REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	9/18/96	Technical

4. TITLE AND SUBTITLE
An Efficient Newton-Type Iteration for the Numerical Solution of Highly Oscillatory Constrained Multibody Dynamic Systems

5. FUNDING NUMBERS
DAAL03-92-G-0247

6. AUTHOR(S)
Jeng Yen and Linda Petzold

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(ES)
University of Minnesota
Department of Computer Science
4-192 EE/CS Bldg, 200 Union St SE
Minneapolis, MN 55455

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
ARO 29850.5-MA

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. ABSTRACT *(Maximum 200 words)*

In this paper we present a coordinate-split (CS) technique for the numerical solution of the equations of motion of constrained multibody dynamic systems. We show how the coordinate-split technique can be implemented within the context of commonly used solution methods, for increase efficiency and reliability.

A particularly challenging problem for multibody dynamics is the numerical solution of highly oscillatory nonlinear mechanical systems. Highly stable implicit integration methods with large stepsizes can be used to damp the oscillation, if it is of small amplitude. However, the standard Newton iterations is known to experience severe convergence difficulties which forcea restriction of the step size. We introduce a modified coordinate-split (CM) iteration which overcomes these problems. Convergence analysis explains the improved convergence for nonlinear oscillatory systems, and numerical experiments illustrate the effectiveness of the new method.

14. SUBJECT TERMS	15. NUMBER IF PAGES
constrained dynamics, multibody systems, differential-algebraic equations, numerical methods, highly oscillatory systems	34
	16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

DEPARTMENT OF COMPUTER SCIENCE

UNIVERSITY OF MINNESOTA

INSTITUTE OF TECHNOLOGY
UNIVERSITY OF MINNESOTA
4-192 EE/CSci Building
200 Union Street SE
Minneapolis, Minnesota 55455
612/625-4002

TR 96-028

An Efficient Newton-Type Iteration for the Numerical Solution of Highly Oscillatory Contrained Multibody Dynamic Systems

by: Jeng Yen and Linda Petzold

Technical Report

Department of Computer Science
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 96-028

An Efficient Newton-Type Iteration for the Numerical Solution of Highly Oscillatory Contrained Multibody Dynamic Systems

by: Jeng Yen and Linda Petzold

An Efficient Newton-Type Iteration for the Numerical Solution of Highly Oscillatory Constrained Multibody Dynamic Systems *

Jeng Yen †
Linda R. Petzold ‡

March 28, 1996

Abstract

In this paper we present a coordinate-split (CS) technique for the numerical solution of the equations of motion of constrained multibody dynamic systems. We show how the coordinate-split technique can be implemented within the context of commonly used solution methods, for increased efficiency and reliability.

A particularly challenging problem for multibody dynamics is the numerical solution of highly oscillatory nonlinear mechanical systems. Highly stable implicit integration methods with large stepsizes can be used to damp the oscillation, if it is of small amplitude. However, the standard Newton iteration is known to experience severe convergence difficulties which force a restriction of the stepsize. We introduce a modified coordinate-split (CM) iteration which overcomes these problems. Convergence analysis explains the improved convergence for nonlinear oscillatory systems, and numerical experiments illustrate the effectiveness of the new method.

*The work was partially sponsored by the Army High Performance Computing Research Center under the auspices of the Department of Army, Army Research Laboratory cooperative agreement number DAAH04-95-2-003/contract number DDAH04-95-C-0008, and by ARO contract number DAAH04-94-6-0208.

†Department of Computer Science and Army High Performance Computing Research Center, University of Minnesota, Minneapolis, MN 55415.

‡Department of Computer Science and Army High Performance Computing Research Center, University of Minnesota, Minneapolis, MN 55455. The work of this author was partially sponsored by ARO contract number DAAL03-92-G-0247, and by DOE contract number DE-FG02-92ER25130, and by the Minnesota Supercomputer Institute.

1 Introduction

The equations of motion of a constrained multibody system can be written as [8]

$$\begin{aligned} \dot{q} - v &= 0 && \text{(1.1a)} \\ M(q)\dot{v} + G^T\lambda - f(v,q,t) &= 0 && \text{(1.1b)} \\ g(q) &= 0 && \text{(1.1c)} \end{aligned}$$

where $q = [q_1, q_2, ..., q_n]$ are the *generalized coordinates*, $\lambda = [\lambda_1, \lambda_2, ..., \lambda_m]$ are the *Lagrange multipliers*, $M(q) \in \mathbb{R}^{n\times n}$ is the mass-inertia matrix, $f \in \mathbb{R}^n$ is the force applied to the system, $\dot{q} = \frac{dq}{dt}$ is the velocity and $\ddot{q} = \frac{d^2q}{dt^2}$ is the acceleration vector. The constraints $g = [g_1, g_2, ..., g_m]$ are m smooth functions of q, whose Jacobian

$$G(q) = \left[\frac{\partial g_i}{\partial q_j}\right] \in \mathbb{R}^{m\times n}\ ,\ m \leq n \qquad (1.2)$$

is assumed of full row-rank. We assume that $G(q)M(q)G^T(q)$ is symmetric and positive definite for every $q \in \mathbb{R}^n$ to obtain a consistent physics represented by (1.1). The *degrees of freedom* for the system (1.1) is $n - m$. Equation (1.1) is a well-known index-3 DAE [3, 11].

Many methods have been proposed for modeling multibody systems. Direct numerical integration of the index-3 DAE (1.1) suffers from the well-known difficulties inherent in the solution of high-index DAEs [11]. One way to lower the index involves introducing derivatives of the constraint $g(q)$, along with additional Lagrange multipliers μ. This yields the *stabilized index-2* or *GGL* formulation of the constrained equations of motion [6]

$$\begin{aligned} \dot{q} - v + G^T\mu &= 0 && \text{(1.3a)} \\ M(q)\dot{v} + G^T\lambda - f(v,q,t) &= 0 && \text{(1.3b)} \\ G(q)v &= 0 && \text{(1.3c)} \\ g(q) &= 0, && \text{(1.3d)} \end{aligned}$$

which has been widely used in simulation. The Lagrange multiplier variables λ and μ fulfill the role of projecting the solution onto the *position* (1.3d) and the *velocity* (1.3c) constraints, respectively. Equations (1.3) and related systems have been solved by a variety of methods. Here we will consider solution by implicit numerical methods such as BDF or RADAU. A closely related approach is based on explicitly projecting the numerical solution onto the constraints [16, 18, 20, 21] and involves many of the same issues for the implementation that are considered here.

Many of the numerical methods for multibody systems solve the system (1.3) directly. It is also possible to eliminate the Lagrange multipliers and reduce the size

of the system to the number of degrees of freedom. One way to accomplish this begins with the stabilized index-2 system (1.3). Suppose that $G(p)$ is full-rank on the constraint manifold $\mathcal{M} = \{q \in \mathbb{R}^n \mid g(q) = 0\}$. Then one can find an annihilation matrix $P(q) \in \mathbb{R}^{(n-m)\times n}$ such that $P(q)G^T(q) = 0$, $\forall q \in \mathcal{M}$. Premultiplying (1.3a) and (1.3b) by $P(q)$ yields an index-1 DAE

$$\begin{aligned} P(q)(\dot{q} - v) &= 0 && \text{(1.4a)} \\ P(q)(M(q)\dot{v} - f(v,q,t)) &= 0 && \text{(1.4b)} \\ G(q)v &= 0 && \text{(1.4c)} \\ g(q) &= 0. && \text{(1.4d)} \end{aligned}$$

There is a potential gain in efficiency for this formulation due to the size-reduction of the nonlinear system, compared to (1.3). *An important practical consequence of (1.4) is that (μ, λ) have been eliminated from the DAE, via multiplication of (1.3a, 1.3b) by the nonlinear $P(q)$.* Thus, the error test and Newton iteration convergence test in a numerical implementation of (1.4) no longer need to include (μ, λ). These higher-index variables can cause problems in the direct numerical solution of (1.3). One could in principle also consider removing (μ, λ) from the test in the solution of (1.3), however it is not usually possible to justify this action, particularly in the case of the Newton convergence test. Elimination of these variables from the Newton convergence test in the solution of (1.3) can lead to a code which sometimes produces incorrect solutions. It is the fact that multiplying by the *nonlinear* $P(q)$ eliminates (μ, λ) from the nonlinear system, which allows these variables to be excluded from the tests in the solution of (1.4).

Direct numerical solution of (1.4) presents some challenges. First we must have a means of generating $P(q)$ which is reliable and cheap. Further, we note that the Jacobian matrix for the Newton iteration involves complicated terms which arise from the derivatives of $P(q)$. We need a means of generating the Jacobian matrix. Finally, practical issues such as the error test and Newton convergence test must be considered.

In the first part of this paper, we show how the numerical solution of (1.4) can be accomplished reliably and efficiently. In Section 2, we show how to obtain a cheap representation for $P(q)$, and then how to compute the Jacobian matrix without directly computing the complicated derivatives of P. We show that the nonlinear iteration converges. The effectiveness of this method for mechanical systems will be demonstrated in numerical experiments in Section 5.

Our approach for obtaining a cheap representation of $P(q)$ is based on a *coordinate-splitting* of the variables. A widely-used method which is related in the sense of also making use of a splitting of the coordinates is the generalized coordinate partitioning

method [20], which yields n-m differential equations

$$\hat{M}(x, h(x))\ddot{x} = \hat{f}(\dot{x}, \frac{dh}{dx}\dot{x}, x, h(x), t) \tag{1.5}$$

where $q = Xx + Yy$ such that $X \in \mathbb{R}^{n\times p}$ and $Y \in \mathbb{R}^{n\times m}$, whose columns constitute the standard basis for $\mathbb{R}^n$. The matrix Y is selected so that $(G(q)Y)^{-1}$ exists in a neighborhood of q, and $h(x)$ is the implicit function of y defined by the constraints. However, this differs substantially from the approach we outline here because $P(q)$ *associated with this method is not orthogonal to* $G^T(q)$. *Hence the index-reduction by differentiating the constraints and projecting to the invariant space must be carried out explicitly.* In particular, this requires forming the derivative of the velocity constraints (i.e., the acceleration constraints) explicitly. Another method for (1.4) has been proposed by [7, 16, 17, 18], where $\mathcal{P}$ is chosen to be an orthonormal basis of the local tangent space of the constraint manifold. Choosing a smoothly varying $\mathcal{P}$ is required and may cause some practical difficulties.

Direct numerical solution of (1.4) via our coordinate-split approach yields an efficient and reliable method for solving equations of motion for most multibody mechanical systems. However, there is a class of multibody systems which present additional computational challenges. These are the problems with high-frequency nonlinear oscillations. Highly oscillatory components are often used to model devices with strong potential energy. Typical examples of such problems arise from modeling flexible multibody mechanical, and molecular dynamic systems. For many problems, oscillations of a sufficiently small amplitude are not important for the model, but they severely restrict the stepsize for numerical methods. For these types of problems, stiffly stable implicit numerical integration methods can be used to damp out the oscillation [15]. However, the stepsize may still be severely restricted due to difficulties in converging the Newton iteration for larger stepsizes [15]. We have studied this class of oscillating problems in [22]. The solutions are composed of a low-amplitude high-frequency oscillation around a *smooth solution* [19]. Along the smooth solution, the eigenstructure of the local Jacobian matrix varies smoothly. However, along the solutions which are nearby to the smooth solution, the local eigenstructure oscillates with the high frequency, and is very badly behaved. The standard Newton iteration inside a damping numerical method starts from a *predictor* which is on a nearby solution, and attempts to find the smooth solution. It evaluates its Jacobian matrix on the nearby solution, which determines the direction it takes toward the smooth solution. Unfortunately, these Jacobian matrices do not yield good directions for nonlinear oscillating problems as described above, unless the predictor is already extremely close to the smooth solution. Thus, the standard Newton method must be coupled with a severe reduction in the timestep to achieve an adequate predictor.

In Section 3 we introduce a modification to the Newton iteration which we call the CM-iteration. This iteration is easy to implement, effective for non-oscillatory

problems, and particularly effective for nonlinear highly oscillatory problems. The basic idea of the CM-iteration is that there are terms in the Jacobian which involve derivatives of the projection onto the smooth solution. These terms are complicated to compute, large and oscillatory away from the smooth solution, and zero on the constraint manifold. The CM-iteration sets these terms to zero, yielding a reliable direction towards the smooth solution for the Newton-type iteration. In Section 3, we outline the CM-iteration, give details for its implementation, and prove its convergence. In Section 4 we describe in more detail the structure of nonlinear oscillatory mechanical systems, and derive estimates for the rates of convergence of the CS and CM-iterations applied to these oscillatory systems. The difference in convergence rate explains why the CM-iteration is highly effective for oscillatory systems, and shows that its rate of convergence for non-oscillatory systems is similar to that of the CS iteration. In Section 5, numerical experiments are given which demonstrate the effectiveness of these methods, particularly for oscillatory nonlinear mechanical systems.

2 The Coordinate-Split Technique

In this section we present the coordinate-split (CS) technique. We show how to define the matrix $P(q)$ via a coordinate splitting and how to compute this matrix cheaply. Although at first glance it would appear that implementation of this method would be difficult due to complications in computing the Jacobian of $P(q)$, we show that the special form of the pseudo-inverse can be used to give a much simpler derivation of the Jacobian. We outline an efficient implementation for solving the nonlinear system, and observe that the CS technique leads to a natural and effective error estimator. Finally, we prove the convergence of the CS iteration.

2.1 Construction of $P(q)$

The construction of the annihilation matrix $P(q)$ involves the solution of a class of pseudo-inverses of the constraint Jacobian $G(q)$. An effective way to obtain the projected vector $P(q)r$ is to use a splitting of the original coordinates.

Definition 2.1 *[Coordinate-Splitting Matrix] Let X and Y be the matrices whose columns constitute the standard basis of $\mathbb{R}^{n\times n}$ such that $\|(G(q)Y)^{-1}\|$ is bounded in a neighborhood U_0 of q_0. The $p \times n$* coordinate-splitting matrix *for (1.1) is defined by*

$$P(q) = X^T - Q(q)^T Y^T = X^T(I - G(q)^T(G(q)Y)^{-T}Y^T) \tag{2.1}$$

where $Q(q) = (G(q)Y)^{-1}G(q)X$.

Remark 2.1 *From the construction of the CS matrix $P(q)$, one can easily see that $P(q)G^T(q) = 0$ for all $q \in \mathbb{R}^n$, i.e., $P(q)$ is orthogonal to range(G^T). Furthermore, the row vectors of $P(q)$ are orthonormal, i.e., $P(q)^T P(q) = I_p$ where I_p is the identity matrix in $\mathbb{R}^p$.*

The computation of $P(q)$ can be carried out using the LU-factorization of the constraint Jacobian matrix. Applying Gaussian elimination with row-pivoting to G^T yields

$$E_m \cdots E_1 G^T = L_m \cdots L_1 U \tag{2.2}$$

where E_i is the elementary permutation and L_i is a Gauss transformation, $i \in \{1, 2, ..., m\}$. From the factorization, we have

$$[Y, X] = E = E_m \cdots E_1. \tag{2.3}$$

Using the standard solution technique by LU-decomposition, the projected vector $P(q)r$ can computed in a straightforward and relatively cheap way. In addition, the derivative (2.9) can be computed using the same factorization of G^T and the intermediate result $s = -(GY)^{-1}Y^T r$ from the computation of $P(q)r$.

Remark 2.2 *Alternatively, one can apply QR-factorization to G^T for the computation of $P(q)r$. Using QR-factorization with partial column pivoting [10], we obtain*

$$G^T \tilde{E}_1 \cdots \tilde{E}_m = \tilde{L}_1 \cdots \tilde{L}_m \tilde{U} \tag{2.4}$$

where $\tilde{E}_i$ is the elementary permutation, $\tilde{L}_i$ is the Householder matrix, $i \in \{1, 2, ..., m\}$, and $\tilde{U}$ is upper triangular. The last $(n - m)$ columns of the orthogonal matrix $\tilde{L} = \prod_{i=1}^{m} \tilde{L}_i$ constitute a basis for the null-space of G. Thus we can write

$$\left[\tilde{Y}, \tilde{X}\right] = \tilde{L} = \tilde{L}_1 \cdots \tilde{L}_m. \tag{2.5}$$

Note that $\tilde{X}$ and $\tilde{Y}$ are usually subsets of the standard basis in $\mathbb{R}^n$.

2.2 Computation of Jacobian and projected vector

The derivative of (2.1) can be obtained from the formulas given in [9] (Theorem 4.3, pp. 420). Here, we give a much simpler derivation of the Jacobian of $P(q)$, because of the special form of the pseudo-inverse in $P(q)$. For $G(q)$ a smooth function, we can in addition derive an approximation of the projected vector function $P(q)r$ using a nearby point.

Lemma 2.1 *Suppose $A(z) \in \mathbb{R}^{N\times N}$ is a nonsingular matrix-valued function of $z \in \mathbb{R}^N$, whose components are non-constant smooth functions of z. Then the derivative of $A(z)^{-1}r$ may be obtained as*

$$\frac{dA^{-1}(z)r}{dz} = -A^{-1}(z)\frac{d}{dz}[A(z)w], \text{ with } w = A^{-1}r \tag{2.6}$$

where $r, w \in \mathbb{R}^N$.

Proof. Let $r(z)$ be a smooth vector-valued function. Then the derivative of $A^{-1}(z)r(z)$ with respect to z leads to

$$\frac{d}{dz}(A^{-1}(z)r(z)) = \frac{d}{dz}A^{-1}(z)r + A^{-1}(z)\frac{dr(z)}{dz} \tag{2.7}$$

according to the product rule. Choosing $r(z) = A(z)w$ yields $A^{-1}(z)r(z) = w$ a constant vector, and (2.7) becomes

$$\frac{d}{dz}A^{-1}(z)r = -A^{-1}(z)\frac{dA(z)w}{dz} \tag{2.8}$$

where w is $A^{-1}r$. □

Lemma 2.2 *The derivative of $P(q)r$ is*

$$\frac{d}{dq}P(q)r = P(p)\frac{dG^Ts}{dq}, \text{ with } s = -(GY)^{-T}Y^Tr \tag{2.9}$$

where $P(q)$ is defined by (2.1) and $r \in \mathbb{R}^{n\times 1}$.

Proof. Differentiating $P(q)r$ with respect to q yields

$$\frac{d}{dq}P(q)r = \frac{d}{dq}X^T(I - Q^T(q)Y^T)r. \tag{2.10}$$

Since $\frac{dX^Tr}{dq}$ vanishes, differentiating (2.10) by the product rule yields

$$\frac{d}{dq}P(q)r = X^T\frac{dG^T(q)s}{dp} + (GX)^T\frac{d(G(q)Y)^{-T}Y^Tr}{dq} \tag{2.11}$$

where $s = -(GY)^{-T}Y^Tr$. According to (2.6) the second term on the right-hand side of (2.11) becomes

$$(GX)^T\frac{d(G(q)Y)^{-T}Y^Tr}{dq} = -Q^T\frac{d(GY)^Ts}{dq}.$$

This may be substituted back into (2.10) to obtain (2.9). □

2.3 Solving the nonlinear system

Here we outline an efficient implementation of the coordinate-split iteration for solving the nonlinear system at each time step, and show how the coordinate splitting leads to a natural and reliable error estimator. In particular, the local error estimators based on differences, i.e., the increments of the nonlinear iterations for the discretized nonlinear equations, can be obtained from those on the independent variables only.

Applying, for example, a BDF formula to (1.4) yields the nonlinear system

$$\begin{aligned} P(q_n)(\rho_h q_n - v_n) &= 0 && \text{(2.12a)} \\ P(q_n)(M(q_n)\rho_h v_n - f(v_n, q_n, t_n)) &= 0 && \text{(2.12b)} \\ G(q_n)v_n &= 0 && \text{(2.12c)} \\ g(q_n) &= 0 && \text{(2.12d)} \end{aligned}$$

where ρ_h is the discretization operator, and h the stepsize of the time discretization. Given an initial prediction $(q_n^{(0)}, v_n^{(0)})$, applying Newton-type methods to (2.12) requires the solution of linear equations

$$J(q_n, v_n)(\Delta q_n, \Delta v_n) = -r(q_n, v_n) \tag{2.13}$$

such that Δq_n and Δv_n are the increments of q_n and v_n,

$$J(q_n, v_n) = \begin{bmatrix} P(q_n)[\frac{\partial G(q_n)^T s_1}{\partial q_n} + \frac{\partial \rho_h q_n}{\partial q_n}] & -P(q_n) \\ P(q_n)[\frac{\partial G(q_n)^T s_2}{\partial q_n} + \frac{\partial r_2(q_n, v_n)}{\partial q_n}] & P(q_n)\frac{\partial r_2(q_n, v_n)}{\partial v_n} \\ \frac{\partial (G(q_n)v_n)}{\partial q_n} & G(q_n) \\ G(q_n) & 0 \end{bmatrix} \tag{2.14}$$

and

$$r(q_n, v_n) = [P_n r_1, P_n r_2, G_n v_n, g_n]$$

where $s_1 = -(GY)^{-T}Y^T r_1$, $s_2 = -(GY)^{-T}Y^T r_2$, $r_1 = \rho_h q_n - v_n$, and $r_2 = M(q_n)\rho_h v_n - f(v_n, q_n, t_n)$. The subscript n of a function represents its numerical value at t_n, e.g., $g_n = g(q_n)$.

To solve the Newton equations (2.13) efficiently, we apply a decoupling technique to the equations. To begin, we rewrite the first two equations of (2.13), i.e., corresponding to the derivatives of (2.12a) and (2.12b),

$$\begin{bmatrix} P(q_n) & 0 \\ 0 & P(q_n) \end{bmatrix} \left(J_1(q_n, v_n) \begin{bmatrix} \Delta q_n \\ \Delta v_n \end{bmatrix} + \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \right) = 0 \tag{2.15}$$

where the $2n \times 2n$ matrix J_1 is

$$J_1(q_n, v_n) = \begin{bmatrix} \frac{d\rho_h q_n}{dq_n} + \frac{d(G_n^T s_1)}{dq_n} & -I \\ \frac{d(M(q_n)\rho_h v_n)}{dq_n} - \frac{\partial f_n}{\partial q_n} + \frac{d(G_n^T s_2)}{dq_n} & M\frac{d\rho_h v_n}{dv_n} - \frac{\partial f_n}{\partial v_n} \end{bmatrix}. \tag{2.16}$$

Since J_1 is generally invertible under the assumption of $M(q_n)$ nonsingular, one solution of (2.15) can be computed by

$$(X^T - Q_n^T Y^T)\Delta\hat{q}_n = 0 \quad (2.17a)$$
$$(X^T - Q_n^T Y^T)\Delta\hat{v}_n = 0 \quad (2.17b)$$

where

$$J_1(q_n, v_n)\begin{bmatrix} \Delta\hat{q}_n \\ \Delta\hat{v}_n \end{bmatrix} = \begin{bmatrix} -r_1 \\ -r_2 \end{bmatrix}. \quad (2.18)$$

Note that the solution of (2.17a) is not the necessary but only a sufficient one to (2.15), i.e., there are other solutions to (2.15). Nevertheless, we will show in what follows that (2.17a) is consistent with the iterations for the constraints along the same direction.

Observing that Δq_n can be determined by (2.17a) and the last row of (2.13), i.e., $G_n\Delta q_n = -g_n$, the resulting increments can be used to compute $\frac{dGv}{dq}(q_n)\Delta q_n$, such that Δv_n is uniquely determined by (2.17b) and the third row of (2.13), e.g., $G_n\Delta v_n = -G_n v_n - \frac{dGv}{dq}(q_n)\Delta q_n$. Thus, $(\Delta q_n, \Delta v_n)$ can be obtained by solving two linear systems of the form

$$\begin{bmatrix} P_n \\ G_n \end{bmatrix} u = \begin{bmatrix} a \\ b \end{bmatrix} \quad (2.19)$$

for $u \in \mathbb{R}^n$, $a \in \mathbb{R}^{(n-m)}$, and $b \in \mathbb{R}^m$. It is important to note that (2.19) represents Gauss-Newton iteration along the column space of P^T. Since the a-vector in the right-hand side of (2.19) has been computed, we can then compute u with $b_2 = P_n\Delta\hat{q}_n$ for Δq_n and $b_2 = P_n\hat{\Delta}v_n$ for Δv_n.

Denoting $u = Xu_x + Yu_y$ and $p = n - m$, the first p equations of (2.19) are in the form

$$u_x - Q^T u_y = a \quad (2.20)$$

and the second m equations yield

$$u_y = (GY)^{-1}(b - (GX)u_x). \quad (2.21)$$

Substituting (2.21) into (2.20) and solving for u_x yields

$$(I_p + Q^T Q)u_x = (a + Q^T(GY)^{-1}b) \quad (2.22)$$

where I_p is the $p \times p$ identity matrix. Using the consistent projection vectors $P_n\Delta\hat{q}_n$ and $P_n\Delta\hat{v}_n$, we obtain

$$X^T\Delta q_n \equiv \Delta x_n = -(I_p + Q_n^T Q_n)^{-1}(P_n\Delta\hat{q}_n + Q_n^T(G_nY)^{-1}g_n) \quad (2.23a)$$
$$Y^T\Delta q_n \equiv \Delta y_n = -(G_nY)^{-1}(g_n + (G_nX)\Delta x_n) \quad (2.23b)$$

and

$$X^T\Delta v_n \equiv \Delta w_n = -(I_p + Q_n^T Q_n)^{-1}(P_n\Delta\hat{v}_n + Q_n^T(G_nY)^{-1}\eta_n) \quad (2.24a)$$
$$Y^T\Delta v_n \equiv \Delta z_n = -(G_nY)^{-1}(\eta_n + (G_nX)\Delta w_n) \quad (2.24b)$$

where $\eta_n = \frac{d(Gv_n)}{dq}\Delta q_n + G_n v_n$, and $v_n = Xw_n + Yz_n$. The numerical solutions (2.23) and (2.24) illustrate that the dependent variables y_n and z_n are determined geometrically along the *orthogonal complement* of the column space of P_n^T. The system dynamics is described by the local *independent generalized coordinates* x and its velocity w.

Implementation of simplified Newton method

Simplified Newton iterations are commonly used for the iterative solutions of (2.12). The increments by Newton's method to (2.12) are (2.23) and (2.24). For simplified Newton iteration that uses a fixed approximation $J_n^{(0)}$ of the Jacobian $J(q_n, v_n)$ in (2.13), the numerical solutions of (2.23) and (2.24) are computed by some fixed matrices. For instance, (2.23a) becomes

$$\Delta x_n = -(I_p + Q_n^{(0)^T} Q_n^{(0)})^{-1}(P_n\Delta\hat{q}_n + Q_n^{(0)^T}(G_n^{(0)}Y)^{-1}g_n)$$

where the superscript (0) indicates the function approximated at some $(q_n^{(0)}, v_n^{(0)})$. Since $(I_p + Q_n^{(0)^T} Q_n^{(0)})$ is symmetric positive-definite, the increments, under the weighted-norm induced by $diag\{I_p + Q_n^{(0)^T} Q_n^{(0)}, I_m\}$, become

$$\Delta x_n^S \equiv P_n\Delta\hat{q}_n^S + Q_n^{(0)^T}(G_n^{(0)}Y)^{-1}g_n \quad (2.25a)$$
$$\Delta y_n^S \equiv -(G_n^{(0)}Y)^{-1}(g_n + (G_n^{(0)}X)\Delta x_n^S) \quad (2.25b)$$
$$\Delta w_n^S \equiv P_n\Delta\hat{v}_n^S + Q_n^{(0)^T}(G_n^{(0)}Y)^{-1}\eta_n^S \quad (2.25c)$$
$$\Delta z_n^S \equiv -(G_n^{(0)}Y)^{-1}(\eta_n + (G_n^{(0)}X)\Delta w_n^S). \quad (2.25d)$$

where $\Delta\hat{q}_n^S$ and $\Delta\hat{v}_n^S$ result from

$$J_1(q_n^{(0)}, v_n^{(0)}) \begin{bmatrix} \Delta\hat{q}_n^S \\ \Delta\hat{v}_n^S \end{bmatrix} = \begin{bmatrix} -r_1 \\ -r_2 \end{bmatrix}. \quad (2.26)$$

Efficient implementation of a simplified Newton iteration for (2.12) is carried out using (2.2) as follows

Algorithm 2.1 *[CS iteration]*

Step 1. *Apply (2.2) to* $G^T(q_n^{(0)})$ *and* $J_1(q_n^{(0)}, v_n^{(0)})$ *as in (2.16).*

Step 2. *Solve for* $\Delta\hat{q}_n^S$ *and* $\Delta\hat{v}_n^S$ *in (2.26).*

Step 3. *Apply (2.2) to* $G^T(q_n)$*, then compute* $P_n\Delta\hat{q}_n^S$ *and* $P_n\Delta\hat{v}_n^S$*.*

Step 4. *Compute* $(G_n^{(0)}Y)^{-1}g_n$*, where* $b_2 = 0$*, then apply* $Q_n^{(0)^T}$ *to the result to obtain (2.25a).*

Step 5. *Use (2.25a) to compute* Δy_n^S*, where* $b_2 = \Delta x_n^S$*.*

Step 6. *Use (2.25a) and (2.25b) to compute* η_n^S*, then repeat* **Step 4 - 5** *for (2.25c) and (2.25d).*

For a straightforward implementation, the iterative solutions use (2.14) directly, which requires the solution of $4n$ projected vectors for the first $2(n-m)$ rows. For Algorithm 2.1, there is no need to compute the $4n$ vectors. Instead, two projected vectors and four solutions of a $m \times n$ linear system are required for each iteration in addition to the solution of a $2n \times 2n$ linear system, e.g., (2.14) and (2.16) for the straightforward and proposed implementations, respectively. Therefore, the above algorithm is preferred when n is large. Note that the computational cost of the iterative solution by Algorithm 2.1 is almost equal to that of the solution of a $2(n+m) \times 2(n+m)$ linear system. Using LU-decomposition, the solution of a $2(n+m) \times 2(n+m)$ linear system requires $O(4(n+m)^2)$ *flop*s, while the proposed iterative solution requires $O(6(nm)^2) + O(4n^2)$ *flop*s with an additional factorization and two solutions of the matrix $(G_nY)^{-T}Y^T$ in **Step 3** in the above algorithm.

Applying numerical integration, convergence of the iterative solutions by Algorithm 2.1 can be achieved if the initial guess $q_n^{(0)}$ from the predictor is close enough to the numerical solution q_n. In addition, the local error estimator based on the *predictor-corrector* difference can be modified for a more effective approximation. From (2.25b, 2.25d), the increments of y_n and z_n are bounded by

$$\|\Delta y_n^{(i)}\| \le \|(G_n^{(0)}Y)^{-1}\|(\|G_n^{(0)}X\|\|x_n^{(i)}\| + \|g_n^{(i)}\|)$$
$$\|\Delta z_n^{(i)}\| \le \|(G_n^{(0)}Y)^{-1}\|(\|G_n^{(0)}X\|\|w_n^{(i)}\| + \|\eta_n^{(i)}\|)$$

for all i. Ideally, the error estimator should be based on the *dynamics* of such systems, which is described by x and w. Under the assumption that the constraints are smooth, the local error can then be approximated using the alternative sequences of $\Delta\tilde{y}_n^{(i)}$ and $\Delta\tilde{z}_n^{(i)}$ by setting $g_n^{(i)}$ and $\eta_n^{(i)}$ to zero in (2.25b) and (2.25d), respectively. Since (q_n, v_n) converges in N iterations, the difference can be estimated by

$$\|(q_n, v_n) - (q_n^{(0)}, v_n^{(0)})\| = \sum_{i=1}^{N}(\|\Delta q_n^{(i)}\| + \|\Delta v_n^{(i)}\|) \approx \kappa_n \sum_{i=1}^{N}(\|\Delta x_n^{(i)}\| + \|\Delta w_n^{(i)}\|) \quad (2.27)$$

where $\kappa_n = \|(G_n^{(0)}Y)^{-1}G_n^{(0)}X\|$. Therefore, the local errors of numerical integration of (2.12) can be approximated using only the increments of $x_n^{(i)}$ and $w_n^{(i)}$, e.g., the right-hand side of (2.27).

2.4 Convergence of the CS iteration

For simplicity we now consider, instead of the second-order constrained equations of motion (1.1), a first-order system

$$\dot{q} - f(q,t) + G^T\lambda = 0 \tag{2.28a}$$
$$g(q) = 0 \tag{2.28b}$$

since the convergence of (2.28) can be trivially extended to (1.1). Applying stiffly stable numerical methods, the convergence result is well-known, see [11] pp. 494-498. Convergence of discretization methods for the index-1 system,

$$P(q)(\dot{q} - f(q,t)) = 0 \tag{2.29a}$$
$$g(q) = 0 \tag{2.29b}$$

obtained by applying the coordinate-splitting matrix $P(q)$ to (2.28a), is also well-developed. By the construction of $P(q)$, it is easy to see that the solution of the CS iteration is equivalent to that of the local state-space ODE of the independent coordinate x.

The convergence of the CS iteration can be carried out on a smooth constraint manifold $\mathcal{M}$. Assume that for any $q_0 \in \mathcal{M}$, there exist $X \in \mathbb{R}^{p\times n}$ and $Y \in \mathbb{R}^{m\times n}$ such that

$$\|(G(q)Y)^{-1}\| \le C_1 \tag{2.30}$$
$$\|G(q_1)^T - G(q_2)^T\| \le C_2\|q_1 - q_2\| \tag{2.31}$$

for some C_1 and C_2, where q, q_1, and q_2 are in a neighborhood $U(q_0)$ of q_0. Applying a linear discretization operator ρ_h with stepsize h to (2.29) yields a nonlinear system

$$F(q) = \begin{bmatrix} P(q)r(q) \\ g(q) \end{bmatrix}, \tag{2.32}$$

where the *residual* function is

$$r(q,t) = \rho_h(q) - f(q,t). \tag{2.33}$$

Let $\{q_j\}$ be the solution obtained by applying the simplified Newton method to (2.32), and $\{\tilde{q}_j\}$ be the solutions obtained by applying Algorithm 2.1. Convergence of the CS iteration can be shown under the conditions (2.30) and (2.31). Suppose that the iterative solutions $\{q_j\}$ from applying the simplified Newton method to (2.32) converge to q^*. For $\tilde{q}_0 = q_0$, $\{\tilde{q}_j\}$ generated by the CS iteration for (2.32) satisfies

$$\Delta\tilde{x}_k = (I + Q(q_0)^T Q(q_0))\Delta x_k,$$

where Δx_k is defined by the simplified Newton iteration. Consequently, $\{\tilde{q}_j\}$ converges to q^* as $\{q_j\}$ did.

Convergence of the CS iteration can be assured for a sufficiently accurate initial guess. Another sufficient condition for convergence requires that the numerical integration satisfies

$$\| \left(\frac{dr}{dq} + \frac{dG^T s}{dq} \right)^{-1} \| \leq C_0 < 1 \tag{2.34}$$

for some C_0 and $s = -(GY)^{-T} Y^T r$ in a neighborhood of q^*. This implies an upper bound on the stepsize h. For linear multistep integration, e.g., $\frac{\partial \rho_h(q)}{\partial q} = \frac{\beta}{h}$, the stepsize must satisfy

$$\| \left(\frac{\beta}{h} - \frac{\partial}{\partial q}(f - G^T s) \right)^{-1} \| \leq 1 \tag{2.35}$$

where β is the leading coefficient of the numerical integration formula.

For highly oscillatory dynamic systems with a wide frequency band, the error tolerance TOL of the numerical solution often satisfies $TOL \leq \max_{q_n} \| \frac{\partial f}{\partial q} \|$. Applying a stiffly stable numerical method to (2.29), such as BDF of order ≤ 2, one may take a larger stepsize to follow the trajectory of the equilibrium, i.e., $f - G^T s = 0$. However, convergence of the Newton iteration requires (2.35), a further restriction on the stepsize. Depending on how close the predictor is to the equilibrium of highly oscillatory components, the Newton direction imposed by the Jacobian can excite the high-frequency oscillations. When applying the Newton method directly to the discretization of (2.28), an even more severe problem in Newton convergence is observed, and illustrated by the numerical experiments in Section 5. The limitation on the stepsize due to the Newton convergence failures for highly oscillatory nonlinear multibody systems can be overcome via a modification of the CS iteration which we call the CM iteration.

3 Highly Oscillatory Systems and the CM Iteration

3.1 The CM iteration

In large-scale multibody mechanical systems, most of the unwanted oscillations are due to the *noise* of high-frequency forces, where the amplitude is well below the solution tolerance. However, small perturbations in the position can cause drastic changes in the Newton direction. This results in difficulties for convergence of Newton-type methods. To remedy this problem in the CS iteration, we reduce the noise from the oscillations by setting $\frac{dP(q)}{dq} r = 0$ in the Newton iteration matrix, since it is the main source contributing to the rapidly changing Newton direction. The basic idea

of the CM iteration is to approximate the Newton direction of (2.29) via an oblique projection to the unconstrained ODE

$$P(q_0)(\dot{q} - f(q,t)) = 0 \tag{3.1}$$

for a q_0 close to the solution q, e.g., $G(q_0)G^T(q)$ invertible. Since $P(q_0)$ is no longer varying with q, $\frac{dP}{dq}r = 0$ is attained. When applying a stiffly stable numerical integrator to highly oscillatory problems, this modification, for some q_0 close enough to the smooth solution, overcomes the difficulties in the CS iteration.

Applying a discretization method to (3.1) coupled with constraint (2.29b) leads to the nonlinear system

$$F_0(q) = \begin{bmatrix} P(q_0)r(q) \\ g(q) \end{bmatrix}. \tag{3.2}$$

The corresponding Lagrange multiplier form of (3.1) is

$$\dot{q} + G^T(q_0)\hat{\lambda} - f(q,t) = 0 \tag{3.3}$$

where

$$\hat{\lambda} = (G(q_0)Y)^{-T}Y^T(f(q,t) - \dot{q}).$$

A convergence result for the modified CS iteration, denoted by CM, is given in the following. A detailed algorithm for applying the CM iteration to (1.1) is presented at the end of this section.

We first give an upper bound for the difference between the derivative of the projected vector $P(q)r(q)$ and the projected derivative $P(q)\frac{dr(q)}{dq}$.

Lemma 3.1 *Suppose conditions (2.30) and (2.31) hold. Then*

$$\|\frac{d}{dq}[P(q)r(q)] - P(q)\frac{dr(q)}{dq}\| \leq \varrho_0 C_1 C_2 \|Y^T r(q)\| \tag{3.4}$$

in $D(q_0, \varrho_0) \subseteq U(q_0)$, where $D(q_0, \varrho_0)$ is the disc in $\mathbb{R}^n$ with center q_0 and radius ϱ_0.

Proof. The inequality is a direct consequence of (2.9). Subtracting (2.9) from $P(q)\frac{dr}{dq}$ and taking the norm of the remainder yields

$$\|\frac{d}{dq}[P(q)r(q)] - P(q)\frac{dr(q)}{dq}\| = \|P(q)\frac{dG(p)^T(GY)^{-1}Y^T r}{dq}\|.$$

Since the row vectors of $P(q)$ are p orthonormal vectors in $\mathbb{R}^n$, applying the Cauchy inequality gives

$$\|P(q)\frac{dG(p)^T(GY)^{-1}Y^T r}{dq}\| \leq \|\frac{dG(q)^T s}{dq}\| \leq \varrho_0 C_2 \|(GY)^{-T}Y^T r(q)\|$$

for all $q \in D(q_0, \varrho_0) \subseteq U(q_0)$. Condition (2.30) implies the result in (3.4). □

3.2 Convergence of the CM iteration

An estimation of the distance between the solutions of (2.32) and (3.2) is presented in the following.

Theorem 3.1 *Suppose conditions (2.30), (2.31), and*

$$\| \left(\frac{dr}{dq} \right)^{-1} \| \leq C_0 < 1 \tag{3.5}$$

hold in a neighborhood of q^* such that $\{q_j\}$ generated by the CS iteration converges to q^*. Choosing $\bar{q}_0 = q_0$, the sequence $\{\bar{q}_k\}$ generated by the CM iteration

$$\bar{q}_{k+1} = \bar{q}_k - \bar{J}(\bar{q}_0)^{-1} F_0(\bar{q}_k) \tag{3.6}$$

where $\bar{J} = \frac{dF_0}{dq}$, converges to $\bar{q}^*$. Furthermore, the distance between $\bar{q}^*$ and q^* is bounded above by

$$\|\bar{q}^* - q^*\| \leq C(\|Y^T r(q^*)\| \|q^* - q_0\| + \|q^* - q_0\|^2 + \|\bar{q}^* - q_0\|^2) \tag{3.7}$$

for some moderate constant C.

Proof. Since $\bar{J}$ is nonsingular and its components are smooth functions, using (2.21) and (2.22) we can write

$$\bar{J}^{-1} = \begin{bmatrix} I_p + Q^T Q & 0 \\ -Q & I_m \end{bmatrix}^{-1} \begin{bmatrix} I_p & Q^T (GY)^{-1} \\ 0 & (GY)^{-1} \end{bmatrix} \begin{bmatrix} dr \\ dq \end{bmatrix}^{-1}$$

for the CM iteration, provided that $\frac{dr}{dq}$ is invertible. By conditions (2.30) and (2.31), we have

$$\| \begin{bmatrix} I_p + Q^T Q & 0 \\ -Q & I_m \end{bmatrix}^{-1} \| \leq C_3$$

and

$$\| \begin{bmatrix} I_p & Q^T (GY)^{-1} \\ 0 & (GY)^{-1} \end{bmatrix} \| \leq C_4$$

for some constants C_3 and C_4. Thus, the contractive condition (3.5) implies convergence of the CM iteration.

To show (3.7), we first observe from (3.3) that

$$r(\bar{q}^*) + \mathcal{G}(q_0) r(\bar{q}^*) = 0 \tag{3.8}$$

where

$$\mathcal{G}(q_0) = -G^T(q_0)(G(q_0)Y)^{-T} Y^T.$$

Also, q^* is a solution of (3.3), i.e.,

$$r(q^*) + \mathcal{G}(q^*)r(q^*) = 0. \tag{3.9}$$

Using (3.8) and (3.9), and adding and subtracting $\mathcal{G}(q_0)r(q^*)$ yields

$$(I + \mathcal{G}(q_0))(r(\bar{q}^*) - r(q^*)) = (\mathcal{G}(q_0) - \mathcal{G}(q^*))r(q^*).$$

Applying the mean-value theorem to the differentiable functions r and $\mathcal{G}$, we have

$$(I + \mathcal{G}(q_0))\frac{dr}{dq}(\tilde{q}_1)(\bar{q}^* - q^*) = \frac{d\mathcal{G}}{dq}(\tilde{q}_2)r(q^*)(q_0 - q^*)$$

for some $\tilde{q}_i$ $i = 1, 2$. Premultiplying the above equation by X^T yields

$$P(q_0)\frac{dr}{dq}(\tilde{q}_1)(\bar{q}^* - q^*) = X^T\frac{d\mathcal{G}}{dq}(\tilde{q}_2)r(q^*)(q_0 - q^*).$$

Since $g(\bar{q}^*) - g(q^*) = 0$, we obtain

$$\begin{bmatrix} P(q_0) \\ G(q_0) \end{bmatrix}(\bar{q}^* - q^*) = \begin{bmatrix} X^T\frac{d\mathcal{G}}{dq}(\tilde{q}_2)r(q^*)(q_0 - q^*) \\ \frac{dG}{dq}(q_0)(q^* - q_0)^2 + \frac{dG}{dq}(q_0)(\bar{q}^* - q_0)^2 \end{bmatrix},$$

using the expansion of $g(q^*)$ and $g(\bar{q}^*)$ around q_0. From (2.21), (2.22) and the assumption of an invertible $\begin{bmatrix} P(q_0) \\ G(q_0) \end{bmatrix}$, we can write

$$\|\bar{q}^* - q^*\| \leq C_5\|X^T\frac{d\mathcal{G}}{dq}r(q^*)\|\|q_0 - q^*\| + C_6(\|q^* - q_0\|^2 + \|\bar{q}^* - q_0\|^2)$$

for some C_5 and C_6. This implies (3.7). □

Note that (3.5) for the CM method is analogous to (2.34) for the CS method. Instead of (2.35) for multistep integration methods using the CS iteration, the stepsize condition for the CM iteration is given by

$$\| \left(\frac{\alpha}{h} - \frac{\partial f}{\partial q}\right)^{-1} \| \leq 1 \tag{3.10}$$

in accordance with (3.5).

It is easy to see that $\{q_k\}$ of the CS iteration and $\{\bar{q}_k\}$ of the CM iteration are the same if the constraints $g(q)$ are linear. In general, the rate of convergence of the CM iteration is *superlinear*, using the Dennis-Moré Characterization Theorem [5].

3.3 Implementation of the CM iteration

The CM iteration can be implemented via the same procedures as those in Algorithm 2.1, but the computational cost is considerably lower. First, the Jacobian of (2.13) becomes

$$\bar{J}_1(q_n, v_n) = \begin{bmatrix} \frac{d\rho_h q_n}{dq_n} & -I \\ \frac{d(M(q_n)\rho_h v_n)}{dq_n} - \frac{\partial f_n}{\partial q_n} & M\frac{d\rho_h v_n}{dv_n} - \frac{\partial f_n}{\partial v_n} \end{bmatrix} \tag{3.11}$$

since $P(\hat{q})$ is held fixed in the nonlinear equations. Comparing (3.11) to the Jacobian (2.16), notice that the terms due to the derivatives of $P(q)r$ have vanished. Second, the solutions of $P_n\Delta\hat{q}_n^S$ and $P_n\Delta\hat{v}_n^S$ at each iteration in Algorithm 2.1 are replaced by $P(\hat{q})\hat{q}_n^S$ and $P(\hat{q})\hat{v}_n^S$, which reduces the cost since $P(\hat{q})$ has been previously computed.

Using the simplified Newton iteration for (2.12), the CM iteration is carried out as follows:

Algorithm 3.1 *[CM iteration]*

Step 1. *Apply (2.2) to $G^T(\hat{q})$ and $\bar{J}_1(\hat{q}, \hat{v})$ as in (3.11).*

Step 2. *Solve for $\Delta\hat{q}_n^S$ and $\Delta\hat{v}_n^S$ in (2.26).*

Step 3. *Compute $P(\hat{q})\Delta\hat{q}_n^S$ and $P(\hat{q})\Delta\hat{v}_n^S$.*

Step 4.-6. *same as Algorithm 2.1.*

Remark 3.1 *When applying the CM method to (1.1), the residual function*

$$r_2(v, q, t) = M(q)\rho_h v - f(v, q, t)$$

should be replaced by

$$\hat{r}_2(v, q, t) = \rho_h v - M(q)^{-1} f(v, q, t)$$

for a nonsingular $M(q)$.

4 Rate of convergence for highly oscillatory multibody systems

High frequency oscillatory forces often appear in the modeling of vehicle suspension systems, modal analysis in structural dynamics, or modeling oscillations in computer-aided engineering etc. For simplicity, we consider the constrained dynamic system of

(1.1) with a dominant oscillatory force

$$M(q)\dot{v} + G^T\lambda + \frac{1}{\epsilon}\eta(q) - f(v,q,t) = 0 \quad (4.1a)$$

$$g(q) = 0 \quad (4.1b)$$

where $\frac{1}{\epsilon}$ may be, for example, the coefficients of stiff springs; i.e., $0 < \epsilon \ll 1$. In practice, $\eta(q)$ is usually oblique towards $KerP(q)$, i.e., the oscillatory force(s) acts on both the independent and the dependent coordinates. For the purpose of obtaining a smooth solution with large stepsizes solving those types of problems [15], we will show that the CM iteration can be very effective for many classes of nonlinear oscillatory forces.

In the modeling of deformable multibody systems, the nonlinear oscillatory forces in (4.1) are usually derived from the theory of linear elasticity, i.e., for some functions $\tilde{q}$ such that the oscillatory forces may be written as $\frac{1}{\epsilon}\tilde{q}$. We can use these functions $\tilde{q}$ to write the nonlinear force, e.g.

$$\frac{1}{\epsilon}\eta(q) = \frac{1}{\epsilon}\tilde{q},$$

and then append

$$\tilde{q} - \eta(q) = 0$$

to the constraint equations. The oscillatory forces then become *linear* with respect to the variables $\tilde{q}$. In fact, if the oscillatory forces are produced by a finite element approximation of the deformation of bodies, components of $\tilde{q}$ are associated with some body-fixed local coordinates via the orientation transformation matrix, whose entries often are slowly varying in time.

Deformation forces are the most common potential forces that can produce small amplitude high-frequency oscillations, and they are usually linear with respect to the local coordinates [4, 23]. For these reasons, we consider the class of oscillatory forces in the form

$$\eta(q) = B(t)(q - b_0(t)) \quad (4.2)$$

where the components of B and b_0 are slowly varying. In particular, B and b_0 may be functions of some constraint-driven generalized coordinates. For example, $B(\theta)$ in the 2D bushing problem in [22] has the form

$$B(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} k^x & 0 & 0 \\ 0 & k^y & 0 \\ -k^y\sin\theta & k^x\cos\theta & k^\theta \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \quad (4.3)$$

where k^x, k^y and k^θ are positive constants. When $\theta(t)$ is smooth or constrained, assumption (4.2) is valid.

Using a linear oscillatory force, the Lagrange equations of motion of the multibody system can be written as

$$M(q)\dot{v} + \frac{1}{\epsilon}B(q - b_0) + G^T\lambda - f(v, q, t) = 0 \tag{4.4}$$

where $\frac{1}{\epsilon} \gg \|\frac{\partial f}{\partial(q,v)}\|$. From assumption (2.31) on the constraint manifold, we can also assume that

$$\frac{1}{\epsilon} \gg \max_{\|u_1\|,\|u_2\|=1} \|\frac{dG(q)u_1}{dq}u_2\| \tag{4.5}$$

for all q.

In the context of the CS iteration, the problem of convergence of the Newton iteration can be explained by analyzing the *reduced potential* function. The *reduced potential* of (4.1) is

$$V^r(q) = g(q)^T(GY)^{-T}Y^T r \tag{4.6}$$

where $r = f - M\ddot{q} - \frac{1}{\epsilon}B(q - b_0)$. The *reduced potential force* generated by (4.6) is

$$\nabla V^r(q) \equiv \frac{dV^r}{dq} = G^T(GY)^{-T}Y^T r. \tag{4.7}$$

At each iteration, the reduced potential force acts along the *normal* direction of the constraint manifold. The gradient of the correction term yields

$$\nabla^2 V^r(q) = (I - G^T(GY)^{-T}Y^T)\frac{dG^T(q)s}{dq} \tag{4.8}$$

where $s = (GY)^{-T}Y^T r$. Applying Y^T to (4.8), gives

$$Y^T\nabla^2 V^r(q) = Y^T(I - G^T(GY)^{-T}Y^T)\frac{dG^T(q)s}{dq} = 0$$

and applying X^T to (4.8) yields

$$X^T\nabla^2 V^r(q) = P(q)\frac{dG^T(q)s}{dq}.$$

When high-frequency oscillations appear in the system, e.g., $\epsilon \to 0$, the reduced potential force also becomes oscillatory if $Y^T r$ is nonzero. This is the general case when the solution is not at an equilibrium position. Nevertheless, convergence of the CS iteration can be achieved by using a small enough stepsize, e.g., $h \approx \sqrt{\epsilon}$.

Theorem 4.1 *Let (q, v) be the solution of the nonlinear system which results from numerical integration of (4.1) using ρ_h with a stepsize h. Suppose the starting value (q_0, v_0) satisfies $\|q_0\| = O(h^2)$ and $\|v_0\| = O(h)$, and $J(q_0)$ is nonsingular. Then the CS iteration converges if $h^2 \leq c\epsilon$ for some moderate c.*

Proof. For the convergence of the CS iteration, we need to show that (2.34) is valid, where $r(q)$ is defined in (4.4). For (2.34), we have

$$\|\frac{\partial r}{\partial q}(q_0) + \frac{dG^T s}{dq}(q_0)\| = \|\frac{\beta\|M\|}{h^2} - \frac{1}{\epsilon}\| + O(h)$$

where $\beta > 0$ is the leading coefficient of ρ_h, and $\|M\|$ is not zero. Consequently, for $\epsilon \ll 1$, (2.34) is valid provided that $h \approx \sqrt{\epsilon}$. □

Under the conditions of the above theorem, a convergence result for the CM iteration can be obtained provided the assumptions of Theorem 3.1 are valid. In many applications, following the oscillations is *not* of interest. Instead, one wants to use a large time step to damp out the oscillations of small amplitude but high frequency. For this reason, we now consider only the multistep numerical integration methods that are *strictly stable* at infinity and A-stable, such as the lower order (i.e., ≤ 2) BDF methods [11]. The convergence of L-stable implicit Runge-Kutta methods to the *smooth* solution of the highly oscillatory ODE of multibody mechanical systems can be found in [15]. Here we focus on the convergence of the CM iteration for constrained multibody systems with oscillatory forces when applying the above-mentioned linear multistep methods.

Numerical solutions on the *slow manifold* can be evaluated using the equilibrium of (4.1), i.e., the *slow* solution [2, 13] satisfies

$$\eta(q) - \epsilon(f(v,q) - \nabla V^r(q) - M(q)\dot{v}) = 0,$$

and the *smooth* solution is its asymptotic expansion to some order of ϵ around the manifold $\{q \mid \eta(q) = 0\}$. In the linear form, the smooth solution of (4.1) is not far from $B(q - b_0) = 0$ since $\frac{1}{\epsilon} \gg \|\frac{\partial f}{\partial(q,v)}\|$. For the strongly damped numerical solution q_n, $B(q_n - b_0) \to O(\epsilon)$ as $t_n \to \infty$. During the iterative solution onto the *slow manifold*, the constraints may not be satisfied, which causes a large reaction force in the form of (4.7). This may cause oscillations in the CS iteration, while the CM iteration annihilates these nonlinear oscillations generated by the reduced potential. This yields a superior performance of the CM iteration as compared to the CS iteration for computing the smooth solution of (4.1). The result is explained in the following.

Lemma 4.1 *Let (q^*, v^*) be the smooth solution of (4.1), $\eta(q)$ linear, and h the step-size of the multistep integration method. Suppose the starting values (q_0, v_0) for (q^*, v^*) on the smooth solution of (4.1), i.e., $\|q^*\| = O(\epsilon)$ and $r(q^*, v^*) = O(h)$, satisfy (2.30), (2.31) and*

$$M(q_0)\rho_h(v_0) - f(v_0, q_0) = O(h) \tag{4.9}$$

where ρ_h is the corresponding discretization operator. Applying the CS and CM iterations to (4.1), the approximate Jacobian matrix for the CS iteration satisfies

$$\|J(q_0, v_0) - J(q^*, v^*)\| = \frac{\delta}{\epsilon} O(h) + O(h) \tag{4.10}$$

where $\delta = \|Bq_0 - Bq^*\|$, and $J(q, v)$ is the Jacobian of (4.1). For the CM iteration, we have

$$\|\bar{J}(q_0, v_0) - J(q^*, v^*)\| = O(h) \tag{4.11}$$

where $\bar{J}$ is the approximate Jacobian in the CM iteration.

Proof. The difference between the Jacobian at (q_0, v_0) and (q^*, v^*) can be written as

$$\|J_0 - J^*\| \leq \|P(q_0)\frac{\partial r}{\partial q}(q_0) - P(q^*)\frac{\partial r}{\partial q}(q^*)\| + \|\frac{dP}{dq}(q_0)r(q_0) - \frac{dP}{dq}(q^*)r(q^*)\| + O(h)$$

since the initial values satisfy (4.9). Under the conditions (2.30) and (2.31), we may choose common X and Y for $P(q_0)$ and $P(q^*)$ such that the first term on the right-hand side of the above inequality can be rewritten as

$$\|P(\hat{q})(\frac{\partial r}{\partial q}(q_0) - \frac{\partial r}{\partial q}(q^*))\| = O(h)$$

for some $\hat{q} \in [q_0, q^*]$, since $\frac{\partial r}{\partial q} = \frac{1}{\epsilon}B + O(h)$ allowing the cancellation of $\frac{1}{\epsilon}B$. The second term yields

$$\|\frac{dP}{dq}(q_0)r(q_0) - \frac{dP}{dq}(q^*)r(q^*)\| \leq \|r(q_0) - r(q^*)\|O(h) = \frac{1}{\epsilon}\|Bq_0 - Bq^*\|O(h)$$

according to Lemma 3.1. Thus, (4.10) is proved. Recalling $\bar{J}(q_0, v_0)$ from (3.11), we have

$$\|\bar{J}_0 - J^*\| \leq \|P\|\|\frac{\partial^2 r}{\partial q^2}\| = O(h),$$

using again Lemma 3.1. □

Theorem 4.2 *For the initial values (q_0, v_0), suppose the conditions in Lemma 4.1 hold. Suppose that the CS and the CM iterations are carried out by applying a simplified Newton method, where the iteration matrix is computed at the starting values (q_0, v_0). If both iterations converge, then the rate of convergence of the CS iteration $\sigma^{(CS)}$ compared to that of the CM iteration $\sigma^{(CM)}$ is given by*

$$\sigma^{(CS)} = \frac{\delta}{\epsilon} O(h) + O(h)$$

where $\delta = \|B(q_o - q^*)\|$, and

$$\sigma^{(CM)} = O(h).$$

Proof. Since we apply the simplified Newton iteration, the solution of the CS iteration can be written as

$$\begin{bmatrix} q_{k+1} \\ v_{k+1} \end{bmatrix} = H(q_k, v_k)$$

where

$$H(q,v) = \begin{bmatrix} q \\ v \end{bmatrix} - J_0^{-1} F(q,v).$$

Similarly, the CM iteration can be written as the fixed-point iteration of the function

$$\bar{H}(q,v) = \begin{bmatrix} q \\ v \end{bmatrix} - \bar{J}_0^{-1} F(q,v).$$

Applying the Contractive Mapping Theorem, see [5] pp. 93-94, we obtain the rates of convergence of the CS and CM iterations

$$\sigma^{CS} = \|I - J_0^{-1} J(q^*, v^*)\| = \|J_0^{-1}(J_0 - J^*)\|$$

and

$$\sigma^{CM} = \|I - \bar{J}_0^{-1} J(q^*, v^*)\| = \|\bar{J}_0^{-1}(\bar{J}_0 - J^*)\|$$

where $J(q^*, v^*) = J^*$ is the Jacobian at the solution of the discretized system, and the superscripts denote the respective iterations. For J_0^{-1}, we have

$$J_0 = \begin{bmatrix} \frac{\beta}{h} + O(h) & -I \\ \frac{\delta}{\epsilon} B + O(h) & \frac{\beta}{h} M + O(h) \end{bmatrix}.$$

When $\epsilon \to 0$, the dominant components of J_0^{-1} are of $O(1)$. From Lemma 4.1, the rates are

$$\sigma^{CS} = \frac{\delta}{\epsilon} O(h) + O(h)$$

and

$$\sigma^{CS} = O(h),$$

since $\bar{J}_0^{-1}$ has no component of $O(\frac{\delta}{\epsilon})$. □

5 Numerical Experiments

5.1 Point-mass with oscillatory force

The first example is a simple constrained multibody system under the influence of a highly oscillatory force. Consider a unit point-mass constrained to the 2D unit

circle, using $q = [x, y]^T$, the velocity $v = \frac{dq}{dt} = [w, z]^T$, and the constraint equation $g(q) = \frac{1}{2}(x^2 + y^2 - 1)$. The equations of motion are

$$\begin{aligned} \dot{w} + x\lambda - f^x &= 0 \\ \dot{z} + y\lambda - f^y &= 0 \end{aligned}$$

where $f = [f^x, f^y]^T$ is the applied force. Differentiating the constraint $g(q) = \frac{1}{2}(x^2 + y^2 - 1)$ twice with respect to time, an explicit form of the multiplier λ is obtained by

$$\lambda = \frac{1}{x^2 + y^2}(xf^x + yf^y + w^2 + z^2).$$

For a highly oscillatory force $f(q)$, one can see that $\lambda(t)$ is oscillating with the frequency of $f(q)$, and with amplitude proportional to the magnitude of $\|f\|$.

The numerical experiments are carried out using BDF of order ≤ 2 in DASSL [3], where the local error estimation has been modified. For the stabilized index-2 DAE (1.3) denoted by GGL, the local error is estimated using only the position, i.e., q. Moreover, we have also included some experiments where the Newton convergence test of GGL has been modified to exclude the multipliers. The corresponding numerical solution is denoted by GGL^*. For the coordinate-split and modified coordinate-split iterations, denoted by CS and CM, respectively, the local error is estimated using the independent variable $X^T q$, as recommended in Section 2.3. The CM iteration updates the matrix $P(\hat{q})$ when a new Jacobian is required.

Linear oscillation

Let a unit gravitational force act along the negative y-direction, and apply a linear oscillatory force

$$f = \begin{bmatrix} \frac{1}{\epsilon}x \\ \frac{1}{\epsilon^2}(y + 1) - 1.0 \end{bmatrix}.$$

There is a stable *equilibrium* at $q = [0, -1]$ and $v = [0, 0]$. The natural frequency of the system is $\omega = \frac{1}{\epsilon}$, and no dissipative force is present.

The numerical solution has been carried out with a moderate solution tolerance $ATOL = RTOL = TOL = 10^{-3}$. For a 0 to 0.25 second simulation, the results of several combinations of the stiffness coefficient ϵ are presented in Table 5.1, where the initial values are $q = [0.0, -1.0]$ and $v = [1.0, 0]$. The CM iterations show better efficiency than those of CS, GGL and GGL^* in all cases, i.e., comparing the numbers of function and Jacobian evaluations in Table 5.1. In the Table, $etfs$ and $ctfs$ denote the number of failures of the error test and Newton convergence test, respectively, in DASSL. Comparing the results of GGL^* with those of GGL, we observe an improved Newton convergence. As $\epsilon \to 0$, i.e., for higher frequencies of the oscillation, the CM iteration becomes even more efficient. In Figure 5.1, we plot the total energy of

Method	TOL	ϵ	*no. steps*	*no. fevals.*	*no. jevals.*	*no. etfs.*	*no. ctfs.*
GGL	10^{-3}	10^{-4}	446	1713	263	47	0
GGL^*	10^{-3}	10^{-4}	478	1304	127	45	0
CS	10^{-3}	10^{-4}	381	1036	106	46	0
CM	10^{-3}	10^{-4}	373	963	96	38	0
GGL	10^{-3}	10^{-5}	431	1726	261	34	2
GGL^*	10^{-3}	10^{-5}	500	1324	139	52	2
CS	10^{-3}	10^{-5}	447	1281	116	56	2
CM	10^{-3}	10^{-5}	456	1180	115	58	1
GGL	10^{-3}	10^{-6}	427	1693	275	41	8
GGL^*	10^{-3}	10^{-6}	478	1247	123	44	4
CS	10^{-3}	10^{-6}	414	1139	119	38	5
CM	10^{-3}	10^{-6}	325	844	81	34	1

Table 5.1: Results of the Constrained Point-Mass with a Linear Oscillatory Force

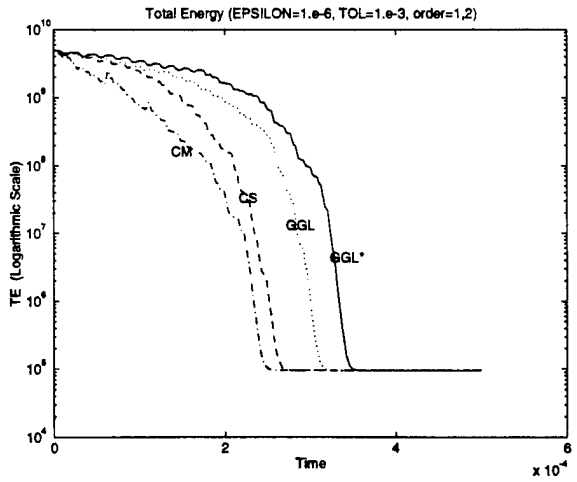

Figure 5.1: Total Energy Comparison of Linear Oscillatory Force Example; $\epsilon = 10^{-6}$ and $TOL = 10^{-3}$

Method	TOL	ϵ	*no. steps*	*no. fevals.*	*no. jevals.*	*no. etfs.*	*no. ctfs.*
GGL	10^{-3}	10^{-4}	109	383	56	13	0
GGL^*	10^{-3}	10^{-4}	109	337	45	13	0
CS	10^{-3}	10^{-4}	150	474	58	20	0
CM	10^{-3}	10^{-4}	72	208	31	7	0
GGL	10^{-3}	10^{-5}	105	416	68	11	0
GGL^*	10^{-3}	10^{-5}	105	387	47	12	0
CS	10^{-3}	10^{-5}	135	455	56	18	0
CM	10^{-3}	10^{-5}	66	205	32	4	0
GGL	10^{-4}	10^{-4}	822	2534	337	78	0
GGL^*	10^{-4}	10^{-4}	818	2572	242	82	0
CS	10^{-4}	10^{-4}	943	2727	217	66	0
CM	10^{-4}	10^{-4}	193	577	56	14	0

Table 5.2: Results of the Constrained Point-Mass with an Oscillatory Linear Spring Force

each numerical solution. The CM iteration achieves the strongest damping because DASSL is able to increase the stepsize faster with the CM iteration.

Linear spring force

In the next test, we replace the linear oscillatory force in the previous constrained system by a spring force

$$\frac{1}{\epsilon^2}\frac{l-l_0}{l}\begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix}$$

where $l = \sqrt{(x-x_0)^2+(y-y_0)^2}$ for (x_0, y_0) the attachment point of the spring, l_0 the natural length, and $\frac{1}{\epsilon^2}$ the stiffness coefficient, as shown schematically in Figure 5.2. For unit mass and unit gravitational force, we set the spring attached to $(x_0, y_0) = (0, -0.5)$ and the natural length $l_0 = 0.4$, such that the equilibrium is at $(0, -1, 0, 0)$.

Using the initial conditions $[0.04471, -0.999, 0, 0]$, the results of the 0-0.05 second simulation by the GGL, GGL^*, CS, and CM iterations are shown in Table 5.2. Because DASSL is able to increase the stepsize, and hence damp the solution faster with the CM iteration, the CM method is quite effective in these tests. In Figure 5.3, we plot the total energy of each solution. The numerical solutions of x, w, and λ are presented in Figure 5.4.

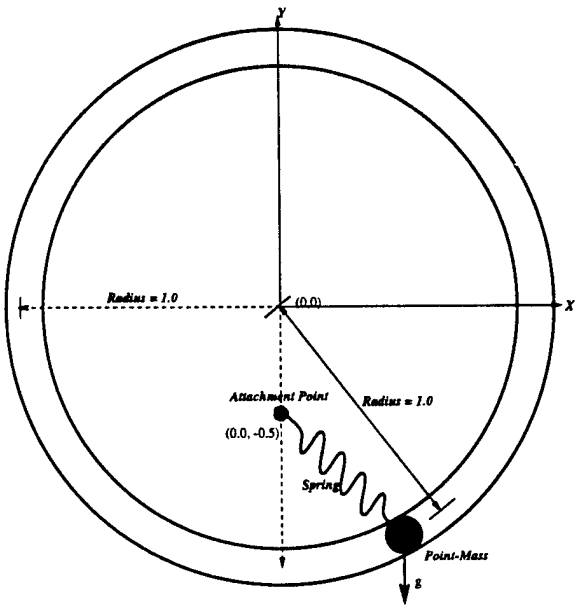

Figure 5.2: Constrained Point-Mass with a Linear Spring

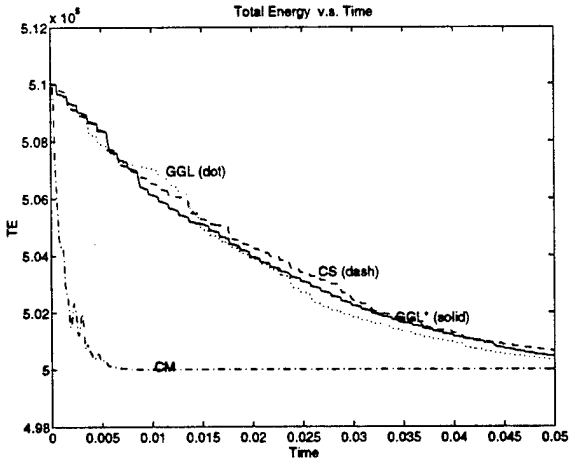

Figure 5.3: Total Energy Comparison of Oscillatory Spring Example; $\epsilon = 10^{-4}$ and $TOL = 10^{-4}$

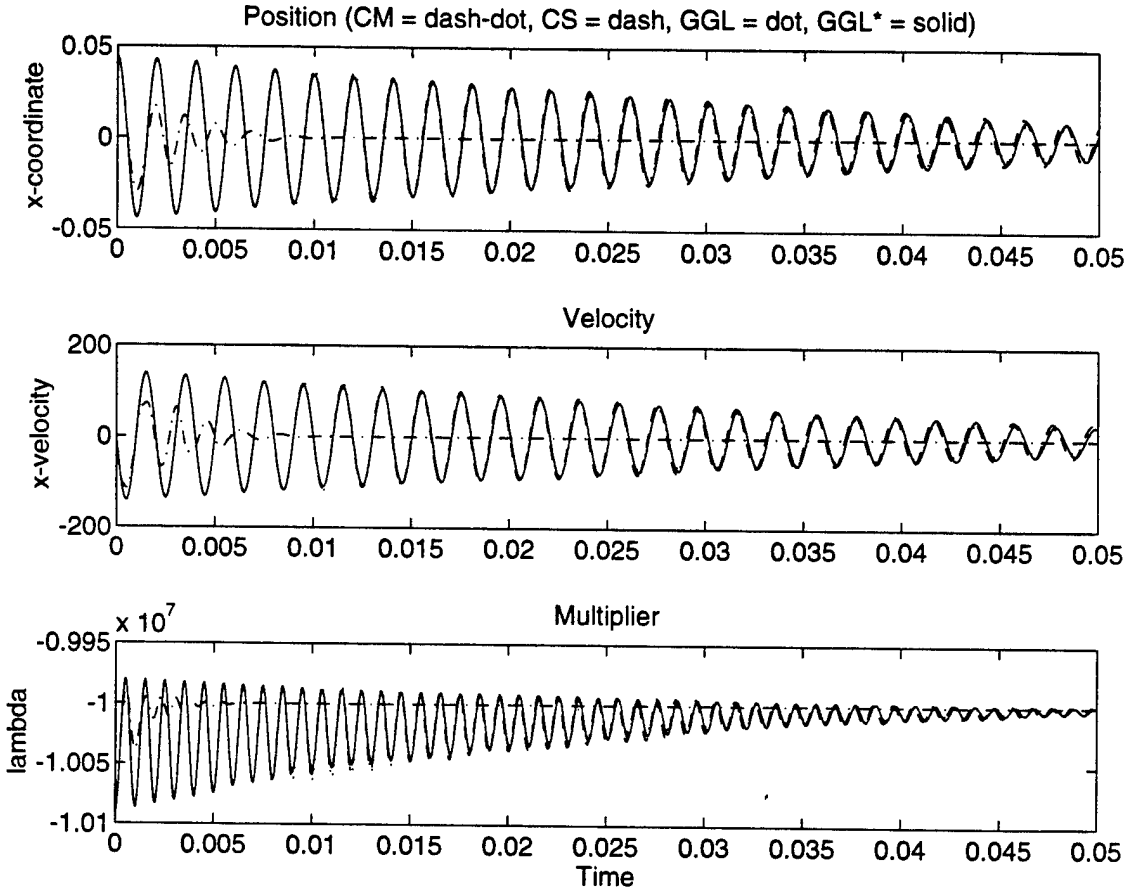

Figure 5.4: Results of Oscillatory Spring Example; $\epsilon = 10^{-4}$ and $TOL = 10^{-4}$

5.2 Two-body pendulum with bushing

The second example is a two-body pendulum in 2D Cartesian coordinates. Six generalized coordinates, $q = [x_1, y_1, \theta_1, x_2, y_2, \theta_2]^T$, locate the centers of mass and the orientation of the bodies. The first body is grounded, and the second body is constrained such that the distance between a point A of the first body and another point B of the second body is fixed, and its orientation is held constant. This leads to five constraint equations

$$g_1 = x_1 \tag{5.1a}$$
$$g_2 = y_1 \tag{5.1b}$$
$$g_3 = \theta_1 \tag{5.1c}$$
$$g_4 = d^{AB^T} d^{AB} - l^{AB} \tag{5.1d}$$
$$g_5 = \theta_2 - \theta \tag{5.1e}$$

where l^{AB} and θ are constant, and

$$d^{AB} = \begin{bmatrix} x_1 + a_1 \cos\theta_1 - a_2 \sin\theta_1 - (x_2 + b_1 \cos\theta_2 - b_2 \sin\theta_2) \\ y_1 + a_1 \sin\theta_1 + a_2 \cos\theta_1 - (y_2 + b_1 \sin\theta_2 + b_2 \cos\theta_2) \end{bmatrix}$$

such that $A = [a_1, a_2]$ and $B = [b_1, b_2]$ in the local reference coordinate systems of body 1 and body 2, respectively. In this example, we use $A = [0, 0]$, $B = [0, 0]$, $l^{AB} = 1$, and $\theta = 0$ for the constraint equations (5.1).

We apply a nonlinear oscillatory force formulated using *nonlinear beam* theory [4]. This type of force arises commonly in flexible multibody dynamics [23]. As described in (4.2), the deformation force between the i*th* and j*th* components is a function of the relative displacement of the reference frames X_i'-Y_i'-Z_i' and X_j'-Y_j'-Z_j', as shown schematically in Figure 5.5. Typically, the relative displacement is measured by

$$d_{ij} = r_j + A_j s_j' - r_i - A_i s_i' \tag{5.2}$$

where s_i' and s_j' are constant vectors to the origins of the force reference frames in their respective *local* coordinate systems, i.e., X_i-Y_i-Z_i and X_j-Y_j-Z_j, where r_i, r_j are the corresponding origins in a *global* coordinate system and A_i and A_j are the transformation matrices from the *global* to the *local* coordinate system [8]. The relative angles, $\Theta_{ij} = [\psi_{ij}, \theta_{ij}, \phi_{ij}]^T$, are calculated as

$$A_{ij} = (A_i B_i)^T A_j B_j$$
$$\psi_{ij} = -A_{ij}(2,3)$$
$$\theta_{ij} = A_{ij}(1,3)$$
$$\phi_{ij} = \arctan\left(\frac{A_{ij}(2,1)}{A_{ij}(2,2)}\right)$$

where $A_{ij}(k,l)$ is the component of the kth row and lth column of A_{ij}. The matrix A_{ij} is the relative orientation matrix of two force reference frames, i.e., B_i and B_j are constant. The relative velocity is the time derivative of the relative displacement $\dot{d}_{ij} = \frac{d}{dt} d_{ij}$, and the relative angular velocity is $\omega_{ij} = \omega_j - \omega_i$, where ω_i and ω_j are the angular velocities of bodies i and j respectively.

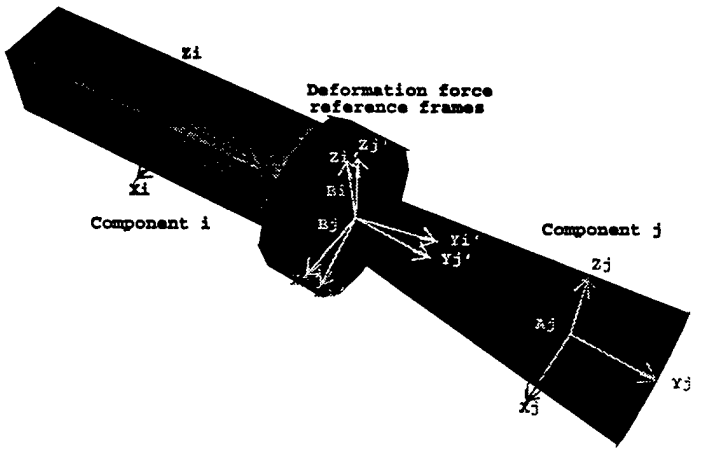

Figure 5.5: Deformation Force of a Flexible Body

Using the above defined notation, the force acting between the ith and jth components due to the deformation can be written as

$$f_{ij} = A_i B_i (K^f (A_i B_i)^T d_{ij} + C^f (A_i B_i)^T \dot{d}_{ij})$$

where K^f is a 3×3 structural stiffness matrix and C^f is the 3×3 damping coefficient matrix. Similarly, the torque acting between the components is

$$\tau_{ij} = A_i B_i (K^\tau \Theta_{ij} + C^\tau (A_i B_i)^T \omega_{ij})$$

where K^τ and C^τ are analogous to K^f and C^f. Note that the force and torque in this form are linear functions of the relative displacement (d_{ij}, Θ_{ij}) and the relative velocity $(\dot{d}_{ij}, \Omega_{ij})$.

Here, the 2D *bushing* force has stiffness matrix

$$K^f = \frac{1}{\epsilon} \begin{bmatrix} k^x & 0 & 0 \\ 0 & k^y & 0 \\ 0 & 0 & k^\theta \end{bmatrix}$$

where k^x, k^y, and k^θ are $O(1)$, and damping matrix

$$C^f = \begin{bmatrix} c^x & 0 & 0 \\ 0 & c^y & 0 \\ 0 & 0 & c^\theta \end{bmatrix}$$

where c^x, c^y, and c^θ are $O(1)$. For this 2D bushing example, (4.3) becomes

$$B(\theta) = \frac{1}{\epsilon}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & k^\theta \end{bmatrix} + |k^x - k^y| \begin{bmatrix} \cos^2\theta & \cos\theta\sin\theta & 0 \\ \cos\theta\sin\theta & \sin^2\theta & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The attachment points of the force device are $s_1' = [0.5, 0]$ and $s_2' = [-0.5, 0]$ in the body-fixed reference frames of bodies 1 and 2, respectively. The bushing force introduces oscillatory applied forces, causing small oscillations of the numerical solution, and yielding highly oscillatory multipliers in the index-2 DAE (1.3). The multipliers associated with the highly oscillatory components exhibit high-frequency oscillations with large amplitude. The standard convergence test of the Newton iteration depends heavily on these multipliers. Therefore, we modified the convergence test in DASSL to exclude the test for the multipliers. In addition, the multipliers are computed in the GGL^* by applying the pseudo-inverse $(GY)^{-T}Y^T$ to $r_1 = v^{(0)} - \rho_h q^{(0)}$ and to $r_2 = f(q^{(0)}, v^{(0)}, t) - M\rho_h v^{(0)}$, where $(q^{(0)}, v^{(0)})$ is the predictor in DASSL, and ρ_h is the discretization operator of BDF. The local error is estimated by the predictor-corrector difference of $(X^T q, X^T v)$ for CS, CM, and GGL^*, and of (q, v) for GGL.

Using the initial values q = [0,0,0,9.9989e-1,-1.4852e-2,0] and v = [0,0,0,-6.75e-5,-4.5444e-3,0], numerical results with $ATOL = RTOL = TOL$ are shown in Table 5.3, where $\epsilon = 10^{-3}$, $k^x = k^y = k^\theta = 1$, and $c^x = c^y = c^\theta = 10$. For this moderate stiffness $\epsilon = 10^{-3}$, all the methods perform well. Note that the constraint violation with these initial values is $O(10^{-3})$. This implies that the difference of the constraint reaction force at the initial value q_0 and that of a q^* on the constraint manifold is $\delta \approx \|q_0 - q^*\| = O(10^{-2})$, e.g., $\delta = O(\sqrt{10^{-3}})$. According to Theorem 4.2, the rate of convergence of the CS iteration is proportional to $\frac{\delta}{\epsilon}O(h)$. Increasing numbers of the convergence test failures in DASSL are expected as $\epsilon \to 0$. In this example, frequent convergence test failures occured when $\epsilon \leq 10^{-5}$ and $TOL \leq 10^{-3}$. We observe the same difficulties in the Newton convergence of the GGL and GGL^* iterations. On the other hand, the CM iteration with its better Newton convergence, as explained by Theorem 4.2, is able to take much larger time steps and the nonlinear oscillation is damped effectively. In Table 5.4, the results of $\epsilon = 10^{-6}$, $k^x = k^y = k^\theta = 1$, and $c^x = c^y = c^\theta = 10$ are shown. In Figure 5.6, we plot the stepsize taken by DASSL for GGL, GGL^*, CS, and CM using the stiffness coefficient $\epsilon = 10^{-5}$.

Method	TOL	*no. steps*	*no. fevals.*	*no. jevals.*	*no. etfs.*	*no. ctfs.*
GGL	10^{-3}	59	141	46	0	12
*GGL**	10^{-3}	62	154	48	0	13
CS	10^{-3}	62	156	48	0	13
CM	10^{-3}	62	156	48	0	13
GGL	10^{-4}	61	136	27	1	5
*GGL**	10^{-4}	77	193	65	1	16
CS	10^{-4}	77	193	65	1	16
CM	10^{-4}	77	193	65	1	16
GGL	10^{-5}	108	259	77	1	21
*GGL**	10^{-5}	87	215	54	0	13
CS	10^{-5}	87	215	54	0	13
CM	10^{-5}	87	215	54	0	13

Table 5.3: Results of Two-Body Pendulum with a Bushing Force, $\epsilon = 10^{-3}$, $c^x = c^y = c^\theta = 10$

Method	TOL	*no. steps*	*no. fevals.*	*no. jevals.*	*no. etfs.*	*no. ctfs.*
GGL	10^{-4}	5267	10536	7899	0	2633
*GGL**	10^{-4}	2251	4900	3360	1	1120
CS	10^{-4}	2252	4901	3361	1	1120
CM	10^{-4}	20	40	7	0	0

Table 5.4: Results of Bushing Problem, $\epsilon = 10^{-6}$, $c^x = c^y = c^\theta = 10$

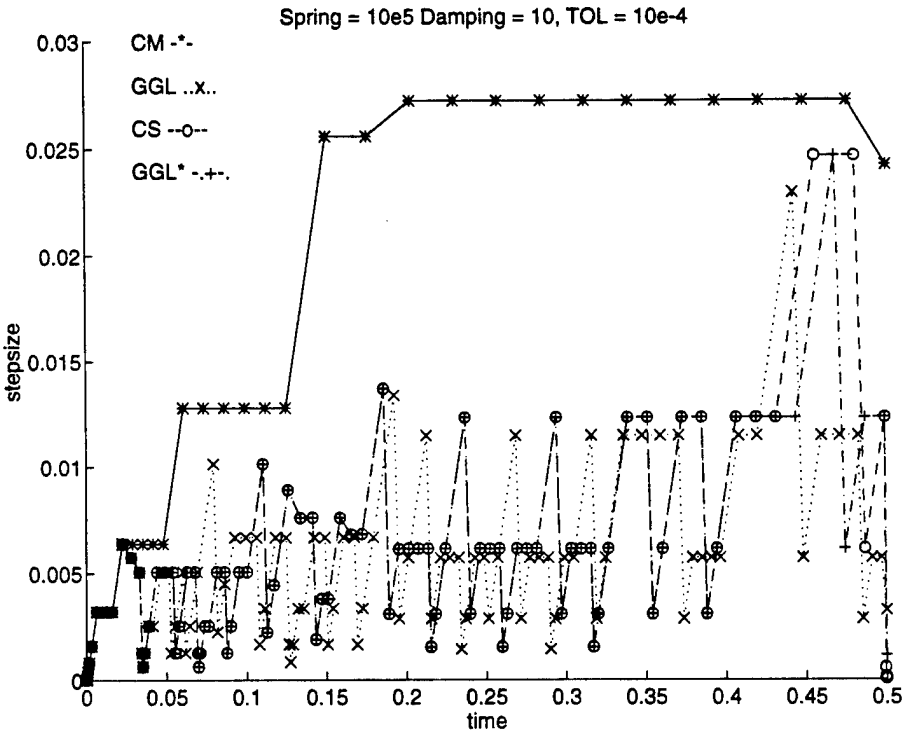

Figure 5.6: Time Steps Used in Solving the Bushing Problem, $\epsilon = 10^{-5}$, $c^x = c^y = c^\theta = 10$

References

[1] U. Ascher and L. R. Petzold, *Projected implicit Runge-Kutta methods for differential-algebraic equations*, SIAM J. Numer. Anal., 28 (1991), pp. 1097-1120.

[2] N. N. Bogoliubov and Y. A. Mitropolski, *Asymptotic Methods in the Theory of Nonlinear Oscillations*, Hindustan Publishing Corp., Delhi, India, 1961.

[3] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, second edition, SIAM, 1995.

[4] R. R. Craig, *Structural Dynamics*, John Wiley & Sons, New York, 1981.

[5] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[6] C.W. Gear, G.K. Gupta and B.J. Leimkuhler, *Automatic integration of the Euler-Lagrange equations with constraints*, J. Comp. Appl. Math., 12 (1985), pp. 77-90.

[7] E. Griepentrog, *Index reduction methods for differential-algebraic equations*, Seminarberichte Nr. 92-1, Humboldt-Universität zu Berlin, Fachbereich Mathematik, 1992.

[8] H. Goldstein, *Classical Mechanics, 2nd ed.*, Addison-Wesley, Inc., Reading, Mass., 1980.

[9] G. H. Golub and V. Pereyra, *The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate*, SIAM J. Numer. Anal., 10 (1973), pp. 413-432.

[10] G. H. Golub and C. F. Van Loan, *Matrix Computations, second edition*, Johns Hopkins University Press, 1989.

[11] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential- Algebraic Problems*, Springer-Verlag, Berlin, 1991.

[12] E. J. Haug, *Computer Aided Kinematics and Dynamics of Mechanical Systems Volume I: Basic Methods*, Allyn-Bacon, 1989.

[13] J. Kevorkian and J.D. Cole, *Perturbation Methods in Applied Mathematics*, Springer-Verlag, New York, 1981.

[14] C. Lanczos, *The Variational Principles of Mechanics*, University of Toronto Press, 1949.

[15] Ch. Lubich, *Integration of stiff mechanical systems by Runge-Kutta methods*, ZAMP, 44 (1991), pp. 1022-1053.

[16] F. Potra and W.C. Rheinboldt, *On the numerical solution of the Euler-Lagrange equations*, J. Mech. Struct. Mach., 19 (1991), pp. 1-18.

[17] P.J. Rabier and W.C. Rheinboldt, *On the numerical solution of the Euler-Lagrange equations*, Technical Report ICMA-93-117, Univ. of Pittsburgh, PA, 1993.

[18] S. Reich, *On a geometric interpretation of differential-algebraic equations*, Circ. Syst. Sign. Proc., 9 (1990), pp. 367-382.

[19] S. Reich, *A free energy approach to the torsion dynamics of macromolecules*, Preprints SC 95-17, Konrad-Zuse-Zentrum für informationstechnik Berlin, 1995.

[20] R.A. Wehage and E. J. Haug, *Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems*, ASME J. Mech. Design, 134 (1982), pp. 247-255.

[21] J. Yen, *Constrained equations of motion in multibody dynamics as ODEs on manifolds*, SIAM J. Numer. Anal., 30 (1993), pp. 553-568.

[22] J. Yen and L.R. Petzold, *Numerical solution of nonlinear oscillatory multibody dynamic systems*, Proc. of 16th Numer. Anal. Conf., University of Dundee, to appear.

[23] W.S. Yoo and E.J. Haug, *Dynamics of articulated structures, part I theory*, J. Mech. Struct. Mach., 14 (1986), pp. 105-126.