}^{6 \times 6}$ of the block diagonal matrix $Y \in \mathbb{R}^{6n \times 6n}$ are defined via the recursion:

\[
Y(n + 1) = 0
\]

for $k = n \cdots 1$

\[
Y(k) = \psi^* (k + 1, k) Y(k + 1) \psi (k + 1, k) \tag{4.16}
\]

\[+ H^\alpha (k) D^{-1} (k) H(k) \]

end loop
This allows us to further simplify the expression for $T_c(i)$ as described in the following lemma.

**Lemma 4.3.**

$$T_c(i) = \text{Trace} \{ P(i) Y(i) H(i) \} \tag{4.17}$$

**Proof.** Using Eq. (4.15) in Eq. (4.13) it follows that

$$T_c(i) = \text{Trace} \{ P \dot{\tilde{Y}} + \dot{\tilde{Y}} Y \} = \text{Trace} \{ P Y H \} = \text{Trace} \{ P(i) Y(i) H(i) \}.$$ 

Here we used the fact that $\text{Trace} \{ P Y H \} = 0$ because $\tilde{Y}$ is strictly lower triangular.

A final simplification step for $T_c(i)$ is described in the following lemma.

**Lemma 4.4.** Let the partitioned form of the $6 \times 6$ matrix $P(i)Y(i)$ be given by

$$P(i)Y(i) = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix},$$

where $Q_{ij} \in \mathbb{R}^{3 \times 3}$. Then

$$T_c(i) = -h(i) \mathcal{F}[Q_{11} + Q_{22}], \tag{4.18}$$

where the mapping $\mathcal{F}[:,] : \mathbb{R}^{3 \times 3} \to \mathbb{R}^3$ is defined via the relation:

$$v = \mathcal{F}[A] \text{ if and only if } \tilde{v} = A - A^*$$

**Proof.** We have

$$P(i)Y(i)H(i) = \begin{pmatrix} Q_{11} \tilde{h}(i) & Q_{12} \tilde{h}(i) \\ Q_{21} \tilde{h}(i) & Q_{22} \tilde{h}(i) \end{pmatrix}, \tag{4.19}$$

Therefore,

$$\text{Trace} \{ P(i) Y(i) H(i) \} = \text{Trace} \{ (Q_{11} + Q_{22}) \tilde{h}(i) \}.$$ 

Using the easily established identity that $\text{Trace} \{ A \tilde{v} \} = -v^* \mathcal{F}[A]$ for an arbitrary matrix $A \in \mathbb{R}^{3 \times 3}$ and vector $v \in \mathbb{R}^3$ in Eq. (4.19) establishes the result.

Notice that this expression for $T_c(i)$ is vastly simpler compared with the original expression involving the mass matrix inverse and its sensitivity. Furthermore, the expression involves articulated body inertia quantities many of whom are available from the computational steps for solving the equations of motion. It is easy to verify that the the compensating torque vector for the six base cluster degrees of freedom is zero.

**$O(\cdot \cdot \cdot )$ Constrained Dynamics Algorithm with Compensating Potential**

The overall algorithm for solving the equations of motion with the compensating mass matrix potential is defined by the following steps:

1. Carry out the first base-to-tip part of the recursion in Eq. (3.1) to compute the $V(k)$, $a(k)$ and $b(k)$ terms for all the clusters.

2. Carry out the tip-to-base recursion in Eq. (4.16) to compute all the articulated body inertia quantities such as $P_i, D_i$ etc.

3. Carry out the base-to-tip recursion in Eq. (4.16) to compute $Y(k)$ for all the links. Also compute the compensating torque $T_c(k)$ for each link simultaneously using Eq. (4.18) and $T'(k)$ using Eq. (2.10).

4. Carry out the recursions in Eq. (3.9) and Eq. (3.10) to solve for the $\dot{\theta}(k)$ hinge accelerations with $T(k)$ replaced with $T'(k)$.

The only significant change from the constrained dynamics algorithm in [2] is the additional Step 3 for the compensating torque. This additional step is also of $O(\cdot \cdot \cdot )$ computational complexity, and hence, the overall computational cost of the constrained dynamics algorithm remains $O(\cdot \cdot \cdot )$. Moreover this method adds only marginal cost since it makes use of the articulated body inertia quantities available from the computations for the regular solution for the internal coordinate accelerations. This algorithm makes possible the easy incorporation of the compensating potential into potential into constrained molecular dynamics simulations involving tree-topology molecular systems. This algorithm has been implemented as a part of the NEIMO software package.

### Extensions

Reference [28] describes an alternative method for computing the $Y(k)$ terms using dual articulated body inertias instead of Eq. (4.16). This method offers advantages for parallel implementation since these computations can be done concurrently with Step 2 instead of sequentially following it.

The derivations and algorithmic descriptions in the previous sections have focused upon serial-chain molecules with one degree of freedom rotational hinges. This was done for notational simplicity. The extension of the algorithm to tree-topology molecules is identical to the extension discussed in Ref. [2]. In this case the system has multiple tips and a single designated base cluster. To summarize, all the tip-to-base and base-to-tip recursions are replaced
by tips-to-base and base-to-tips recursions. At each hinge with branches, the recursions proceed through “scatter” and “gather” steps. To handle multiple degree of freedom hinges, it is simply a matter of recognizing that such hinges can be modeled as a sequence of single degrees of freedom hinges interconnected by pseudo-clusters of zero mass and inertia. With these modifications, the algorithm described above extends to general tree-topology molecular systems.

**Spatial Operator Expression for \( \gamma_c \)**

While explicit knowledge of the compensating potential \( \gamma_c(\theta) \) is not required during dynamics simulations, the nature and sensitivity of its dependence on the configuration coordinates can be used to study its effect on the statistical averages obtained from constrained dynamics simulations. The following lemma gives a closed form expression for the compensating potential \( \gamma_c \).

**Lemma 4.5.**

\[
\gamma_c(\theta) = \frac{1}{2} \sum_{i=1}^{N} \ln \det(D(i)).
\]  

**Proof.** From the innovations factorization in Eq. (3.5a), it follows that

\[
\det(\mathcal{A}) = \det(I + H\phi K)^2 \det(D)
\]

However, since \([I + H\phi K]\) is lower triangular with identity matrix blocks along its diagonal, its determinant is 1. Hence

\[
\det(\mathcal{A}) = \det(D).
\]

The result of the lemma follows simply by taking the logarithm of the above identity. \[ \square \]

The expression in Eq. (4.20) allows the easy computation of \( \gamma_c \) by making use of Algorithm 3.1 for computing the required \( D(k) \) quantities.

**5. CONCLUSIONS**

Internal rigid constraints, together with internal coordinates, are often used to speed up molecular dynamics computations. However, it has been recognized that the use of such constrained dynamics introduces systematic biases into the computation of statistical averages, and a compensating mass matrix potential is required to offset these biases. The lack of a tractable method for computing these compensating terms has rendered their use impractical to date.

This paper derives analytical expressions and algorithms for including the compensating mass matrix potential and its gradient into constrained MD simulations for general tree-topology molecular systems. The algorithm is an extension of the previously described \( O(\sqrt{N}) \) algorithms for internal coordinate molecular dynamics simulations. The computational complexity of the new algorithm remains \( O(\sqrt{N}) \). Extension of this compensating potential algorithm to closed-topology molecular systems is the subject of ongoing research.

**ACKNOWLEDGMENTS**

The author thanks Dr. N. Vaidehi for reviewing the paper and for many helpful suggestions. The research described in this paper has been performed partially at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration and with support from NSF Grant ACS 92-17368.

**REFERENCES**

17. A. Vaidehi, A. Jain, and W. Goddard, Constant temperature con-


