A study of alternative formulations for optimization of structural and mechanical systems subjected to statics and dynamic loads

Qian Wang
University of Iowa

Follow this and additional works at: https://ir.uiowa.edu/etd
Part of the Civil and Environmental Engineering Commons

A STUDY OF ALTERNATIVE FORMULATIONS FOR OPTIMIZATION OF STRUCTURAL AND MECHANICAL SYSTEMS SUBJECTED TO STATIC AND DYNAMIC LOADS

by

Qian Wang

An Abstract

Of a thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy degree
in Civil and Environmental Engineering in
the Graduate College of
The University of Iowa

December 2006

Thesis Supervisor: Professor Jasbir S. Arora

ABSTRACT

The concept of simultaneous analysis and design (SAND) is revisited with two objectives in mind: (i) to propose and evaluate alternative formulations for various structural and mechanical system optimization problems, and (ii) to study implementation aspects of the formulations. SAND formulates the optimization problem in a mixed space of design variables and behavior variables, to imbed the state equations in one single optimization framework. Therefore explicit analysis and design sensitivity analysis of the system are not needed.

Several alternative formulations for structural design optimization based on the SAND concept are defined using displacements, and forces or stresses as optimization variables. As sample application areas, optimal design of trusses and frames are considered. Existing analysis software is integrated with an optimizer to solve example problems. Only the pre- and post-processing capabilities of the analysis software are needed to evaluate the problem functions. In addition, at least one of the alternative formulations does not even require assembly of the global stiffness matrix for the structure.

Alternative formulations for transient dynamic response optimization and digital human motion simulation are also presented, analyzed and evaluated. Similar to the SAND approach used for optimization of structures subjected to static loads, the equations of motion are not integrated explicitly; they can be imposed as equality constraints in these formulations.

For the alternative formulations, the optimization problem is quite large in terms of the numbers of variables and constraints. However, the problem functions are quite sparse, which is exploited in the optimization process. Performance of various formulations is evaluated with extensive numerical experiments and their advantages and disadvantages are discussed. It is concluded that the alternative SAND-type formulations are more efficient compared to the conventional approach where gradients of implicit functions must be evaluated. In addition, they offer flexibility and ease of numerical implementation because linear systems of equations are not solved for analysis or design sensitivity analysis.

The alternative formulations represent a fundamental shift in the way analysis and design optimization are currently treated. This shift in paradigm needs to be further nurtured and developed for optimization of more complex multidisciplinary systems.

Abstract Approved: ________________________________
Thesis Supervisor

____
Title and Department

____
Date

A STUDY OF ALTERNATIVE FORMULATIONS FOR OPTIMIZATION OF STRUCTURAL AND MECHANICAL SYSTEMS SUBJECTED TO STATIC AND DYNAMIC LOADS

by

Qian Wang

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy degree
in Civil and Environmental Engineering in
the Graduate College of
The University of Iowa

December 2006

Thesis Supervisor: Professor Jasbir S. Arora

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

____

PH.D. THESIS

____

This is to certify that the Ph.D. thesis of

Qian Wang

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Civil and Environmental Engineering at the December 2006 graduation.

Thesis Committee: _________________________
Jasbir S. Arora, Thesis Supervisor

____
Karim Abdel-Malek

____
Asghar Bhatti

____
Jia Lu

____
Colby C. Swan

Moving Forward
- To My Family

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my dissertation advisor, Professor Jasbir S. Arora for his valuable advice, support and encouragement throughout this research and my years as a Ph.D. student at Iowa. I deem it as an honor and pleasure to be his student and work with him in the area of engineering optimization. Professor Arora acts as a true mentor assisting my professional development, and my development as a person.

Grateful thanks are also extended to Professor Karim Abdel-Malek, Professor Asghar Bhatti, Professor Jia Lu, and Professor Colby C. Swan for not only being the members of my thesis committee but also offering constructive advice and helpful guidance as well.

I wish to express my gratitude to all my colleagues and friends for their support, encouragement, friendship and insightful suggestions.

I would like to acknowledge the support of my parents and sister. I am deeply grateful to my wife, Yi Ding, for her encouragement, understanding as well as her love. They are the sources of power for me to move forward, whatever I do, wherever I am.

Support for this research and related conference presentations provided by the following resources, is gratefully acknowledged:

- University of Iowa Carver Research Initiation Grant
- Virtual Soldier Research (VSR) Program
- University of Iowa Civil and Environmental Engineering Teaching Assistantship

- University of Iowa Graduate Student Senate (GSS) Travel Funds
- University of Iowa Student Government (UISG) Scholarly Presentations
- 2005 Chinese Government Award for Outstanding Self-financed Students Abroad

ABSTRACT

The concept of simultaneous analysis and design (SAND) is revisited with two objectives in mind: (i) to propose and evaluate alternative formulations for various structural and mechanical system optimization problems, and (ii) to study implementation aspects of the formulations. SAND formulates the optimization problem in a mixed space of design variables and behavior variables, to imbed the state equations in one single optimization framework. Therefore explicit analysis and design sensitivity analysis of the system are not needed.

Several alternative formulations for structural design optimization based on the SAND concept are defined using displacements, and forces or stresses as optimization variables. As sample application areas, optimal design of trusses and frames are considered. Existing analysis software is integrated with an optimizer to solve example problems. Only the pre- and post-processing capabilities of the analysis software are needed to evaluate the problem functions. In addition, at least one of the alternative formulations does not even require assembly of the global stiffness matrix for the structure.

Alternative formulations for transient dynamic response optimization and digital human motion simulation are also presented, analyzed and evaluated. Similar to the SAND approach used for optimization of structures subjected to static loads, the equations of motion are not integrated explicitly; they can be imposed as equality constraints in these formulations.

For the alternative formulations, the optimization problem is quite large in terms of the numbers of variables and constraints. However, the problem functions are quite sparse, which is exploited in the optimization process. Performance of various formulations is evaluated with extensive numerical experiments and their advantages and disadvantages are discussed. It is concluded that the alternative SAND-type formulations are more efficient compared to the conventional approach where gradients of implicit functions must be evaluated. In addition, they offer flexibility and ease of numerical implementation because linear systems of equations are not solved for analysis or design sensitivity analysis.

The alternative formulations represent a fundamental shift in the way analysis and design optimization are currently treated. This shift in paradigm needs to be further nurtured and developed for optimization of more complex multidisciplinary systems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Latin letters

A	cross-sectional area
A_i	cross-sectional area of the *i*th structural member
A_i^U, A_i^L	upper and lower bounds for cross-sectional area of a structural member *i*
$\mathbf{A}$	a vector of cross-sectional areas of a structure
$\mathbf{A}$	a recursive position transformation matrix
$\mathbf{b}_i^{FQ}$	a transformation vector between axial force and nodal forces for truss member *i*
$\mathbf{b}_i^{QF}$	a transformation vector between nodal forces and axial force for truss member *i*
$\mathbf{B}$	a recursive velocity transformation matrix
$\mathbf{B}_i$	a vector linking stress and global nodal displacements for truss member *i*
$\mathbf{B}_i^a$	a matrix linking forces and global nodal displacements for frame member *i*
$\mathbf{B}_i^b$	a matrix linking forces and global nodal displacements for frame member *i*
c_0	a function
$\mathbf{c}$	a vector of functions
$\mathbf{c}_k^i$	coefficient vectors of piecewise Hermite interpolation
$\mathbf{C}$	generalized damping matrix
$\mathbf{C}$	a recursive acceleration transformation matrix
d	number of degrees of freedom per structural member

d	dimension of vectors $\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}$
d	number of degree of freedoms in a human model
$\mathbf{d}$	a vector of behavior/state variables
$\mathbf{D}$	a recursive transformation matrix
e	number of constraints in $\mathbf{g}$
$\mathbf{e}_j$	a 6×1 Boolean vector
E	modulus of elasticity
$\mathbf{E}$	a recursive transformation matrix
f	objective function
$\mathbf{f}$	first derivatives of state variables in state space representation
F_i	internal force of member *i*
$\mathbf{F}$	a vector of internal forces of a structure
$\mathbf{F}(\mathbf{x}, t)$	a generalized force vector
$\overline{\mathbf{F}}(\mathbf{x}, t)$	a system matrix in state space representation
g	an inequality constraint
g_i^σ	the *i*th member stress inequality constraint
g_j	the *j*th inequality constraint
$\mathbf{g}$	a vector of inequality constraints
$\mathbf{g}^\sigma$	a vector of member stress inequality constraints
h_0	a time-dependent function

h_i^e	the *i*th member equilibrium equality constraint
h_j	the *j*th equality constraint
h_j^g	the *j*th global equilibrium equality constraint
$\mathbf{h}$	a vector of equality constraints
$\mathbf{h}$	a vector of time-dependent functions
$\mathbf{h}^e$	a vector of member equilibrium equality constraints
$\mathbf{h}^g$	a vector of global equilibrium equality constraints
$\mathbf{H}$	Hessian of the Lagrangian
i	index of members
I	moment of inertia
I_i	moment of inertia of member *i*
$\mathbf{I}$	identity matrix
j	index of constraints
J	a cost function or performance index
k	index of members
$\mathbf{k}_i$	stiffness matrix of member *i* in the global coordinate system
$\mathbf{k}_i'$	stiffness matrix of member *i* in the local coordinate system
$\bar{\mathbf{k}}_i$	stiffness matrix of truss member *i* independent of design variables
$\mathbf{k}_i^a$	stiffness matrix of frame member *i* independent of design variables
$\mathbf{k}_i^b$	stiffness matrix of frame member *i* independent of design variables

$\mathbf{K}$	global stiffness matrix in the global coordinate system
$\overline{\mathbf{K}}$	a system matrix in state space representation
$\mathbf{l}_i$	a 6×1 transformation vector
L	number of loading conditions
L_i	the length of the *i*th structural member
m	total number of degrees of freedom in a structure
m	number of elements in the design variable vector
m_i	mass of the *i*th link
M_i	end bending moment of structural member *i*
$\mathbf{M}$	generalized mass matrix
n	total number of members in a structure
n	number of control points in a cubic B-spline
N	number of time intervals
N_g	number of foot ground penetration constraints in a gait simulation
$N_{j,4}(t)$	B-spline basis function for the *j*th control point
N_s	number of foot strike position constraints in a gait simulation
N_Z	number of ZMP stability constraints in a gait simulation
NE	number of members connected to the same node
$\mathbf{p}_i$	a 6×1 transformation vector
P_i	axial force of structural member *i*
$\mathbf{P}$	a vector of control points

$q_j, \dot{q}_j, \ddot{q}_j$	displacement, velocity and acceleration of the *j*th joint angle
$\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$	joint angle displacement, velocity and acceleration vectors
$\mathbf{q}^U, \mathbf{q}^L$	upper and lower bounds of a joint angle displacement vector
$\mathbf{q}_i$	nodal displacement vector for member *i* in the global coordinate system
Q_{jk}	nodal force of member *k* along the *j*th global displacement
$\mathbf{Q}$	a vector of member-end forces for all structural members
$\mathbf{Q}_i$	member-end force vector for member *i* in the global coordinate system
r_j	the *j*th global displacement in a structure
r_j^U, r_j^L	upper and lower bounds for *j*th global displacement in a structure
$\mathbf{r}$	a nodal displacement vector in the global coordinate system
$\mathbf{r}^U, \mathbf{r}^L$	upper and lower bounds of a nodal displacement vector
${}^0\mathbf{r}_d, \mathbf{r}_d$	augmented 4×1 vectors defining global and local Cartesian positions
R_j	resultant external load acting at a node in the direction of displacement *j*
$\mathbf{R}$	external load vector in the global coordinate system
S	section modulus
S_i	section modulus of a structural member *i*
t	time
$t_{c,i}$	center of a time interval
t_i	time grid point *i*
t_i	height of the cross-section for a structural member *i*

T	the total time of a motion
$\mathbf{T}$	a transformation matrix between the local and global coordinate systems
u	a local time parameter in each time interval
$\mathbf{x}$	a vector of design or optimization variables
$\mathbf{x}^U, \mathbf{x}^L$	upper and lower bounds of a vector of variables
$\mathbf{X}(\mathbf{q})$	a position vector of a point in the global Cartesian space
$\mathbf{X}_d$	a position vector in the local Cartesian space
y_{ZMP}	y coordinate of ZMP
$\mathbf{y}, \dot{\mathbf{y}}$	generalized displacement and velocity vectors in state space representation
z_{ZMP}	z coordinate of ZMP
$\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}$	generalized displacement, velocity and acceleration vectors
$\mathbf{Z}_i$	a $6\times m$ Boolean matrix

Greek letters

α	a constant in relationships among cross-sectional properties
β	a constant in relationships among cross-sectional properties
β	a constant in Newmark's method
γ	a constant in relationships among cross-sectional properties
γ	a constant in Newmark's method
Δt	time interval
ζ	a vector of defect equations
η	one component in $\mathbf{b}_i^{QF}$, representing a direction cosine

$\lambda_x, \lambda_y, \lambda_z$	direction cosines with respect to the global *x*, *y* and *z* directions
σ_i	stress of the *i*th structural member
σ_i^U, σ_i^L	upper and lower bounds of stress of the *i*th structural member
$\boldsymbol{\sigma}$	a vector of stresses of a structure
$\boldsymbol{\sigma}^U, \boldsymbol{\sigma}^L$	upper and lower bounds of a vector of stresses
$\boldsymbol{\tau}$	a vector of joint torques
$\boldsymbol{\tau}^U, \boldsymbol{\tau}^L$	upper and lower bounds of a vector of joint torques
ϕ	a constant in relationships among cross-sectional properties

LIST OF ABBREVIATIONS

AAO	all-at-once
AISC	American Institute of Steel Construction
CPU	central processing unit
DAE	differential-algebraic equation
DE	differential equation
DBO	displacement-based optimization
DH	Denavit-Hartenberg
DOF	degree of freedom
EOM	equation of motion
FEM	finite element method
FSR	foot support region
GA	genetic algorithm
IP	infeasible path
MDO	multidisciplinary design optimization
MPEC	mathematical programs with equilibrium constraints
NAND	nested analysis and design
NLP	nonlinear programming
OC	optimality criteria
OCP	optimal control problem
PC	personal computer
PDE	partial differential equation

QP	quadratic programming
RAM	random access memory
SAND	simultaneous analysis and design
SLP	sequential linear programming
SQP	sequential quadratic programming
ZMP	zero moment point

CHAPTER 1
INTRODUCTION

1.1 Introduction and Motivation

Since 1960s, various formulations for optimization of problems in many diverse fields, such as structural, chemical, industrial and mechanical engineering, economics, optimal control and others have been developed and discussed in the literature. These formulations have been reviewed with the objective of possible cross fertilization of ideas that can lead to better approaches for optimization of complex systems (Arora and Wang 2005).

The most common approach for structural optimization and optimal control problems has been the one where only the design variables are treated as optimization variables. This is the conventional formulation (also called nested analysis and design, NAND) where only the design variables are treated as the independent optimization variables. All other response quantities, such as displacements, stresses, and internal forces in a structure are treated as implicit functions of the design variables. These response quantities are also called *dependent variables*. Therefore, in the optimization process, analysis is performed to obtain the response variables and to evaluate functions, and hence a nested process of analysis-optimization is set up naturally for design synthesis. However, this optimization formulation method is difficult to use for design of practical structural and mechanical systems. There are three main reasons for this difficulty:

1. Many practical applications are complex requiring interaction between several

disciplines; i.e., require use of different analysis software components that are discipline-specific. Since they are independent programs, they are impossible to integrate into the current design optimization environment.

2. The current methods of optimization require design sensitivity analysis which can be implemented into analysis software but only with substantial effort and resources. Any addition for new capability to the analysis software requires updating of the design sensitivity analysis part as well. Thus it is difficult to take advantage of advances in the analysis methods.
3. Many practical problems are nonlinear and involve transient dynamic loads making it even more difficult to implement and use the existing optimization formulations and methods.

The main difficulty with the existing formulations noted in (2) is that the response quantities, called the *behavior/state variables*, are treated as implicit functions of the design variables. Derivatives of state variables with respect to the design variables are needed in the conventional approaches for design optimization. This is known as *design sensitivity analysis*. Although a considerable body of literature is available on design sensitivity analysis of different classes of problems (Haug *et al*. 1986; Arora 1995), its implementation into analysis software is quite difficult, especially for multidisciplinary problems requiring use of different discipline-specific analysis software.

Note also that the conventional formulation requires iterations where design changes are made and the structure is re-analyzed for its response. An evaluation of any design change requires simulation of the system, the process can be quite tedious and time-consuming. To alleviate some of the difficulties noted above, several different research avenues have been explored in the literature. First, efficient structural reanalysis methods for analyzing a modified structure have been developed (Kirsch 2000; Kirsch and Papalambros 2001; Kirsch 2002). These methods can be useful for efficient analysis of updated designs and for calculation of the design derivatives during the optimization process. Second, various methods to develop approximate models, the so-called meta-models, such as the response surface approximations, have been proposed and evaluated for optimization of complex structural and mechanical systems (Myers and Montgomery 2002; Krishnamurthy 2003). Third, some alternative formulations have also been proposed and evaluated for optimization of structural and mechanical systems. Optimization methods have been developed such that there is no need for analysis and design sensitivity analysis to optimize systems. This way the algorithms for analysis of systems and algorithms for design optimization of systems are somewhat uncoupled. One such formulation is called simultaneous analysis and design (SAND) formulation. By formulating the optimization problem in a mixed space of design variables and behavior/state variables, the analysis equations are imbedded as equality constraints in one single optimization problem, therefore no explicit analysis is needed.

1.2 Objectives of Research

The major objectives of this research are listed as follows:

1. To propose and develop different alternative formulations for structural and mechanical systems subjected to static and dynamic loads.
2. To develop computer-based automatic optimization programs for the optimization of different structural and mechanical systems. The sparse structure of the alternative formulations will be used for numerical

implementation. Existing programs will be used and their role in the alternative formulations will be evaluated.

3. To evaluate the proposed alternative formulations and compare their performance with the conventional formulation by solving various example problems.

In this research, the SAND idea is utilized and various alternative formulations are developed and evaluated for optimization of different structural and transient dynamic response optimization problems. Various behavior variables, such as nodal displacements, forces and stresses in a structure, and generalized displacements, velocities and accelerations in a dynamic system are treated as independent variables in the optimization process. Therefore the system governing equations are treated as equality constraints in the optimization formulation. The optimization constraints can be expressed explicitly in terms of the optimization variables. The resulting optimization problems are solved using powerful nonlinear programming (NLP) methods. Numerical examples are needed to explore the applicability of this proposed optimal design procedure, and their solutions are compared to those available in the literature. The advantages and disadvantages of different optimization formulations are also discussed.

The challenges and difficulties include the integration of existing analysis software with explicit optimization formulations, and the numerical implementation of alternative formulations for large-scale optimization problems, such as the treatment of sparse matrices and various variables with different orders of magnitude. Even if design variable linking technique can be applied, the size of the optimization problem, i.e., the numbers of variables and constraints, can still be very large. However, this research will shed light on the optimization of other optimization problems and potential for practical applications of the alternative formulations.

1.3 Overview of the Dissertation

In Chapter 2, alternative formulations for optimization and simulation of structural and mechanical systems, and other related fields are reviewed. Mathematical programs with equilibrium constraints (MPECs), partial differential equations (PDE) – constrained optimization, and optimal control are all surveyed. The conventional formulation and a general framework for alternative formulations for structural and mechanical system optimization, and optimal control problems are described in Chapter 3. Advantages and disadvantages of these formulations are described. Three alternative formulations for sizing optimization of elastic trusses are studied and evaluated in Chapter 4. Several numerical examples from the literature are solved and their solutions are compared. In Chapter 5, optimal design of framed structures is considered. Two alternative formulations are proposed and several example problems are solved and solutions compared with those available in the literature. In Chapter 6, some large-scale structural design examples and their numerical results are presented. The sparsity features of different formulations are discussed and a powerful sparse SQP solver is used as an optimizer. In Chapters 7 and 8, alternative formulations for transient dynamic response optimization are proposed and evaluated. These formulations are based on different discretization techniques of first and second order differential equations (DEs). Digital human dynamic motion prediction problems are considered in Chapter 9. These are essentially optimal control problems that are solved by numerical optimization techniques. Various alternative formulations are developed and discussed. A lower body gait model is used as an example. Chapter 10 includes discussion, conclusions, and some future research topics.

CHAPTER 2
REVIEW OF LITERATURE

2.1 Introduction

In the structural optimization literature, three basically different formulations for optimum design have been presented. The first one is called the *conventional formulation* where only the structural design variables are treated as the optimization variables. This is also called the *nested analysis and design* (NAND) approach. The second set of formulations is known as the *simultaneous analysis and design* (SAND) approach. In these formulations, some of the state variables, such as the displacements, are also treated as optimization variables in addition to the traditional design variables. The governing equilibrium equations are treated as equality constraints. The third formulation is known as the displacement based two-phase approach where the displacements are treated as optimization variables in the outer loop and the design variables as the unknowns in the inner loop.

Parallel developments of SAND-type optimization formulations and their solutions strategies have also taken place in other fields since 1970s. A general class of formulations known as *mathematical programs with equilibrium constraints*, or in short MPECs, has been developed and studied. The word “equilibrium” in MPEC refers to the variational equalities or inequalities that model the equilibrium phenomenon in engineering and other applications. Another class of formulations that has been presented and analyzed recently is known as the *partial differential equations* (PDE)-constrained optimization. In these formulations, the equilibrium equations are expressed in a continuum form, the PDEs. In addition to these literatures, SAND-type approaches have been used to solve optimal control and multidisciplinary design optimization (MDO) problems. We shall present an overview of these literatures.

A thorough review of various formulations for optimization of structural and mechanical systems has been presented by in Kirsch and Rozvany (1994), and Wang and Arora (2005). In the paper by Kirsch and Rozvany (1994), several alternative but equivalent formulations for structural optimization problems were presented based on different independent variables, analysis methods and forms of resulting constraints. These included design variable space (conventional), SAND, optimality criteria (OC), and some simplified SAND formulations. Details of the formulations were discussed for truss-type structures. A more recent review by Arora and Wang (2005) covers various SAND formulations for sizing, shape, topology and multidisciplinary applications as well as displacement-based formulations, MPEC, PDE-constrained optimization, optimal control, as well as MDO problems. Objective of this chapter is to review the current literature on alternative formulations for optimization of structural and mechanical systems. These formulations may offer some clues on how best to formulate the problem with the consideration of practical applications. Some of the stochastic methods that do not require gradients in their calculations are not considered here for presentation.

2.2 Simultaneous Analysis and Design (SAND)

If all design variables and response variables are combined together in a single optimization problem, such that no explicit analysis is needed, then the approach is usually called the simultaneous analysis and design (SAND) formulation. Some of the earliest attempts to include state variables in the structural optimization problem were by

Schmit and Fox (1965) and Fuchs (1982, 1983). Explicit expressions for the objective function and the constraints could be obtained. A SAND formulation based on an element-by-element preconditioned conjugate gradient technique was proposed by Haftka (1985), and Haftka and Kamat (1989). Shin *et al.* (1988) considered the simultaneous analysis and design approach to solve the problem with eigenvalue constraints. Ringertz (1989, 1992, 1995) presented SAND methods for the optimal design of nonlinear structures. The SAND formulations usually include large numbers of variables and constraints. However, the matrix sparsity in the constraint Jacobian can be exploited and utilized for numerical efficiency (Orozco and Ghattas 1992a, Ringertz 1995). Orozco and Ghattas (1997) also developed a reduced SQP method to optimize geometrically nonlinear truss structures.

In recent years, various SAND formulations have been successfully applied to the configuration and topology design of structures (Bendsøe and Sigmund 2003). It is well-known that a crucial step for success of the SAND formulations is the solution of very large scale optimization problems. Therefore considerable focus has been put on the development of new algorithms to solve large-scale optimization problems (Ben-Tal and Bendsøe 1993; Ringertz 1995; Ben-Tal and Roth 1996; Ben-Tal and Nemirovski 1997; Orozco and Ghattas 1997; Jarre *et al.* 1998; Maar and Schulz 2000; Ben-Tal *et al.* 2000; Herskovits *et al.* 2001; Hoppe *et al.* 2002; and others).

2.3 Displacement-Based Optimization (DBO)

Another alternative approach of optimum structural design is the so-called displacement based two-phase procedure. The design variables and response variables can be combined together to form in a multi-phase optimization problem, such that no analysis is needed. In a paper by Missoum and Gürdal (2002), the two-phase optimization procedure of McKeown (1977, 1989, 1998) was presented and applied to optimize trusses. The formulation solved the problem in two phases, the inner and outer problems. In the inner problem, the cost was minimized subject to satisfaction of the equilibrium equations. The displacement field was specified and the design variables were the independent variables. In the outer problem, the displacements were determined to minimize the cost function subject to the stress and displacement constraints. That work has also been extended to nonlinear problems (Gu *et al.* 2002; Missoum *et al.* 2002a,b). Limitations of the method are that the cost function is an implicit function of the displacements in the outer problem, and some times it is nondifferentiable.

2.4 Mathematical Programs with Equilibrium Constraints (MPEC)

It turns out that the SAND-type formulations have also been discussed in other fields since 1970s. These are known as mathematical programs with equilibrium constraints, or in short MPECs. An MPEC is an optimization problem having primary constraints that are expressed as a parametric variational inequality or a complementarity system. The MPECs can also be viewed as a generalization of the so-called bilevel programs, also known as mathematical programs with optimization constraints. The basic idea of MPEC was introduced in the operations research literature in the early 1970s by Bracken and McGill (1973). These ideas can also be traced back the economic problem of Stackelberg game (Stackelberg 1952). The MPEC has evolved as a major research field in recent years and has been put on a firm mathematical foundation (Lou *et al.* 1996; Outrata *et al.* 1998). The MPEC formulation covers many diverse applications, such as economics, chemical engineering, and many more. As a particular example, structural analysis and design problems in unilateral frictional contact have been discussed with the MPEC formulation (Hilding *et al.* 1999).

2.5 Partial Differential Equations (PDE) – Constrained Optimization

Other developments of optimization formulations and their solutions strategies have also taken place recently. These are known as PDE-constrained optimization problems (Hoppe *et al.* 2002; Biegler *et al.* 2003; Schulz 2004). Most simulation problems in engineering fields involve solutions of partial differential equations. Therefore, following the SAND concept, the simulation variables can also be treated as optimization variables and the PDEs as equality constraints. Many times the PDEs are obtained as a result of some variational principle to model an equilibrium phenomenon. Therefore, PDE-constrained optimization can be viewed as a special case of the MPEC.

2.6 Optimal Control

Several viable approaches have been used to solve open-loop numerical optimal control problems (Betts 1998; Hull 2003). These problems involve the integration of differential algebraic equations (DAEs), or just differential equations (DEs). If the design variables together with the state variables and control variables are all treated as optimization variables, the approach is called the *direct collocation/transcription* method (Hargraves and Paris 1987). If the control variables are eliminated from the system (i.e., only the design variables and the state variables are treated as optimization variables), it is called the *differential inclusion* method (Seywald 1994). If the state variables are eliminated, it is called *state variable elimination* method (Hull 2003). Another possibility is the so-called *multiple shooting* technique (Betts and Huffman 1991; Leineweber *et al.* 2003).

The basic idea of the SAND-type approach (direct collocation) is to discretize the system of first order differential equations, and define a finite dimensional approximations or parametric representation for the state and control variables. The discretized state equations are treated as equality constraints in the optimization process, converting the optimal control problem into an NLP problem, which is solved numerically. This has been used to solve open-loop optimal control problems for trajectory design in aerospace engineering (Enright and Conway 1991; Betts 2000), chemical process engineering (Cuthrell and Biegler 1986; Biegler 1998), and robotics or human motion planning (Kaplan and Heegaard 2002; Bottasso and Croce 2004).

2.7 Multidisciplinary Design Optimization (MDO)

Other applications of SAND can be found in heat transfer (Hrymak *et al*. 1985) and aerodynamics (Shubin 1995; Frank and Shubin 1992; Orozco and Ghattas 1992b, 1996). There is also a growing interest in its use to formulate multidisciplinary design optimization (MDO) (Haftka *et al*. 1992; Cramer *et al*. 1994; Shubin 1995; Balling and Sobieszczanski-Sobieski 1996; Balling and Wilkinson 1997). The formulation has also been called the all-at-once (AAO) or infeasible path (IP) approach in the literature (Cramer *et al*. 1994; Shubin 1995; Frank and Shubin 1992; Orozco and Ghattas 1996).

2.8 Discussion

Alternative formulations for optimization of structural and mechanical systems, and other fields are surveyed. These include simultaneous analysis and design (SAND), displacement based two-phase approach, mathematical programs with equilibrium constraints (MPEC), and partial differential equations (PDE)-constrained formulations. In addition to the foregoing literature, SAND-type formulations for the optimal control problem and multidisciplinary optimization problems are briefly reviewed. Based on this review of the literature, the current status and future opportunities for research on alternative formulations for optimization of structural and mechanical systems are as follows:

1. Most of the formulations in the structural optimization literature have been discussed for truss structures; they need to be extended to other complex structures.
2. Most of the formulations have focused on use of the displacement based FEM. Other analysis methods need to be considered, such as the force methods, mixed methods, meshless methods, boundary element methods, and others.
3. Implementation aspects with the existing analysis programs have not been adequately discussed; this important aspect needs to be addressed (Biegler *et al*. 2003; Wang and Arora 2005a).
4. Sparse matrix approaches must be used to solve the problem with the SAND formulation since the optimization problem is large but sparsely populated.
5. Transformation of the solution variables needs to be considered, since various variables can be of different orders of magnitude.
6. Dynamic problems need to be considered to investigate applicability of the formulations for such problems.
7. Parallel processing must be considered to solve very large-scale problems.

CHAPTER 3
OPTIMIZATION FORMULATIONS

3.1 Introduction

The basic structural and mechanical system optimization problem includes optimal system design and optimal control. It is to determine design parameters, such as material properties, sizing/shape/topology variables, or/and control histories of a structural or mechanical system, to achieve a certain goal, e.g., minimization of cost or performance indices, while satisfying certain performance requirements.

In this chapter, the conventional formulation and a general framework for alternative formulations are presented and discussed for structural and mechanical system optimization problems. This is done by considering a linear structural problem (small displacements and linearly elastic material model) in the discretized form. The approach can also be described using a continuum form of the analysis equations that is more general because it is not tied to any particular discretization. However, this will not be done here to keep presentation of the basic ideas clear and straightforward. In the remaining chapters, the framework for alternative formulations is implemented to different structural and mechanical system optimization problems and numerical results are presented.

3.2 General Optimization Problems

The structural and mechanical system optimization problem is to find design variable vector $\mathbf{x}$, representing sizing/shape/topology and other system properties, to minimize a cost function

$$f = f(\mathbf{x}) \tag{3.2.1}$$

which may be the cost of the system, or any other function. Performance requirements are imposed mostly as inequality constraints (although equality constraints can also be treated routinely) as,

$$\mathbf{g}(\mathbf{x}) \le \mathbf{0} \tag{3.2.2}$$

Note that if optimal control problems are considered, the design variable vector $\mathbf{x}$ also includes control force variables, which are time histories in a transient dynamic system (Hull 1997, 2003). For simplicity of presentation, it is assumed that $\mathbf{x}$ includes all the design variables for optimal design problems, and design and control variables for optimal control problems. Therefore these two types of problems can be expressed and presented in the same way.

In structural and mechanical engineering, an important problem is to determine the response of the system to given inputs, such as displacement, deformation and eigenvalues. However, these values need to satisfy certain requirements, to make sure a system is safe and serviceable. Note here the inequality constraints may include the following stress and displacement constraints, and explicit bounds on the design variables, as

$$\boldsymbol{\sigma}^L \le \boldsymbol{\sigma} \le \boldsymbol{\sigma}^U \tag{3.2.3}$$

$$\mathbf{r}^L \le \mathbf{r} \le \mathbf{r}^U \tag{3.2.4}$$

$$\mathbf{x}^L \le \mathbf{x} \le \mathbf{x}^U \tag{3.2.5}$$

where $\boldsymbol{\sigma}^L, \mathbf{r}^L, \mathbf{x}^L$ and $\boldsymbol{\sigma}^U, \mathbf{r}^U, \mathbf{x}^U$ are the lower and upper bounds for the member stresses, nodal displacements and design variables, respectively. If the structural and mechanical system is a transient dynamic system, the constraints in Eqs. (3.2.3) and (3.2.4) are time-dependent, i.e., they are functions of time.

3.3 Conventional Formulation – Only Design Variables as Optimization Variables

As mentioned in Chapter 2, the conventional formulation means the optimization is carried out in the space of design variables, also called "nested" formulation. This is the most common way that has been used to formulate the optimization problems where sizing/shape or other design parameters are treated as optimization variables. The formulation includes the minimum number of optimization variables. Unfortunately, the constraint functions are implicit in terms of the variables since the displacement vector $\mathbf{r}$ is an implicit function of the variables. Therefore, the gradients of functions cannot be obtained easily.

3.3.1 Formulation

The optimization problem is to find design variable vector $\mathbf{x}$, to minimize the cost function in Eq. (3.2.1), subject to the constraints defined in Eq. (3.2.2). Equations (3.3.2) may include constraints in Eqs. (3.2.3) - (3.2.5) or other performance requirements. To analyze a structural system in order to obtain the response variables, such as stress $\boldsymbol{\sigma}$ and displacement $\mathbf{r}$, the equilibrium equations for static systems and the equations of motion for dynamic systems need to be solved using given stiffness and loading information. Note that since $\boldsymbol{\sigma}$ and $\mathbf{r}$ in Eqs. (3.2.3) - (3.2.5) are implicit with respect to the variables $\mathbf{x}$, they can only be evaluated by performing analyses. Thus the equilibrium equations have to be satisfied at each iteration.

3.3.2 Gradient Evaluation

Usually finite difference methods, including forward, backward and central differences, are very popular in engineering applications, since they are easy to implement and explicit expressions for functions are not needed. However, the finite difference methods have accuracy problems, i.e., the so-called "step-size" dilemma (Haftka and Gürdal 1992). Another drawback is that they are slow because they require a repeated solution of the state equations.

To evaluate gradients of the constraints analytically, implicit differentiation procedures need to be used. Taking the static analysis as an example, numerical values for $\mathbf{r}$ can be obtained from the state equations once $\mathbf{x}$ is specified:

$$\mathbf{K}(\mathbf{x})\mathbf{r} = \mathbf{R}(\mathbf{x}) \tag{3.3.1}$$

where (3.3.1) is the displacement-based equilibrium equation. $\mathbf{K}$ is the global stiffness matrix, whose elements, in many cases, are explicit with respect to $\mathbf{x}$. $\mathbf{R}$ is the external load vector that might also be function of $\mathbf{x}$, i.e., when the structural self-weight is considered. However, an explicit functional form for $\mathbf{r}$ in terms of $\mathbf{x}$ cannot be obtained. In the optimization process derivatives of the constraint functions $g(\mathbf{x},\mathbf{r}(\mathbf{x}))$ with respect to $\mathbf{x}$ are needed. This is called *design sensitivity analysis*. Taking total derivative of $g(\mathbf{x},\mathbf{r}(\mathbf{x}))$ with respect to $\mathbf{x}$, we get

$$\left.\frac{dg(\mathbf{x},\mathbf{r}(\mathbf{x}))}{d\mathbf{x}}\right|_{n\times 1} = \left.\frac{\partial g(\mathbf{x},\mathbf{r})}{\partial \mathbf{x}}\right|_{n\times 1} + \left.\frac{d\mathbf{r}(\mathbf{x})}{d\mathbf{x}}\right|_{n\times m} \left.\frac{\partial g(\mathbf{x},\mathbf{r})}{\partial \mathbf{r}}\right|_{m\times 1} \tag{3.3.2}$$

Calculation of the partial derivatives of $g(\mathbf{x},\mathbf{r}(\mathbf{x}))$ with respect to $\mathbf{x}$ and $\mathbf{r}$ presents no particular difficulty because explicit dependence of the function on $\mathbf{x}$ and $\mathbf{r}$

is known. However, calculation of $\frac{d\mathbf{r}(\mathbf{x})}{d\mathbf{x}}$ in Eq. (3.3.2) leads to two types of analytical sensitivity techniques, the so-called *direct differentiation method* and *adjoint variable method*. Under certain circumstances, one method is more efficient than the other. This depends on the relative number of constraints and design variables. Both methods require the derivatives of the stiffness matrix and the load vector with respect to design variables.

In any case, these analytical methods are quite difficult to implement into existing analysis codes. To evaluate these derivatives, element level quantities need to be differentiated with respect to design variables. Differentiated matrices need to be assembled to form global matrices. Also, the analysis software needs to be recalled to solve for displacement gradients or the adjoint vectors. Gradients of response functionals need to be assembled using the adjoint vectors or the displacement gradients. Further, implementation of design sensitivity analysis for nonlinear and multi-physics problems becomes more complex because $\mathbf{K}$ and $\mathbf{R}$ depend on the state of the system as well. This is one of the main stumbling blocks in practical applications of optimization. For structures subjected to dynamic loads or optimal control of dynamic systems, similar sensitivity analyses are needed and tend to be more complicated (Tseng and Arora 1989). Substantial literature is available that describes theoretical as well as implementation aspects of the design sensitivity analysis approaches (Haug *et al.* 1986; Arora 1995).

3.4 Alternative Formulation – Design and Behavior Variables as Optimization Variables

If some of the state or behavior variables, such as displacements and forces, are treated as variables in the optimization formulation, the implicit optimization problem can be transferred to an explicit form. SAND basically formulates the optimization problem in a mixed space of design and state variables, to imbed the analysis equations as equality constraints in one single optimization problem; therefore no explicit structural analysis or design sensitivity analysis is needed. The SAND formulation in the current section follows the most common way of presentation in the literature.

3.4.1 Formulation

In the SAND approach, the formulation of the problem is modified by treating the design and state variables, $\mathbf{x}$ and $\mathbf{d}$ as independent variables of optimization, to minimize

$$f = f(\mathbf{x},\mathbf{d}) \tag{3.4.1}$$

Subject to

$$\mathbf{h}(\mathbf{x},\mathbf{d}) = \mathbf{0} \tag{3.4.2}$$

$$\mathbf{g}(\mathbf{x},\mathbf{d}) \leq \mathbf{0} \tag{3.4.3}$$

where the vector $\mathbf{d}$ represents behavior variables such as displacements, forces, or stresses. Note that Eqs. (3.4.2) are the equality constraints between the variables, which are in fact a system of state equations representing the analysis of a structural and mechanical system. They are differential algebraic equations (DAEs), or differential equations (DEs) in a transient dynamic system. Equations (3.4.3) are the inequality constraints between the variables. Inequality constraints (3.4.3) may include those listed in Eqs. (3.2.3) - (3.2.5), while Eqs. (3.4.2) may include Eq. (3.3.1) and other equations. If the behavior vector $\mathbf{d}$ includes the displacement vector $\mathbf{r}$, Eqs. (3.3.1) become explicit with respect to the variables $\mathbf{d}$. Note that the equilibrium equations (3.3.1) can be written in a form such that the assembly of the global stiffness matrix is not needed. These aspects will be illustrated further in later chapters for different optimization problems.

3.4.2 Gradient Evaluation

In the optimization process, partial derivatives of the functions with respect to $\mathbf{x}$ and $\mathbf{d}$ are needed. Since the objective and constraint functions in Eqs. (3.4.1) - (3.4.3) in the alternative formulation are all explicit in terms of the optimization variables $\mathbf{x}$ and $\mathbf{d}$, the gradients of functions can be obtained easily by direct differentiation. Partial derivatives of $\mathbf{h}$ and $\mathbf{g}$ with respect to $\mathbf{x}$ and $\mathbf{d}$ can be easily calculated as noted before. Partial derivative of Eq. (3.3.1) with respect to $\mathbf{r}$ gives the stiffness matrix $\mathbf{K}$ and the partial derivative of Eq. (3.3.1) with respect to $\mathbf{x}$ needs the derivatives of the stiffness matrix. However, $d\mathbf{r}/d\mathbf{x}$ is not needed and no system of equations needs to be solved in the numerical solution process. Detailed derivations for different structural and mechanical system optimization problems will be presented in later chapters.

3.5 Discussion

Note that the SAND formulation does not require $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ be satisfied exactly at each iteration of the optimization process, i.e., the equilibrium equation (3.3.1) need not be satisfied at every iteration, which can be advantageous for nonlinear and dynamic problems. It needs to be satisfied only at the final solution point. This actually implies that the vector $\mathbf{h}(\mathbf{x,d}) = \mathbf{0}$ never needs to be solved for behavior variable vector $\mathbf{d}$, because $\mathbf{d}$ is treated as independent variables. The element level equilibrium equations can be used in the solution process. Thus the alternative formulation is ideally suited for implementation on a parallel computer where each finite element can be assigned to one processor. All processors can be used to generate the element level quantities and thus speed-up the optimization process considerably (Haftka 1985; Haftka and Kamat 1989).

Also as noted, the equilibrium equation (3.3.1) may not be the displacement based FEM equation, even though it is the most commonly used one. Most work in the literature has used displacements as optimization variables. However, similar to the conventional formulation, the force method or the mixed method can also be combined with the SAND formulations (Kirsch and Rozvany 1994). In summary, advantages of the alternative formulation are:

1. An explicit formulation is obtained, and gradient information can be calculated easily.
2. It does not require calculation or implementation of $\frac{d\mathbf{r}(\mathbf{x})}{d\mathbf{x}}$ which is the most troubling part of the optimization process.
3. It does not require decomposition of the $\mathbf{K}$ matrix; i.e., does not require solution of the equilibrium equation at each iteration of the optimization process.
4. Since the calculations can be performed at the element level, the approach can be easily implemented on massively parallel computers.
5. Implementation of the approach appears to be simpler.
6. Multidisciplinary (multi-physics) problems can be treated more easily.

Though looks very attractive, the alternative SAND formulations have some obvious drawbacks, requiring attention in this research.

1. The numbers of optimization variables and constraints become very large.

Therefore optimization methods for large-scale problems need to be used.

2. The optimization variables can have different orders of magnitudes. This may require development of special procedures for their treatment.

Many aspects of the alternative formulations will be discussed in details in later chapters when particular optimization problems and formulations are presented. Numerical examples will be used to illustrate the comparisons between the conventional and various alternative formulations.

CHAPTER 4
APPLICATION TO TRUSS DESIGN

4.1 Introduction

Truss is a simple but widely-used form of structures. In the design of trusses, like other structures, sufficient strength against failure and adequate stiffness need to be satisfied. Due to their simplicity, great success has been achieved in the design optimization of plane and space truss structures by various methods (Venkayya 1971; Haug and Arora 1979; Kirsch 1993). However, the traditional optimization methods need sensitivity analysis that is generally computationally expensive due to the implicit nature of the constraint functions, i.e., the displacement with respect to the design variables - the cross-sectional properties. Zhou and Rozvany (1992) developed an optimality criteria (OC) method for solving large-scale structural optimization problems. Adeli and Cheng (1994) applied genetic algorithm (GA) to the optimization of large-scale trusses. Adeli and Soegiarso (1999) optimized large-scale structures by parallel computing.

Although different formulations have been presented in the literature based on the SAND concept, limited work has been done to compare and evaluate their relative merits. In addition, a limited emphasis has been placed on implementation of the formulations with the existing simulation codes. In the present work, the idea of SAND is investigated with respect to these two aspects using linearly elastic trusses as the application area. The idea is to gain insights into the numerical performance of the formulations and their implementation with existing analysis codes. Three separate SAND formulations for the problem are defined, studied and evaluated. It is noted that although the formulations can be extended for topology design optimization, only the sizing design optimization problem is treated. It turns out that the topology design problem needs some additional considerations for implementation with existing analysis codes because the formulations lose some of their nice features that are available for the sizing design problem. This is explained later in this chapter.

In the present work, a robust sparse SQP program is used for numerical solutions of different formulations. An available structural analysis code, ANSYS (2002), is integrated in the optimization process to evaluate various structural response quantities. The conventional NAND formulation is also implemented with the ANSYS program. The role of the structural analysis software in the optimization process with these alternative formulations is studied and explained. This part of the investigation will facilitate integration of different analysis codes into the optimization process for multidisciplinary applications. All the formulations are evaluated using several truss structures that have been used as test problems in the literature. Results with different formulations are compared and advantages and disadvantages of the formulations are discussed (Wang and Arora 2005a).

4.2 Truss Analysis

For a general space truss, the equilibrium equation for member *i* in a global coordinate system is given as:

$$\mathbf{Q}_i = \mathbf{k}_i \mathbf{q}_i = A_i \bar{\mathbf{k}}_i \mathbf{q}_i \tag{4.2.1}$$

where A_i is cross-sectional area of member *i*, $\mathbf{k}_i$ is the member stiffness matrix in the global coordinate system, and $\bar{\mathbf{k}}_i$ is independent of design variable A_i. $\mathbf{Q}_i$ and $\mathbf{q}_i$ are member-end force and displacement vectors in the global coordinate system. The internal axial force F_i in member *i* is calculated as

$$F_i = \mathbf{b}_i^{FQ}\mathbf{Q}_i \tag{4.2.2}$$

where $\mathbf{b}_i^{FQ}$ is a 1 × 6 transformation vector between nodal forces and axial force of member *i*, given as

$$\mathbf{b}_i^{FQ} = \begin{bmatrix} 0 & 0 & 0 & \lambda_x & \lambda_y & \lambda_z \end{bmatrix}_i \tag{4.2.3}$$

λ_x, λ_y and λ_z are the direction cosines of member *i* with respect to the global *x*, *y* and *z* directions. Expressing Eq. (4.2.1) in another way, we have,

$$\mathbf{Q}_i = \mathbf{b}_i^{QF} F_i \tag{4.2.4}$$

in which $\mathbf{b}_i^{QF}$ is a 6 × 1 transformation vector, given as

$$\mathbf{b}_i^{QF} = \begin{bmatrix} -\lambda_x & -\lambda_y & -\lambda_z & \lambda_x & \lambda_y & \lambda_z \end{bmatrix}_i^T \tag{4.2.5}$$

Therefore, the component in the direction *j* for the member-end force vector $\mathbf{Q}_i$ is given as

$$Q_{ji} = [\mathbf{Q}_i]_j = \eta_{ji} F_i \tag{4.2.6}$$

where η_{ji} represents one component in $\mathbf{b}_i^{QF}$. The member nodal displacement vector $\mathbf{q}_i$ in the global coordinate system is related to the total displacement vector $\mathbf{r}$ by a $6 \times m$ Boolean matrix $\mathbf{Z}_i$:

$$\mathbf{q}_i = \mathbf{Z}_i \mathbf{r} \tag{4.2.7}$$

Therefore from Eqs. (4.2.1), (4.2.2) and (4.2.7), the internal force and stress in member *i* are given as

$$F_i = A_i \mathbf{B}_i \mathbf{r}\, ; \quad \sigma_i = \mathbf{B}_i \mathbf{r}\, ; \text{ where } \mathbf{B}_i = \mathbf{b}_i^{FQ} \overline{\mathbf{k}}_i \mathbf{Z}_i \tag{4.2.8}$$

$\mathbf{B}_i$ is a $1 \times m$ vector that is independent of $\mathbf{A}$ and $\mathbf{r}$ and contains stiffness coefficients per unit area in the global coordinate system for member *i*. Vectors $\mathbf{B}_i$ need to be calculated only once during the entire solution process.

4.3 Conventional Formulation (T-CF) – Only Areas as Optimization Variables

Only areas of the members are taken as optimization variables and the general forms of Eqs. (3.2.1) to (3.2.5) are reduced to:

Minimize

$$f(\mathbf{A}) = \sum_{i=1}^{n} L_i A_i \tag{4.3.1}$$

Subject to

$$\sigma_i^L \le \sigma_i(\mathbf{A}) \le \sigma_i^U , \qquad i = 1, n \tag{4.3.2}$$

$$r_j^L \le r_j(\mathbf{A}) \le r_j^U , \qquad j = 1, m \tag{4.3.3}$$

$$A_i^L \le A_i \le A_i^U , \qquad i = 1, n \tag{4.3.4}$$

where L_i is the length, and A_i^L and A_i^U are the lower and upper bounds on the areas of member *i*, respectively. Although upper bounds on the areas may not be necessary for all applications, they are kept in the formulation for generality. Some applications may need to limit the largest size of a member that can be used in the structure. Note that since the stress σ_i and the displacement r_j are implicit functions of areas, implicit differentiation procedures are needed to solve for their derivatives, as explained earlier.

4.4 Alternate Formulation 1 (T-AF1) – Areas and Nodal Displacements as Optimization Variables

If the displacements are also treated as optimization variables in the formulation, the implicit problem formulation becomes explicit in terms of the variables. This formulation has been used by several researchers in the literature (Haftka 1985; Orozco and Ghattas 1992a). The problem becomes: minimize the weight of Eq. (4.3.1) subject to the following constraints, in addition to the explicit bound constraints in Eqs. (4.3.3) and (4.3.4):

$$h_j^g = \sum_{k=1}^{NE_j} Q_{jk}(\mathbf{A},\mathbf{r}) - R_j = 0\,;\;\; Q_{jk} = \eta_{jk} A_k \mathbf{B}_k \mathbf{r}\,, \qquad j = 1, m \tag{4.4.1}$$

$$g_i^\sigma = \sigma_i(\mathbf{r}) - \sigma_i^U \le 0\,;\;\; \sigma_i = \mathbf{B}_i \mathbf{r}\,, \qquad i = 1, n \tag{4.4.2}$$

where Q_{jk} is the nodal force of member *k* along the *j*th global displacement. A total of NE_j members are connected to the same node. R_j is the external load acting at the node along the *j*th displacement. Equation (4.4.1) is the equilibrium equation for the *j*th degree of freedom; i.e., it represents the *j*th row of Eq. (3.3.1) and $\eta_{jk} A_k \mathbf{B}_k$ contributes to the elements of the *j*th row of $\mathbf{K}$. Thus this formulation requires assembly of the stiffness matrix, although its decomposition is not needed.

Since all the functions in Eqs. (4.4.1) and (4.4.2) are explicit in terms of the optimization variables A_i and r_j, the required derivatives are obtained quite easily. Note that differentiation of the equilibrium equation (4.4.1) with respect to the displacements r_j gives elements of the stiffness matrix $\mathbf{K}$.

4.5 Alternate Formulation 2 (T-AF2) – Areas, Nodal Displacements and Internal Forces as Optimization Variables

In T-AF1, all functions of the problem were required to be expressed in terms of the areas and displacements for various derivations and numerical calculations. However, if member forces or stresses are also treated as variables, it will permit expressing some constraints in terms of forces or stresses, thus simplifying their expressions. This may lead to simpler computer implementation and gradient evaluations. The idea of using internal forces as additional optimization variables has been studied for reformulation of topology optimization problems in the literature (Achtziger 1999; Stope and Svanberg 2003, 2004). Although there are variations of different formulations, the essential idea is to replace the equilibrium equations (4.4.1) by a system of linear equations in terms of the member forces.

By treating the displacements and the member forces F_i as variables in the optimization process, the T-AF2 is to minimize the weight of Eq. (4.3.1), subject to

$$h_j^g = \sum_{k=1}^{NE_j} Q_{jk}\left(\mathbf{A}, \mathbf{r}, \mathbf{F}\right) - R_j = 0, \qquad j = 1, m \tag{4.5.1}$$

$$h_i^e = F_i - A_i \mathbf{B}_i \mathbf{r} = 0, \qquad i = 1, n \tag{4.5.2}$$

$$g_i^\sigma = F_i - \sigma_i^U A_i \le 0, \qquad i = 1, n \tag{4.5.3}$$

Substituting for Q_{jk} from Eq. (4.2.6), the equality constraint in Eq. (4.5.1) - the equilibrium equations for each degree of freedom - is obtained as linear equation in the variables F_i as

$$h_j^g = \sum_{k=1}^{NE_j} \eta_{jk} F_k - R_j = 0, \qquad j = 1, m \tag{4.5.4}$$

The derivatives of Eqs. (4.5.2) to (4.5.4) with respect to the variables A_i, r_j and F_k can be calculated easily. Introduction of the force variables brings more freedom in the formulation. For example, it is also viable to keep the equilibrium equations in the same form as in Eq. (4.4.1).

4.6 Alternate Formulation 3 (T-AF3) – Areas, Nodal Displacements and Stresses as Optimization Variables

Using the displacements and the member stresses simultaneously as optimization variables, another explicit formulation for the problem is obtained. Similar to T-AF2, various stress-based formulations have been used for truss topology optimization (Achtziger 1999; Stope 2003). The problem is to minimize the weight of Eq. (4.3.1), subject to

$$h_j^g = \sum_{k=1}^{NE_j} Q_{jk}(\mathbf{A}, \mathbf{r}, \boldsymbol{\sigma}) - R_j = 0, \qquad j = 1, m \tag{4.6.1}$$

$$h_i^e = \sigma_i - \mathbf{B}_i \mathbf{r} = 0, \qquad i = 1, n \tag{4.6.2}$$

The constraints for stresses and displacements in Eqs. (4.3.2) and (4.3.3) represent explicit bounds on the variables. To obtain the equality constraints in Eq. (4.6.1) in terms of stresses, substitute Eq. (4.2.6) and $F_k = A_k \sigma_k$, to obtain

$$h_j^g = \sum_{k=1}^{NE_j} \eta_{jk} A_k \sigma_k - R_j = 0, \qquad j = 1, m \tag{4.6.3}$$

Since the member stress and nodal displacements are treated as independent variables, the equality constraints in Eq. (4.6.2) are given in the linear form. As for T-AF2, introduction of the stress variables brings more freedom in the formulation. For example, it is viable to keep the equilibrium equations in the same form as in Eq. (4.4.1), such as that formulated by Stope (2003). This formulation has been tested and shown to be computationally not as efficient as Eq. (4.6.3). Therefore Eq. (4.6.3) is adopted in the present implementation.

Table 4.1 Sizes of different formulations for trusses

Item	T-CF	T-AF1	T-AF2	T-AF3
No. of Variables	n	$n+Lm$	$n+L(n+m)$	$n+L(n+m)$
No. of Equality Constraints	0	Lm	$L(m+n)$	$L(m+n)$
No. of Inequality Constraints	$L(n+m)$	Ln	$2Ln$	0
No. of Simple Bounds	n	$n+Lm$	$n+Lm$	$n+L(n+m)$

Table 4.1 summarizes the sizes of all the formulations in terms of numbers of variables and constraints (L is the number of loading conditions). It is clear that the size of the SAND formulations (in terms of the numbers variables and constraints) can be quite large depending on the number of degrees of freedom and the number of loading cases, L. Note however that although the conventional formulation has the smallest number of optimization variables, it has the largest number of inequality constraints. These constraints are also implicit functions of the variables requiring the use of special design sensitivity analysis procedures. However, this formulation has no equality constraints while the alternative formulations have many.

4.7 Implementation with Existing Programs

The SAND formulations are solved using the sparse SQP algorithm in SNOPT (Gill *et al.* 2002), while the conventional formulation is solved using the dense SQP solver since the problem is dense. SNOPT is a stand-alone program that uses text files to communicate with the commercial package ANSYS. It is noted that an interior point method was also tried with the formulations; however, its performance was not as good as with SQP for the example problems. Therefore, only the results with SNOPT are reported. To use the algorithm, cost and constraint functions and their gradients need to be calculated. For the conventional formulation, ANSYS (2002) is used to analyze the structure and to calculate the displacement gradients. ANSYS is also called during the step size calculation to analyze the structure and evaluate the problem functions.

For the T-AFs, the cross-sectional areas and displacements are sent to ANSYS to calculate the member level quantities that are used to form the constraint functions, e.g., member level or node level equilibrium equations. This procedure is also used during the step size calculation along the search direction. Derivatives of various constraint functions are evaluated external to ANSYS using the member stiffness matrices and member connectivity information. It is important to note that for the constraints that are linear in variables, such as in Eqs. (4.5.3) and (4.5.4), their derivatives are calculated only once during the entire solution process.

4.7.1 T-CF

The constraint functions in Eqs. (4.3.2) and (4.3.3) are evaluated using ANSYS results. The constraints are normalized with respect to their limit values. The constraint gradients are evaluated using the direct differentiation method mentioned in Eq. (3.3.2).

The matrix $d\mathbf{r}/d\mathbf{x}$ is calculated by restarting ANSYS with the updated sensitivity loading vectors, whose number depends on the number of design variables. In this process, additional assembly of the global stiffness matrix and its decomposition are not needed.

4.7.2 T-AF1

In numerical implementation, the constraints in Eqs. (4.4.1) and (4.4.2) are normalized by R_j and σ_i^U (or σ_i^L), respectively. In evaluation of the constraints in Eq. (4.4.1), the member nodal forces from ANSYS output are used directly. Also for the inequalities in Eq. (4.4.2), the member stresses from ANSYS output are directly used. The derivatives of the constraints are evaluated using the current values of $\mathbf{A}$ and $\mathbf{r}$, and the vector $\mathbf{B}_i$ calculated for the *i*th member. It is important to note that the calculations for derivatives for T-AF1 and the conventional formulation are similar, in the sense that they both need similar member level derivatives and their assembly to form the final gradients. The difference is that the structural and the sensitivity analysis equations are not solved in T-AF1.

4.7.3 T-AF2

In numerical calculations, R_j is used to normalize the equality constraints in Eq. (4.5.4). Equation (4.5.2) is normalized by using the largest external load. The equilibrium constraints in Eq. (4.5.4) are evaluated directly by using the current values of the force variables F_i. The equality constraints in Eq. (4.5.2) are evaluated using the internal forces read from the ANSYS output file and the force variables F_i; i.e., $\mathbf{A}$ and $\mathbf{r}$ are not used to evaluate Eq. (4.5.2). The derivatives of the constraints functions are evaluated using the current values of the variables and the vector $\mathbf{B}_i$.

4.7.4 T-AF3

Similar to T-AF1 and T-AF2, constraints in Eq. (4.6.3) are normalized using R_j. Also equality constraints in Eq. (4.6.2) are normalized using the allowable member stress. T-AF2 and T-AF3 look similar; however, there are less behavior constraints in T-AF3, since the stress constraints also become simple bound on the variables. The equality constraints in Eqs. (4.6.2) and (4.6.3) and their derivatives are explicit in terms of the optimization variables $\mathbf{A}$, $\mathbf{r}$ and σ_i. The variables σ_i and $\mathbf{A}$ are used directly to evaluate the equality constraints in Eq. (4.6.3). For efficiency of calculations, internal stresses in the ANSYS output file (instead of the current nodal displacement vector $\mathbf{r}$) and the current values of the stress variables σ_i are used directly to evaluate the equality constraints in Eq. (4.6.2).

4.8 Numerical Examples

The conventional and three alternative formulations have been used to solve truss problems. However, for sake of brevity results for only three example problems are presented in order to evaluate the formulations. Since details for the examples have been presented in numerous references (Haug and Arora 1979; Haftka 1985; Zhou and Rozvany 1993), only a brief description of them is included here. A PC with 2.53 GHz processor and 1 GB RAM are used for running the programs and recording the CPU times. ANSYS resides on a local area network and is executed on a server. It is noted that very severe stopping criteria are used to obtain precise optimal solutions. With relaxed stopping criteria, solutions that are very close to the true solutions can be obtained with smaller computational effort.

4.8.1 Example 1 – 25-bar space truss

Stress and displacement limits are imposed for this structure (Haug and Arora 1979; Zhou and Rozvany 1993). The displacement limit is ± 0.00889 m for each node, and the lower limit for cross-sectional areas is 6.4516×10^{-6} m^2. Two loading conditions are imposed and design variable linking is used. There are 8, 44, 94 and 94 independent variables, and 180, 216, 266 and 266 constraints for the four formulations, respectively. The initial values for the variables are taken as follows: areas as one, displacements as 0.00889 m, and the normalized stresses and forces as one. All the four formulations obtain the known optimum volume as 0.0894 m^3.

4.8.2 Example 2 – 72-bar space truss

A 4-story 72-bar space truss is considered next (Haftka 1985; Zhou and Rozvany 1993). The stress limit for each member is 172.375 MPa and the displacement limit is ± 0.00635 m for each node. The minimum member size is taken as 6.4516×10^{-5} m^2. Two design cases are considered. For Case 1, optimum design without displacement constraints is obtained. Design variable linking is not used and only one loading condition is imposed (Haftka 1985). The initial values of the variables are as follows: areas as 6.4516×10^{-5} m^2, displacements as 0.0254, 0.0254, and -0.0254 m along the *x*, *y* and *z* directions for the upper two stories and 0.0127, 0.0127, and -0.0127 m for the lower two stories, respectively, and the normalized stresses and forces as one.

For Case 2, two loading conditions are imposed and design variable linking is used (Zhou. and Rozvany 1993). There are 16 independent sizing variables for all the formulations. For the alternative formulations, there are 112, 256 and 256 optimization variables, and 592, 736 and 736 constraints (including 96, 240 and 240 equality constraints), respectively. The initial values of the variables are as follows: areas as 6.4516×10^{-5} m^2, displacements as 0.00635, 0.00635, and -0.00635 m along the *x*, *y* and *z* directions for the upper two stories and 0.00254, 0.00254, and -0.00254 m for the lower two stories, respectively, and the normalized stresses and forces as one. All formulations give the known optimum solutions and the final volumes for the two cases as 0.0015666 and 0.00622 m^3, respectively.

4.8.3 Example 3 – 200-bar plane truss

Two design cases are considered for this 200-bar plane truss that is subjected to three loading conditions (Haug and Arora 1979). The allowable stress for each member is 206.844 MPa. A displacement limit of ± 0.0127 m is imposed at all the free nodes. The minimum member size is taken as 6.4516×10^{-5} m^2. For Case 1, only the stress constraints are imposed. The members of the structure are linked into 96 design variables. Therefore there are 96, 546, 1146, and 1146 optimization variables, and 600, 1050, 2250 and 1050 behavior constraints (excluding simple bounds) for the four formulations, respectively. The initial values for the variables are as follows: all the cross-sectional areas as one, all the displacements as 0.00254 m, and the normalized stresses and forces as one. The final volume obtained is 0.4323 m^3 with all the formulations, which is 0.0012 m^3 smaller than the one reported by Haug and Arora (1979).

For Case 2, both the displacement and stress constraints are imposed. The numbers of optimization variables are the same as for Case 1 and the numbers of behavior constraints are 1050, 1050, 2250 and 1050, respectively. A comparison of results shows that the final volume with the alternative formulations of 1.5960 m^3 is smaller by 0.0811 m^3 than the one reported by Haug and Arora (1979), and the optimum volume of 1.6664 m^3 obtained with the conventional formulation, is also smaller by 0.0107 m^3 than the reported solution. Note that there are many linear constraints in the alternative formulations; therefore, it is possible that the optimization algorithm finds a better local minimum point compared to the conventional formulation.

4.8.4 Discussion of results

1. Computational effort

It is observed that all the formulations converged to optimal solutions for the problems. For the 200-bar structure, better optimal solutions were obtained than the known solutions. For all the examples, the numbers of iterations and calls to ANSYS are given in Tables 4.2 and 4.3, respectively. It is seen that the alternative formulations generally take more iterations to find optimal solutions, since there are more optimization variables. The size of the problem becomes still larger when the structure is subjected to multiple loading conditions. However, each iteration requires less computation because the solution of the equilibrium equations is avoided, and the design sensitivity analysis equations are not solved. The CPU times of all the examples are reported in Table 4.4. It is seen that the computational efforts for the three alternative formulations are in general smaller than those with the conventional formulation. Among the alternative formulations, T-AF2 generally requires less CPU effort compared to the other two. This is due to the simpler forms of the constraints and their gradients.

It is noted that the number of iterations and the computational effort to obtain the optimal solution can vary depending on the starting point. For example, numerical experiments with Case 2 of the 72-bar truss gave the following range of iterations for different starting points with T-AF1, T-AF2 and T-AF3: 49 – 71, 41 – 59, and 34 – 41.

Table 4.2 Numbers of iterations for design examples

Example Problem		T-CF	T-AF1	T-AF2	T-AF3
25-bar		14	65	31	41
72-bar	Case 1	21	3	7	12
	Case 2	42	49	41	40
200-bar	Case 1	28	54	47	38
	Case 2	91	316	211	114

Table 4.3 Numbers of calls to ANSYS for design examples

Example Problem		T-CF	T-AF1	T-AF2	T-AF3
25-bar		25 + 23*	76	53	59
72-bar	Case 1	40 + 38	7	17	22
	Case 2	85 + 83	64	47	69
200-bar	Case 1	48 + 46	69	75	55
	Case 2	223 + 221	432	245	145

*Analysis + Sensitivity Analysis

Table 4.4 Computing effort for different formulations (CPU, s)

Example Problem		T-CF	T-AF1	T-AF2	T-AF3
25-bar		1.52	3.30	2.48	2.66
72-bar	Case 1	4.67	0.61	1.08	1.34
	Case 2	8.41	4.20	3.48	5.03
200-bar	Case 1	59.63	52.52	20.41	55.61
	Case 2	317.22	205.03	109.59	136.66

It is important to note that the wall-clock times with all the formulations are much larger than the actual CPU times. This is due to the fact that ANSYS is used as an independent program executed on a local area network. For example, the wall clock times for Case 2 of the 200-bar truss with the four formulations are: 15679, 3048, 1329 and 774 s, respectively. These are much larger than the actual CPU time. Thus there is considerable overhead time in the use of a commercial analysis program over the local area network. The conventional formulation has the largest overhead since the number of calls to ANSYS is the largest.

It is also observed that although a smaller number of major iterations is needed by SNOPT for some formulations, the CPU time may still be larger compared to the other formulations. This is due to the use of a larger number of iterations to solve the QP subproblem for the search direction which may be due to the forms of the constraint functions and their gradients. This is seen in Tables 4.2 and 4.4 for Case 2 of the 200-bar truss example with T-AF2 and T-AF3.

In general, since the SAND formulations avoid repeated analysis of the structure and design sensitivity analysis, they are more efficient. This will also be the case for nonlinear structures where the conventional formulations need to solve nonlinear equilibrium equations at each iteration, which is very expensive. SAND formulations also simplify the forms of constraints and Jacobian matrices, which are advantageous for numerical algorithms and implementations.

2. Scaling of variables

An important point to note here is that the alternative formulations include different types of variables, which have different orders of magnitudes. Therefore scaling of some of the variables is necessary to reduce numerical ill-conditioning. SNOPT has two options for automatic scaling: one for linear constraints and variables, and the other for all constraints and variables. In addition, manual scaling of the stress and force variables was implemented. Several solutions for Case 2 of the 72-bar truss were obtained by using different scaling options. In these experiments, the displacement variables were not scaled manually. The numbers of iterations for the T-AF1 to T-AF3 varied as follows: 49 – 69, 34 – 200+, and 40 – 200+, respectively. Thus it is seen that there can be considerable variation in the computational effort to obtain optimal solutions with the SAND type formulations. This clearly shows that more research is needed on this aspect of automatically scaling the optimization variables for good performance of the optimization algorithms.

3. Role of analysis programs

As a result of this research, the role of existing analysis programs has become clearer with the SAND type formulations. Basically, the pre-and post-processing capabilities of the analysis programs can be used directly to evaluate the constraint functions. Evaluation of gradients, however, requires member connectivity information and member stiffness matrices, and needs to be implemented outside the analysis program. These implementations can become easier if the internal member level subroutines and processes of the analysis program are accessible directly from the optimization program. In evaluating the formulations with truss structures, it was possible to implement the member quantities explicitly to evaluate gradients of the constraints. For more complex problems this may be difficult to achieve. In that case, the finite difference method at the member level may offer a reasonable solution, which needs further evaluation.

The foregoing remarks also apply to the implementation of the conventional formulation. For gradient evaluation, however, the right side of Eq. (3.3.5) or Eq. (3.3.7) must be assembled, and the analysis program must be restarted to evaluate the displacement gradients or the adjoint vectors. Therefore, the analysis program must have restart capability; otherwise, the stiffness matrix and its decomposition will have to be re-generated which is inefficient. If the program uses iterative procedures to solve the system of equations or some other approximate procedures, then this step of the solution process will be quite time consuming. Due to these reasons, implementation of the conventional formulation is more tedious than the SAND formulations.

4.9 Evaluation of Formulations

4.9.1 SAND vs. NAND

Advantages and disadvantages of the conventional and alternative SAND formulations are summarized in Table 4.5. A major disadvantage of the conventional formulation is that the equilibrium equations in terms of the displacements must be formed and solved during each iteration. The same process must also be repeated during step size calculation along the search direction. Another disadvantage is that the gradient evaluation requires solution of linear equations which must be done by restarting the analysis program. This is cumbersome making the implementation of the process more tedious depending on the facilities available in the analysis program. In addition, the Jacobian and Hessian matrices of the constraints are dense, limiting the size of the design problem that can be treated efficiently. It is noted that all the formulations need partial derivatives of the member equilibrium equations with respect to the design variables and displacements. The difference is in the use of these derivatives in different formulations.

A major disadvantage of the alternative formulations is that the numbers of variables and constraints become very large, although the problem functions are highly sparse. Therefore sparsity must be utilized to solve the optimization problem efficiently;

i.e., optimization algorithms for large sparse problems must be used. This also requires a thorough knowledge of the sparsity structure of the problem functions.

Table 4.5 Advantages and disadvantages of conventional and alternate SAND formulations

Formulation	Advantages	Disadvantages
Conventional	1. Least number of optimization variables. 2. Equilibrium condition is satisfied at each iteration. 3. Intermediate solutions may be usable.	1. Equilibrium equation must be explicitly solved, which is expensive. 2. Constraints are implicit functions of the variables. Their evaluation requires structural analysis; e.g., during step size calculation as well. 3. Design sensitivity analysis procedures must be used to evaluate gradients. 4. Implementation with the analysis programs is tedious, requiring restart capabilities. 5. Dense Jacobian and Hessian matrices; difficult to treat large number of design variables.
Alternative	1. Formulations are explicit in terms of the variables. 2. Equilibrium equation in terms of the displacements is not solved. 3. Many constraints become linear in variables; the displacements constraints are simple bounds on the variables. 4. Jacobians and Hessian are sparse. 5. Design sensitivity analysis is not needed. 6. Implementation with existing analysis software is relatively straightforward existing software is quite straightforward.	1. Numbers of variables and constraints are very large. 2. Optimization algorithms for large-scale problems and sparsity of the problem must be utilized. 3. Optimization variables must be normalized. 4. Intermediate solutions may not be usable.

4.9.2 SAND

Based on the present implementation and the study, advantages and disadvantages of the three alternative SAND formulations for trusses are summarized in Table 4.6; the framed structures have similar advantages and disadvantages. The displacement constraints are always simple bounds on the variables. In T-AF3, stress constraints also become simple bounds on the variables, and all the remaining constraints are equalities. The stress constraints in Eq. (4.5.3) and the equilibrium constraints in Eq. (4.5.4) in T-AF2 are linear in optimization variables which are treated more efficiently in computations. It is not necessary to include member forces and stresses as variables in the formulations to obtain explicit expressions for constraints and Jacobian matrices; however, a careful look at the formulations shows that T-AF2 and T-AF3 have simpler forms of constraints and their Jacobian matrices than T-AF1, since the vector $\mathbf{B}_i$ appears only once in these two formulations (Eqs. (4.5.2) and (4.6.2) respectively). Both the constraints in Eqs. (4.4.1) and (4.4.2) in T-AF1 contain $\mathbf{B}_i$. Note also that implementation of the constraints in Eq. (4.4.1) and its Jacobian matrix is more tedious, which makes T-AF1 not as attractive as the other two formulations. T-AF2 and T-AF3 "decouple" the system equilibrium equations in Eq. (3.3.1) into separate equilibrium equations for the entire structure in terms of forces, and the equilibrium equations for each member in terms of the displacements. Although more variables are introduced, their inclusion however simplifies the functions and their gradient expressions, and computer implementations. They also lead to more sparse Jacobian matrices which improves performance of the formulations further. Thus, although T-AF2 and T-AF3 have more variables and constraints, they work better than T-AF1 in most cases. Note that even though there are more inequality constraints in T-AF2 than T-AF3, T-AF2 still performed well or even better than T-AF3, since the additional constraints are linear which can be treated more efficiently.

It is also possible to include the 6×1 internal force vector $\mathbf{Q}_i$ in Eq. (4.2.1) as optimization variables in T-AF2, instead of axial force F_i. Although this adds five more variables for one truss element, the formulation is more general for other finite elements, such as frames.

Table 4.6 Comparison of alternative SAND formulations

	T-AF1	**T-AF2**	**T-AF3**
Optimization Variable	$\mathbf{A},\mathbf{r}$	$\mathbf{A},\mathbf{r},\mathbf{F}$	$\mathbf{A},\mathbf{r},\boldsymbol{\sigma}$
Equilibrium Constraints	$\sum_{k=1}^{NE_j} \eta_{jk} A_k \mathbf{B}_k \mathbf{r} = R_j$ Assembly of global stiffness matrix needed	$\sum_{k=1}^{NE_j} \eta_{jk} F_k = R_j$ Linear constraints; No assembly of global stiffness matrix	$\sum_{k=1}^{NE_j} \eta_{jk} A_k \sigma_k = R_j$ Bilinear form; No assembly of global stiffness matrix
Element Equilibrium Constraints	--	$F_i = A_i \mathbf{B}_i \mathbf{r}$ Bilinear form	$\sigma_i = \mathbf{B}_i \mathbf{r}$ Linear constraints
Stress Constraints	$\sigma_i^L \le \mathbf{B}_i \mathbf{r} \le \sigma_i^U$ Linear constraints	$\sigma_i^L A_i \le F_i \le \sigma_i^U A_i$ Linear constraints	Simple bounds
Advantages	1. Fewer optimization variables.	1. The assembly of the global stiffness matrix and its derivatives are avoided. 2. Very sparse Jacobian and Hessian matrices. 3. Implementation with existing programs is straightforward. 4. Stress constraints are linear in T-AF2, and simple bounds in T-AF3.	
Disadvantages	1. The assembly of the global stiffness matrix and its derivatives are needed. 2. Derivative calculation and implementation is more tedious. 3. Denser Jacobian matrices.	1. Larger numbers of variables and constraints.	

The issue of implementation of formulations (essentially T-AF1) with existing simulation codes has been discussed by Biegler *et al.* (2003) for more general applications, such as PDE-constrained optimization problems. It is noted there that use of the Jacobian of the equilibrium constraints (stiffness matrix) in the optimization process is a major difficulty. Some simulation codes do not explicitly generate this Jacobian matrix while others use only an approximation for it. Therefore, optimization with such codes is a challenging problem. The present work shows that with T-AF2 and T-AF3, explicit Jacobian of the equilibrium constraints in terms of displacements is not required. Therefore, it is possible to use these types of formulations for optimization of more general problems with the existing simulation codes.

4.9.3 Topology optimization

Although the current study focuses on the sizing design problem, the formulations can be used for the topology design problem as well. The beauty of the SAND formulations for topology design is that both cross-sectional areas and displacements are treated as independent variables; therefore it is possible for the cross-sectional area to reach a zero value without causing singularity or non-differentiability. If a member is removed from the structure, the corresponding stress constraint is no longer included. This is achieved by simply modifying some constraint expressions in the formulations. In T-AF1, the stress constraints in Eq. (4.4.2) can be re-written as

$$A_i\sigma_i^L \le A_i\mathbf{B}_i\mathbf{r} \le A_i\sigma_i^U, \qquad i = 1, n \tag{4.9.1}$$

If $A_i = 0$, Eq. (4.9.1) is automatically satisfied, which means that the constraints are effectively removed. In T-AF2, Eq. (4.9.1) is obtained by combining Eqs. (4.5.2) and (4.5.3). In T-AF3, the simple bound on the stress constraints must be reformulated as nonlinear constraints:

$$A_i\sigma_i^L \le A_i\sigma_i \le A_i\sigma_i^U, \qquad i = 1, n \tag{4.9.2}$$

When combined with Eq. (4.6.2), the same form as in Eq. (4.9.1) is obtained

Note that in the present work, A_i is not allowed to reach a zero value; therefore all the formulations are equivalent. However, some nice features of the sizing design problem are lost when the formulations are extended to topology design. First of all, Eq. (4.9.1) becomes a bi-linear form instead of the linear one, as in Eq. (4.4.2). In T-AF3, the stress constraints in Eq. (4.9.2) are bi-linear forms instead of simple bounds on the variables. Since the Jocobian matrix of the constraints needs to be calculated during each iteration (as opposed to only once for the linear constraints during the entire solution process), these changes make the implementation more tedious, requiring more computations for the topology design problems.

4.10 Summary

Based on the extensive numerical experiments with the formulations, the following conclusions and observations are made:

1. Alternative SAND formulations are more efficient than the conventional formulation.
2. In T-AF2 and T-AF3, the global equilibrium equations in terms of the displacements are not needed. These equations are formed and used in terms of the element nodal forces. Only the element equilibrium equations in terms of displacements involving the element stiffness matrices are needed. Thus these formulations do not require assembly of the global stiffness matrix; i.e., the Jacobian matrix of the global equilibrium equations in terms of displacements. This is a major advantage of these formulations because for more complex applications, this matrix may not be available from the analysis

code (Biegler *et al.* 2003; Arora and Wang 2005).

3. T-AF2 where the forces are also used as variables is better than T-AF1 and T-AF3. It is easier to generalize the formulation for other finite elements to model and design more complex structures.
4. Normalization (scaling) of the variables is needed in the SAND formulations. More effective automatic scaling procedures need to be developed to improve efficiency of the formulations.
5. Sparsity of the problem functions must be utilized for efficiency and effectiveness of the SAND formulations.
6. Implementations with the SAND formulations with the existing analysis codes is simpler compared to the conventional formulation in the sense that no system of equations needs to be formed and solved for gradient evaluations. This is highly advantageous for complex applications where the simulation code may be using iterative or approximate procedures to solve the governing equations. For such cases the sensitivity calculations for the conventional formulation become tedious and inefficient.

CHAPTER 5
APPLICATION TO FRAME DESIGN

5.1 Introduction

Three alternative formulations based on the concept of simultaneous analysis and design (SAND) have been presented in Chapter 4. The formulations involved combinations of displacements, axial forces and stresses as optimization variables. Introduction of more variables in the formulations changed forms of the constraints and their derivatives. All functions of the formulations became explicit in terms of the optimization variables. Therefore, design sensitivity analysis methods, which require adaptation of structural analysis procedures for optimization, were no longer needed. All the formulations worked quite well and very accurate optimal solutions were obtained. Solutions of sample problems were also compared with those obtained by the conventional formulation. It was concluded that the alternative formulations were more efficient than the conventional formulation in most cases. Also their implementation with the existing analysis software was easier compared to that for the conventional formulation. In the present chapter, two of the formulations are extended and evaluated for optimization of framed structures.

An overview of the literature on different formulations for structural optimization has been presented in Chapter 2 and the review paper by the authors (Arora and Wang 2005). Here an overview of the literature related to the framed structures is presented. Optimal design of framed structures has been actively studies in the literature (Chan *et al.* 1995; Adeli and Soegiarso 1999; Pezeshk *et al.* 2000; Arora 2002). An extensive list of references on the subject can be found in Burns (2002). Khan (1984), Sadek (1992), and Chan *et al.* (1995) formulated the problem of optimum design of frames using some approximate relationships between cross-sectional properties and used the optimality criterion method to obtain solutions. Saka (1980) presented a formulation that included the displacements of joints as optimization variables in addition to the member areas. Member stiffness equations based on the displacement method were imposed as equality constraints. The Simplex method was used to solve the problem after move limits were specified for the linearized subproblems. It was concluded that the proposed approach made it possible to avoid the solution of equilibrium equations and the number of iterations involved was smaller. Another class of alternative formulations for optimum structural design is the so-called displacement-based two-phase procedure. Missoum *et al.* (2002) presented that procedure and applied it to optimize trusses and geometrically nonlinear framed structures. These formulations have severe limitations as noted by the authors and by Arora and Wang (2005).

In the present chapter, the SAND concept is extended to investigate use of the alternative formulations for optimum design of frames with existing analysis programs. There is a possibility to use member forces and displacements as optimization variables. These are considered as major extensions of the formulations used previously for trusses (Wang and Arora 2005). As noted earlier, Saka (1980) has previously used nodal displacements as optimization variables and solved the problem using the sequential linear programming (SLP) method. Major differences between that work and the present work are: (i) member forces are also treated as optimization variables in one of the formulations, (ii) an existing analysis software ANSYS is used directly in the optimization process, (iii) a sequential quadratic programming (SQP) algorithm is used to solve optimization problems which is more robust than the SLP method, and (iv) relative performance of the formulations is studied.

It is noted that a major objective of the current work is to develop and evaluate different optimization formulations using existing design examples with known solutions. Therefore, definitions of the design variables and constraints are taken from the published literature so that a direct comparison of the solutions can be made. It is realized that this definition of the design optimization problem has limitations because not all the design code constraints can be imposed for practical applications. However, this restricted problem definition can still be useful at the preliminary design stage of the structure. The alternative formulations can be applied to more practical frame design problems which will be considered in the future work.

5.2 Optimal Design Problem Statement

5.2.1 General problems

As noted in Chapter 4, the optimization problem is to find a design variable vector representing member sizes to minimize a cost function, which may be volume or weight of the structure subject to the design constraints that are imposed as inequality constraints on stresses, displacements and design variables.

5.2.2 Sizing variables

For a frame member, the design variables can be the cross-sectional dimensions, i.e., depth and width of a rectangular section, radius of a solid circular section, and so on. To calculate the nodal displacements and member stresses, the cross-sectional area, the moment of inertia and the section modulus of the member are required. Areas or moment of inertias of members are also popular to be chosen as primary design variables, while expressing other cross-sectional properties in terms of them by explicit nonlinear relationships. A suggested form of the relationships among the cross-sectional properties, where the member cross-sectional area A is treated as the only design variable, is (Saka 1980):

$$S = \alpha A^{\beta}\,; \quad I = \gamma A^{\phi} \tag{5.2.1}$$

where S is the section modulus and I is the moment of inertia. α and γ are constants that depend on the shape of the cross-section. β and ϕ are positive powers. By performing regression analysis of the cross-sectional data for commercially available wide-flange (W) steel sections, nonlinear relationships between the cross-sectional properties are obtained as follows (unit: inch) (Khan 1984; Sadek 1992; Sedaghati and Esmailzadeh 2003):

$$S = \begin{cases} 1.6634A^{1.511}, & 0 \le A \le 15 \\ \sqrt{281.077A^2 + 84100} - 290, & 15 < A \le 44 \\ 13.761A - 103.906, & 44 < A \le 100 \end{cases} \tag{5.2.2}$$

$$I = \begin{cases} 4.592A^2, & 0 \le A \le 15 \\ 4.638A^2, & 15 < A \le 44 \\ 256.229A - 2300, & 44 < A \le 100 \end{cases} \tag{5.2.3}$$

Although this representation of the design optimization problem is quite restrictive, it is adopted to evaluate the alternative formulations and compare solutions with the previously published results for some example problems. After the evaluation has proven usefulness of the alternative formulations, a more realistic definition of the problem will be pursued in the future work. It is also important to note that relations in Eqs. (5.2.2) and (5.2.3) are discontinuous when A = 15 and A = 44. These discontinuities can be troublesome for the gradient-based optimization methods.

5.2.3 Mathematical statement

The design optimization problem of minimizing the volume of a frame is to determine the design variables A_i, ($i = 1, n$) to

Minimize

$$f = \sum_{i=1}^{n} L_i A_i \tag{5.2.4}$$

Subject to

$$\sigma_i^L \le \sigma_i \le \sigma_i^U, \qquad i = 1, n \tag{5.2.5}$$

$$r_j^L \le r_j \le r_j^U, \qquad j = 1, m \tag{5.2.6}$$

$$A_i^L \le A_i \le A_i^U, \qquad i = 1, n \tag{5.2.7}$$

where n and m are the numbers of members and degrees of freedom for the frame, respectively. A_i and L_i are the ith member cross-sectional area and its length. σ_i^L, r_j^L, A_i^L and σ_i^U, r_j^U, A_i^U are the lower and upper bounds for the stresses σ_i, nodal displacements r_j, and design variables, respectively. To simplify the presentation, only one load case is considered; however, additional loading conditions can be treated similarly. Also, members of the structure are usually linked together into groups for symmetry and other considerations. Such linking, although not shown, can be routinely incorporated into the problem definition.

In the literature, the combined axial and bending stress equation given below has been used by different researchers (Saka 1980; Khan 1984; Sadek 1992). In order to compare solutions, it is adopted here as well

$$\sigma_i = \frac{P_i}{A_i} \pm \frac{M_i}{S_i} \qquad (5.2.8)$$

where P_i and M_i are the axial force and bending moment, respectively. S_i is the section modulus of member *i*. The combined axial and bending stress constraints are imposed at both ends of each frame member. Note that although the shear stress and other design code constraints should be included in the formulation, they cannot be imposed due to the limitations of the design variable definitions (Saka 1980; Khan 1984; Sadek 1992).

5.3 Frame Analysis

For a general frame structure, the equilibrium equation for member *i* in the global coordinate system is expressed as:

$$\mathbf{Q}_i = \mathbf{k}_i \mathbf{q}_i = \left(A_i \mathbf{k}_i^a + I_i \mathbf{k}_i^b\right)\mathbf{q}_i \qquad (5.3.1)$$

where $\mathbf{k}_i$ is the member stiffness matrix in the global coordinate system, and $\mathbf{Q}_i$ and $\mathbf{q}_i$ are nodal force (including moments) and displacement vectors (6×1) in the global coordinate system. Note that $\mathbf{k}_i = \mathbf{T}_i^T \mathbf{k}'_i \mathbf{T}_i$, where $\mathbf{k}'_i$ is the member stiffness matrix in the local coordinate system, and $\mathbf{T}_i$ is a transformation matrix between the local (member) and global coordinate systems. The member stiffness matrix $\mathbf{k}_i$ can be separated into axial and flexural parts, and $\mathbf{k}_i^a$ and $\mathbf{k}_i^b$ are 6×6 constant matrices.

Member nodal displacement vector $\mathbf{q}_i$ can be related with the global nodal displacement vector **r** for the structure by a $6 \times m$ Boolean matrix $\mathbf{Z}_i$:

$$\mathbf{q}_i = \mathbf{Z}_i \mathbf{r} \tag{5.3.2}$$

Therefore from Eqs. (5.3.1) and (5.3.2), nodal force vector for member *i* in the global coordinate system can be written as

$$\mathbf{Q}_i = \left(A_i \mathbf{k}_i^a + I_i \mathbf{k}_i^b\right) \mathbf{Z}_i \mathbf{r} = \left(A_i \mathbf{B}_i^a + I_i \mathbf{B}_i^b\right) \mathbf{r} \tag{5.3.3}$$

It is important to note that $\mathbf{B}_i^a = \mathbf{k}_i^a \mathbf{Z}_i$ and $\mathbf{B}_i^b = \mathbf{k}_i^b \mathbf{Z}_i$ are $6 \times m$ transformation matrices that are independent of A_i and I_i. They are only functions of the elastic modulus E, member length and direction cosines between the local and global coordinate systems.

The *j*th nodal force component Q_{ji} in the global coordinate system for member *i* is

$$Q_{ji} = [\mathbf{Q}_i]_j = \mathbf{e}_j^T \mathbf{Q}_i = \left(A_i \mathbf{e}_j^T \mathbf{B}_i^a + I_i \mathbf{e}_j^T \mathbf{B}_i^b\right) \mathbf{r} \tag{5.3.4}$$

where $\mathbf{e}_j$ is a 6×1 Boolean vector that is independent of A_i and I_i to determine the position of the corresponding nodal force component *j*. From Eq. (5.3.3), the axial force and the nodal bending moment of a frame member (in the local coordinate system) are given as

$$P_i = \mathbf{l}_i^T \mathbf{Q}_i = A_i \mathbf{l}_i^T \mathbf{B}_i^a \, \mathbf{r} \tag{5.3.5}$$

$$M_i = \mathbf{p}_i^T \mathbf{Q}_i = I_i \mathbf{p}_i^T \mathbf{B}_i^b \, \mathbf{r} \tag{5.3.6}$$

where $\mathbf{l}_i = \begin{bmatrix} 0 & 0 & 0 & \lambda_x & \lambda_y & 0 \end{bmatrix}_i^T$, and λ_x and λ_y are the direction cosines of member *i* with respect to the global *x* and *y* directions; $\mathbf{p}_i = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}_i^T$ or

$\mathbf{p}_i = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]_i^T$. $\mathbf{l}_i$ and $\mathbf{p}_i$ are 6×1 transformation vectors that are independent of A_i and I_i. Therefore from Eq. (5.2.8), the combined axial and bending stress for a frame member is

$$\sigma_i = \left(\mathbf{l}_i^T \mathbf{B}_i^a \pm \frac{t_i}{2} \mathbf{p}_i^T \mathbf{B}_i^b \right) \mathbf{r} \tag{5.3.7}$$

where t_i is the height of the cross-section for member *i*, related to the variables as $t_i = \dfrac{2I_i}{S_i}$.

From Eqs. (5.3.5) and (5.3.6), Eq. (5.2.8) can also be expressed directly in terms of the member force vector $\mathbf{Q}_i$, as follows:

$$\sigma_i = \frac{1}{A_i} \mathbf{l}_i^T \mathbf{Q}_i \pm \frac{1}{S_i} \mathbf{p}_i^T \mathbf{Q}_i = \left(\frac{1}{A_i} \mathbf{l}_i^T \pm \frac{1}{S_i} \mathbf{p}_i^T \right) \mathbf{Q}_i \tag{5.3.8}$$

5.4 Conventional Formulation (F-CF) – Only Areas as Optimization Variables

Only the cross-sectional areas of the frame members are considered as design variables. The design problem is to minimize the cost function of Eq. (5.2.4), subject to the constraints of Eqs. (5.2.5) to (5.2.7). This formulation uses the smallest number of optimization variables. However, the problem functions are implicit in terms of the variables, since the displacement vector $\mathbf{r}$ is an implicit function of the member cross-sectional properties. Explicit expressions for $\mathbf{r}$ and, thus, for the constraint functions in terms of the design variables cannot be obtained. The equilibrium equations need to be solved for evaluation of constraint functions. Also, gradients of the functions need to be evaluated using special sensitivity analysis procedures (Arora 1995).

5.5 Alternate Formulation 1 (F-AF1) – Areas and Nodal Displacements as Optimization Variables

If the nodal displacements $\mathbf{r}$ are also treated as independent variables in the optimization formulation, the implicit problem functions are transferred to explicit ones. In addition, the global equilibrium equations need not be solved explicitly as they are treated as equality constraints in the optimization process. Thus decomposition of the global stiffness matrix is avoided. The optimum design problem is to determine $\mathbf{A}$ and $\mathbf{r}$ to minimize the cost function defined in Eq. (5.2.4), subject to the constraints as

$$h_j^g = \sum_{k=1}^{NE_j} Q_{jk}(\mathbf{A},\mathbf{r}) - R_j = 0\,; \quad Q_{jk} = \left(A_k \mathbf{e}_j^T \mathbf{B}_k^a + I_k \mathbf{e}_j^T \mathbf{B}_k^b\right)\mathbf{r}\,, \qquad j = 1, m \qquad (5.5.1)$$

$$g_i^\sigma = \left(\mathbf{l}_i^T \mathbf{B}_i^a \pm \frac{t_i}{2} \mathbf{p}_i^T \mathbf{B}_i^b\right)\mathbf{r} - \sigma_i^U \le 0\,, \qquad i = 1, n \qquad (5.5.2)$$

where Q_{jk} is the nodal force of member *k* which has same direction as the displacement *j*. Member *k* is connected to the node which has displacement *j* as one of the degrees of freedom (in the global coordinate system). NE_j is the number of members connected to the same node. R_j is the resultant external load acting at node *p* in the direction of displacement *j*. Note here that Q_{jk} and R_j can be either forces or moments. Equation (5.5.1) in fact includes the equilibrium equation for the degree of freedom *j*. Displacement constraints in Eq. (5.2.6) become simple bound constraints. The lower bound constraint on stress can be treated similar to that in Eq. (5.5.2).

Differentiating Eqs. (5.5.1) and (5.5.2) with respect to all the variables directly, explicit expressions for derivatives can be obtained. The constraint functions and their derivatives can be easily calculated using the current nodal displacement vector $\mathbf{r}$ and cross-sectional areas $\mathbf{A}$.

5.6 Alternate Formulation 2 (F-AF2) – Areas, Nodal Displacements and Member Forces as Optimization Variables

In F-AF1, the constraint functions in Eqs. (5.5.1) and (5.5.2) need to be expressed explicitly in terms of areas and displacements. It is seen that evaluation of Eqs. (5.5.1) and (5.5.2) for the functions and their derivatives is a little tedious. However, if the member forces are also treated as variables, the constraints in Eqs. (5.5.1) and (5.5.2) become very simple. The stress in Eq. (5.5.2) can now be replaced by Eq. (5.3.8). This leads to simpler expressions and computer implementation. To evaluate this and compare efficiency, nodal forces are also treated as variables in F-AF2. The formulation is to determine $\mathbf{A}$, $\mathbf{r}$, and $\mathbf{Q}$ ($\mathbf{Q}^T = [\mathbf{Q}_1^T \quad \mathbf{Q}_2^T \quad \cdots \quad \mathbf{Q}_n^T]$) to minimize the cost in Eq. (5.2.4), subject to

$$h_j^g = \sum_{k=1}^{NE_j} Q_{jk}(\mathbf{A},\mathbf{r},\mathbf{Q}) - R_j = 0;\ Q_{jk} = \mathbf{e}_j^T\mathbf{Q}_k, \qquad j = 1, m \tag{5.6.1}$$

$$\mathbf{h}_i^e = \mathbf{Q}_i - \left(A_i\mathbf{B}_i^a + I_i\mathbf{B}_i^b\right)\mathbf{r} = \mathbf{0}, \qquad i = 1, n \tag{5.6.2}$$

$$g_i^\sigma = \left(\frac{1}{A_i}\mathbf{l}_i^T \pm \frac{1}{S_i}\mathbf{p}_i^T\right)\mathbf{Q}_i - \sigma_i^U \le 0, \qquad i = 1, n \tag{5.6.3}$$

There are 6 equality constraints for forces $\mathbf{Q}_i$ for member *i*; therefore, Eq. (5.6.2) in fact includes *6n* equality constraints.

It can be seen that once the areas $\mathbf{A}$, nodal displacements $\mathbf{r}$ and member forces

Q are available, Eqs. (5.6.1) to (5.6.3) can be evaluated directly for constraints and their derivatives. Table 5.1 summarizes the sizes of all the formulations in terms of numbers of variables and constraints (*L* is the number of loading conditions).

Table 5.1 Sizes of different formulations for frames

Item	**F-CF**	**F-AF1**	**F-AF2**
No. of Variables	n	$n+Lm$	$n+L(m+6n)$
No. of Equality Constraints	0	Lm	$L(m+6n)$
No. of Inequality Constraints	$L(m+4n)$	$4Ln$	$4Ln$
No. of Simple Bounds	n	$n+Lm$	$n+Lm$

5.7 Implementation with Existing Programs

A sequential quadratic programming (SQP) algorithm in SNOPT package is used to solve optimization problems (Gill *et al.* 2002). It is a stand-alone Windows-based program that uses text files to communicate with the commercial package ANSYS. To use the algorithm, cost and constraint functions and their gradients need to be provided. For all the formulations, ANSYS is used as a black-box and the data from its output files are used to evaluate the functions and their derivatives externally to ANSYS. A system-level command is used to execute the stand-alone program ANSYS. Sparsity of all the function gradients is exploited while using SNOPT with alternate formulations. For F-CF, member areas, moments of inertia and other data are sent to ANSYS which performs structural analysis and writes nodal displacements and member forces in the output file. In the alternative formulations, no equilibrium equations are solved as nodal displacements are also sent to ANSYS and it calculates and outputs member forces.

5.7.1 F-CF

In the conventional formulation, ANSYS is used to analyze the structure and is called again to calculate the sensitivity information by the analytical method (Arora 1995). The constraint gradients are evaluated using the direct differentiation method, as mentioned in Chapter 3. In this process, additional assembly of the global stiffness matrix and its decomposition are not needed. ANSYS is restarted with the new sensitivity load vectors to evaluate the gradients. The sensitivity load vectors are assembled external to ANSYS using data in its output file and the element matrices. Therefore, ANSYS is called twice for one evaluation of both functions and their derivatives, which makes the implementation a little tedious. In addition, each call for function evaluation during line search requires complete structural analysis. This implementation is similar to that for the trusses in Chapter 4 where more details are given.

5.7.2 F-AF1

Since matrices $\mathbf{B}_i^a$ and $\mathbf{B}_i^b$ for member i are fixed and independent of $\mathbf{A}$ and $\mathbf{r}$, the constraint functions in Eqs. (5.5.1) to (5.5.2) and their derivatives can be easily implemented using the current $\mathbf{A}$ and $\mathbf{r}$. $\mathbf{B}_i^a$ and $\mathbf{B}_i^b$ contain information about elastic modulus E, member connectivity information, such as transformation between the local and global coordinate systems, and member length. All these can be read from the ANSYS input or output file. Another possibility is to evaluate the equilibrium equality constraints in Eq. (5.5.1) and stress constraints in Eq. (5.5.2) by directly using the member forces and stresses in ANSYS output file, instead of using $\mathbf{A}$ and $\mathbf{r}$ to calculate them. Derivatives of constraint functions are evaluated external to ANSYS, using the matrices $\mathbf{B}_i^a$ and $\mathbf{B}_i^b$, and the member connectivity information. Note that since matrices $\mathbf{B}_i^a$ and $\mathbf{B}_i^b$ in Eqs. (5.5.1) to (5.5.2) are simple in form, they are programmed in user subroutines.

5.7.3 F-AF2

It is seen that once the cross-sectional areas $\mathbf{A}$, nodal displacements $\mathbf{r}$ and member nodal forces $\mathbf{Q}$ are available, Eqs. (5.6.1) to (5.6.3) can be used to evaluate the constraints and their derivatives. This requires values of the variables $\mathbf{A}$ and $\mathbf{r}$ and the matrices $\mathbf{B}_i^a$ and $\mathbf{B}_i^b$ for the *i*th member. Similar to F-AF1, the matrices $\mathbf{B}_i^a$ and $\mathbf{B}_i^b$ in Eq. (5.6.2) and its derivatives are calculated using the output data from ANSYS. In this study, the equilibrium constraints in Eqs. (5.6.1) and (5.6.2) are calculated directly using the current values of $\mathbf{A}$, $\mathbf{Q}$ and $\mathbf{r}$. However, it is viable to evaluate Eq. (5.6.2) by directly using the member forces in ANSYS output file, instead of using $\mathbf{A}$ and $\mathbf{r}$ to calculate them.

In both the alternative formulations, only the cross-sectional properties and displacements are needed by the analysis code to calculate the member level quantities, such as forces, and to calculate the constraint functions and their derivatives. ANSYS is called only once for one evaluation of both functions and their derivatives; therefore, no restart capability is needed. Basically F-AF1 and F-CF need similar calculations for gradient evaluations, except that no sensitivity analysis equations are solved in F-AF1. In F-AF2, the constraints in Eqs. (5.6.2) and (5.6.3) are both member-level calculations. Eq. (5.6.1) contains global equilibrium equations, which is in a simple sparse linear form and no assembly of global stiffness matrix is needed. The gradient calculation of functions in

Eq. (5.6.1) is performed only once in the optimization process, since they are linear in variables. Therefore, the inclusion of forces as variables provides a decoupled representation of the problem functions, which makes the implementation of F-AF2 easier than F-AF1.

5.7.4 Optimization procedure

The step-by-step optimization procedure is explained as follows:

1. Define the optimization problem, including the objective function, optimization variables and constraints. Estimate initial values of the variables.
2. The optimization code calls the user-supplied subroutines, which calculate the objective and constraint functions and their derivatives. With the current values of the optimization variables, the user-supplied subroutines further call ANSYS to obtain the internal forces and stresses for each frame member. Constraint functions and their derivatives are evaluated explicitly using the member stiffness matrices and connectivity information. During the line search, ANSYS is called again to evaluate the problem functions.
3. Optimization variables are updated and the stopping criteria are checked for optimum solution.

5.7.5 Role of ANSYS

Role of existing analysis software (ANSYS) in different formulations is elaborated here for framed structures. The analysis software provides member connectivity information and direction cosines, etc. Also if desirable, equality constraints in Eq. (5.5.1) can be formed directly using the member forces in ANSYS output file. It may seems that use of the analysis program is not necessary in the alternative formulations, since the member-level matrices for a frame member are simple and explicit, and they can be directly programmed and calculated. However, the pre-and post-processing capabilities of the existing codes are useful, especially for problems that are more complex. In those problems, it may not be possible or trivial to write the finite element matrices explicitly, and their coding may not be straightforward. This aspect will be investigated in future research, when structures that are more complex are considered, such as shells and plates.

In the conventional formulation, since the analysis program must be restarted to evaluate the displacement gradients, the program must have restart capability to make F-CF efficient; otherwise, the stiffness matrix and its decomposition will have to be re-generated which is inefficient. If the program uses iterative procedures to solve the system of equations or some other approximate procedures, then that procedure must be repeated for sensitivity analysis which can be quite time consuming. It is noted here that although ANSYS is used in the present study, it can be replaced with any other analysis program with similar capabilities.

5.8 Evaluation of Formulations

Table 5.1 lists the sizes of the three formulations. Note that the F-CF has the least numbers of variables and constraints. It is obvious that the alternative formulations increase the size of the optimization problem substantially. A major disadvantage of F-CF is that the equilibrium equations in terms of the displacements must be solved in each iteration. The same process must also be repeated during step size calculation along the search direction. Another disadvantage is that the gradient evaluation requires solution of linear equations which must be done by restarting the analysis program. The sensitivity load vectors must be assembled. This is cumbersome making the implementation of the process more tedious depending on the facilities available in the analysis program. The implementation can be facilitated if the analysis software allows call to the element level calculation directly. Also, the Jacobian and Hessian matrices of the constraints are dense in this formulation, limiting the size of the design problem that can be treated efficiently.

Table 5.2 Comparison of alternative SAND formulations

	F-AF1	**F-AF2**
Optimization Variable	$\mathbf{A},\mathbf{r}$	$\mathbf{A},\mathbf{r},\mathbf{Q}$
Equilibrium Constraints	$\sum_{k=1}^{NE_j}\left(A_k\mathbf{e}_j^T\mathbf{B}_k^a + I_k\mathbf{e}_j^T\mathbf{B}_k^b\right)\mathbf{r} = R_j$ Assembly of global stiffness matrix needed	$\sum_{k=1}^{NE_j}\mathbf{e}_j^T\mathbf{Q}_k = R_j$ Linear constraints; No assembly of global stiffness matrix
Member Equilibrium Constraints	--	$\left(A_i\mathbf{B}_i^a + I_i\mathbf{B}_i^b\right)\mathbf{r} = \mathbf{Q}_i$
Stress Constraints	$\left(\mathbf{l}_i^T\mathbf{B}_i^a \pm \frac{t_i}{2}\mathbf{p}_i^T\mathbf{B}_i^b\right)\mathbf{r} \le \sigma_i^U$	$\left(\frac{1}{A_i}\mathbf{l}_i^T \pm \frac{1}{S_i}\mathbf{p}_i^T\right)\mathbf{Q}_i \le \sigma_i^U$
Displacement Constraints	$r_j^L \le r_j \le r_j^U$ Simple bound constraints	$r_j^L \le r_j \le r_j^U$ Simple bound constraints
Advantages	1. Fewer optimization variables.	1. The assembly of the global stiffness matrix and its derivatives are avoided. 2. Very sparse Jacobian and Hessian matrices. 3. Implementation with existing programs is straightforward.
Disadvantages	1. The assembly of the global stiffness matrix and its derivatives are needed. 2. Derivative calculation and implementation is more tedious. 3. Denser Jacobian matrices.	1. Larger numbers of variables and constraints.

Advantages and disadvantages of the two alternative formulations are summarized in Table 5.2. F-AF1 is more suitable for problems that do not involve stress constraints. In that case, only the equilibrium constraints in Eq. (5.5.1) are included as behavior constraints and the displacement constraints are simple bounds on the variables. In F-AF2 with forces also as variables, the optimization problem is larger with more variables and constraints. However, the assembly of the global stiffness matrix and its derivatives are avoided, which is the major difference between F-AF2 and the previous two formulations. Only member-level calculations are required in the constraints in Eqs. (5.6.2) and (5.6.3). Note that the equilibrium constraints in Eq. (5.6.1) are linear and the stress constraints in Eq. (5.6.3) are simpler. The gradients of the linear constraints in Eq. (5.6.1) are programmed independently and calculated only once in the solution process. Obviously it is not necessary to include member nodal forces as variables in the formulations to obtain explicit expressions for the constraints; however, their inclusion simplifies the form of constraints and Jacobians, and hence the numerical implementations. It is also noted that the alternative formulations require study of the sparsity structure of the Jacobian matrices so as to take full advantage of the sparse matrix operations in SNOPT.

5.9 Numerical Examples

To evaluate the formulations, two framed structures are optimized. Each structural member is treated as one finite element. The material used for both examples is the same: steel with modulus of elasticity E = 206,844 MPa and the combined axial and bending stress limit for all members as 165.4752 MPa. A PC with 2.5 GHz processor and 1 GB RAM is used for running the programs and recording the relative CPU times. ANSYS resides on a local area network. Sparsity of the constraints in Eqs. (5.5.1) to (5.6.3) and their gradients is implemented in SNOPT. The lower and upper limits on cross-sectional areas are 0.0032258 and 0.064516 m^2, respectively. Very tight stopping criteria are used in SNOPT to obtain precise optimal solutions. The final optimal volumes and CPU efforts for design examples are listed in Table 5.3, and the numbers of iterations and calls to the analysis program are given in Table 5.4.

5.9.1 Example 1 – 10-member frame

This 10-member rigid frame example is taken from Khan (1984). Only one loading condition is imposed, and no variable linking is used; therefore ten cross-sectional areas are the design variables. The horizontal displacements of the free nodes are limited to ± 0.00254 m. The initial areas are taken as 0.0129032 m^2 for all the formulations. The initial displacements are taken as 0.00254 m, and the rotations are 10^{-3}, respectively. The initial force variables before scaling are taken as unity. The final optimal solutions for the three formulations are similar, i.e., 0.422229 m^3. At the optimum, no stress constraint is active while the horizontal displacements of nodes 3 and 4 are active. The optimal areas are: 0.028387, 0.023364, 0.003226, 0.003226, 0.046499, 0.010240, 0.007217, 0.016479, 0.016286 and 0.003226 m^3. Due to discontinuity of the relationships in Eqs. (5.2.2) and (5.3.3), very accurate optimality conditions could not be satisfied; however, the final areas represent the best feasible solution. It is noted that the optimal design found in this study is better than the three solutions 0.433002, 0.506799 and 0.433002 m^3 obtained by Khan (1984). This is due to the use of a more robust optimization algorithm in the present study.

5.9.2 Example 2 – 25-member frame

The 25-member framed structure has been designed by Khan *et al.* (1978), Khan (1984), and Sadek (1980). All members are 2.54 m long, except for the diagonal ones.

The diagonal members are treated as frame members. Only one loading condition is considered and variable linking is not used. Three design cases are considered.

Case 1: The external loads are $P_1 = P_2 = P_3 = 444.82$ kN, $P_4 = P_7 = 8896.4$ kN, $P_5 = P_6 = 13344.6$ kN, $M_1 = M_2 = 2259.6856$ kN-m, and $M_3 = M_4 = 3389.5284$ kN-m. The section modulus *S* and the moment of inertia *I* are related to area *A* as *S = 9A* and *I = 75A*. Nodal displacement limits of ± 0.0762 m are imposed at nodes 1-3 and 10-12 in both *x* and *y* directions.

Case 2: Same as Case 1, except the displacement limits of ± 0.00127 m at nodes 1-3 and 10-12 in the *x* and *y* directions are imposed. For Cases 1 and 2, only the lower limit of 0.0032258 m^2 on the cross-sectional areas is imposed.

Case 3: The external loads are $P_1 = P_2 = P_3 = 444.82$ kN, $P_4 = P_7 = 2224.1$ kN, $P_5 = P_6 = 3113.74$ kN and $M_i = 0\,(i = 1 \text{ to } 4)$. The nonlinear relationships between *S*, *I* and *A* given in Eqs. (5.2.2) and (5.2.3) are used. The displacements of nodes 1-3 and 10-12 in the *x* and *y* directions are limited to ± 0.00127 m.

A uniform initial design of 0.064516 m^2 is used for all the formulations. The initial displacements are taken as 0.00254 m, and the rotations as 10^{-3}. The initial force variables in F-AF2 before scaling are taken as unity. The final solution for Case 1, 3.070543 m^3, is quite similar to that in Khan *et al.* (1978). For Case 2, the solution is 6.758287 m^3, which is better than 7.595781 m^3 reported by Khan *et al.* (1978). For Case 3, it is seen that a better local optimum has been found with the alternative formulations. Both F-AF1 and F-AF2 gave the same solution as 1.213809 m^3. The F-CF stopped at a feasible solution of 1.243110 m^3, because a very tight stopping criterion could not be satisfied. These solutions are better than those available in the literature; Khan (1984) reported 1.282928 and 1.281364 m^3, and Sadek (1992) reported 1.270551m^3. It is seen that different techniques obtain different final designs. The local optimum obtained with the alternative formulations is the best one. The SQP method in SNOPT led to better optimal solutions. The final cross-sectional areas for all cases are listed in Table 5.5.

Table 5.3 Final optimal volumes and computing efforts for design examples

Example Problem		**Final Optimal Volumes (m^3)**			**CPU (s)**		
		F-CF	**F-AF1**	**F-AF2**	**F-CF**	**F-AF1**	**F-AF2**
10-member		0.422229	0.422229	0.422229	14.8	12.9	29.8
25-member	**Case 1**	3.070543	3.070543	3.070543	1.4	2.0	2.5
	Case 2	6.758287	6.758287	6.758287	3.4	4.5	6.3
	Case 3	1.243110	1.213809	1.213809	38.7	8.5	13.8

Table 5.4 Numbers of iterations and calls to ANSYS for design examples

Example Problem		**Numbers of Iterations**			**Numbers of Calls to ANSYS**		
		F-CF	**F-AF1**	**F-AF2**	**F-CF**	**F-AF1**	**F-AF2**
10-member		27	58	105	145+143*	221	435
25-member	**Case 1**	9	13	16	12+10	22	26
	Case 2	17	38	53	22+20	55	62
	Case 3	71	57	65	320+318	90	93

*Analysis + Sensitivity Analysis

5.9.3 Solutions with modified cross-sectional relationships

Since that relations in Eqs. (5.2.2) and (5.2.3) are discontinuous when $A = 15$ and $A = 44$, the gradient-based optimization methods have difficulty to converge to optimum solutions accurately. These equations are modified by adjusting constants in them so that the relationships are continuous; for *S:* $S = 1.6634A^{1.511}\ (0 \le A \le 15)$, $S = \sqrt{281.077A^2 + 84100} - 284.2952 \quad (15 < A \le 44)$, and $S = 13.761A - 97.1475$ $(44 < A \le 100)$; for *I:* $I = 4.592A^2\ (0 \le A \le 15)$, $I = 4.638A^2 - 10.35\ (15 < A \le 44)$, and $I = 256.229A - 2305.258\ (44 < A \le 100)$. Solutions with the modified cross-sectional relationships are listed in Tables 5.6 and 5.7. The convergence properties were improved, but due to the discontinuity in the gradient expressions, the convergence still cannot be guaranteed. The reported solutions, however, are feasible. The two alternative formulations were more efficient than the conventional formulation.

Table 5.5 Example 2 – Optimal areas for 25-member frame (m^2)

No.	Case 1	Case 2	Case 3		No.	Case 1	Case 2	Case 3	
			F-CF	F-AF1 / F-AF2				F-CF	F-AF1 / F-AF2
1	0.083259	0.298308	0.018570	0.010006	**14**	0.079073	0.085511	0.017158	0.019580
2	0.097819	0.300910	0.013680	0.007146	**15**	0.003226	0.008082	0.009037	0.003226
3	0.097719	0.278152	0.011244	0.004233	**16**	0.034721	0.035142	0.003226	0.004829
4	0.019824	0.014977	0.003226	0.016210	**17**	0.122539	0.316319	0.012804	0.013152
5	0.085233	0.100238	0.015924	0.020245	**18**	0.003226	0.014882	0.028387	0.003226
6	0.003226	0.003226	0.003226	0.003226	**19**	0.078939	0.342469	0.039583	0.016162
7	0.084281	0.113006	0.022680	0.051749	**20**	0.003226	0.034681	0.046984	0.003226
8	0.015009	0.005097	0.028387	0.014879	**21**	0.076971	0.361973	0.064516	0.018748
9	0.109497	0.120205	0.031837	0.031429	**22**	0.003888	0.012421	0.011890	0.013545
10	0.003226	0.003226	0.003226	0.053870	**23**	0.003226	0.003226	0.009008	0.046763
11	0.077104	0.064840	0.016409	0.064516	**24**	0.003226	0.003226	0.003226	0.003226
12	0.071192	0.081542	0.030904	0.020581	**25**	0.029501	0.033969	0.009083	0.003226
13	0.003226	0.003226	0.021445	0.003226					

Table 5.6 Final optimal volumes and computing efforts with modified cross-sectional relationships

Example Problem		Final Optimal Volumes (m^3)			CPU (s)		
		F-CF	F-AF1	F-AF2	F-CF	F-AF1	F-AF2
10-member		0.438365	0.438365	0.438359	16.2	7.0	6.8
25-member	**Case 3**	1.238234	1.208697	1.208697	43.9	10.4	8.3

Table 5.7 Numbers of iterations and calls to ANSYS with modified cross-sectional relationships

Example Problem		Numbers of Iterations			Numbers of Calls to ANSYS		
		F-CF	F-AF1	F-AF2	F-CF	F-AF1	F-AF2
10-member		47	44	51	177+175*	125	109
25-member	**Case 3**	73	66	44	367+365	122	83

*Analysis + Sensitivity Analysis

5.9.4 Discussions of results

Two alternative formulations for optimal design of framed structures based on simultaneous analysis and design (SAND) concept are presented and implemented with a commercial analysis code ANSYS for evaluation. Advantages of the formulations are that no structural analyses and sensitivity analyses are needed. Since all functions become explicit in terms of the optimization variables, derivatives of the functions with respect to the variables are computed explicitly and accurately. However, there are more optimization variables and equality constraints in the two alternative formulations, which requires careful analysis and use of the sparsity structure of the problem functions for efficiency of calculations. Both the formulations worked quite well, and in most cases converged to better solutions than those reported in the literature.

1. Computational effort

It is seen from Table 5.4 that the alternative formulations generally take more iterations to find optimal solutions. This is reasonable since there are more optimization variables and constraints. The CPU times of all the examples are also reported in Table 5.4. It is seen that the computational efforts with all the formulations for the first example and Cases 1 and 2 of the second example are quite similar, with the conventional formulation having an edge. However, for Case 3, the alternative formulations are more efficient than the conventional formulation. Among the alternative formulations, F-AF2 generally requires more CPU effort compared to F-AF1. Results with the modified cross-sectional relationships show the alternative formulations to be more efficient than the F-CF (Table 5.7). Also F-AF2 is more efficient than F-AF1. Since these are small-scale examples, general conclusions about relative efficiency of the formulations cannot be drawn. However, it is clear that the alternative formulations work quite well and converge to either similar or better optimal solutions.

2. Scaling of variables and constraints

An important point to note here is that the alternative formulations include different types of variables, which have different orders of magnitudes. Therefore scaling of some of the variables is necessary to reduce numerical difficulties. SNOPT has two options for automatic scaling: one for linear constraints and variables, and the other for all constraints and variables. In addition, manual scaling of the displacement and force variables is used. R_j values are used to normalize the equality constraint in Eqs. (5.5.1) and (5.6.1). Normalization of the constraint functions helps in reducing numerical ill-conditioning.

3. Role of analysis programs

As a result of this research, the role of existing analysis programs has become clearer with the SAND type formulations. Basically, the pre-and post-processing capabilities of the analysis programs can be used directly to evaluate the constraint functions. Evaluation of gradients, however, requires member connectivity information and member stiffness matrices, and needs to be implemented outside the analysis program. These implementations can become easier if the internal member level subroutines and processes of the analysis program are accessible directly through the optimization program. It was possible to implement the frame element quantities explicitly in the user subroutines to evaluate gradients of the constraints.

4. Code-specified strength constraints

In design codes, such as AISC-LRFD, requirements on member stresses or buckling are expressed directly in terms of forces (AISC 2001). Therefore, inclusion of member forces as optimization variables may be more convenient for design optimization of practical framed structures.

5. Other formulations

Note that other alternative formulations are also possible. Like trusses, where axial forces and stresses are also treated as optimization variables, axial, shear forces and bending moments in the member local coordinate system, instead of $\mathbf{Q}_i$ in Eq. (5.3.3) in the global coordinate system, can be treated as variables. This is similar to the F-AF2. It is also possible to include stresses as independent variables. However, evaluation of stresses, e.g., the shear and bending stresses, needs cross-sectional dimensions, thus requiring different definition of design variables. Moreover, stresses, such as the bending stress in a beam-column, are usually not constant throughout the member. Therefore this formulation may not be as attractive as the one with nodal forces as variables.

It is also important to note that by introducing more variables and constraints in the alternative formulations, more information from optimization process is obtained, which may be useful in the design process. For example, the Lagrange multipliers for the equilibrium constraints at the optimum point can be used to study the benefit of relaxing the constraints, or the penalty associated with tightening them (Arora 2004). Important information on how the external load distribution may affect the optimal design can be estimated. This can provide design guidance to engineers at the preliminary design stage.

5.10 Summary

Two alternative SAND formulations (F-AF1 and F-AF2) and the conventional formulation (F-CF) for framed structures were presented, analyzed and implemented with a commercial analysis code ANSYS. The second alternate formulation where member forces are also treated as optimization variables has not been presented previously in the literature. Implementation details for the optimization process were presented. Numerical example problems were solved to study performance of the formulations. Such comparative evaluation of the formulations has not been carried out previously in the literature for framed structures. Based on the present study, following conclusions are drawn:

1. Both F-AF1 and F-AF2 do not explicitly solve the global equilibrium equations. F-AF2 does not need to form these equations in terms of displacements; however, F-AF1 does need to form them.
2. Alternative formulations have large numbers of variables and constraints. Since all the problem functions are quite sparse, these formulations must exploit this sparsity in the optimization process for numerical efficiency and for solution of larger scale structures.
3. Both F-AF1 and F-AF2 work quite well and for most example problems considered in this chapter, better optimal solutions were obtained than those reported in the literature. For some cases, the alternative formulations were

more efficient than the CF, especially with the modified definition of design variables. Alternative formulations have potential for practical applications; however, they need to be developed further with more realistic definition of the design variables for framed structures.

4. The alternative formulations are easier to implement with the existing analysis programs than the F-CF.
5. For efficient calculations with F-CF, the existing analysis program must have restart capability for gradient evaluation.
6. The design variable representation used in this chapter and the cited literature is not appropriate for practical design optimization of framed structures.

CHAPTER 6
APPLICATION TO LARGE-SCALE DESIGN PROBLEMS

6.1 Introduction

Three alternative formulations for optimal design of linearly elastic trusses based on SAND concept have been studied, analyzed and evaluated in Chapter 4. The formulations involved combinations of displacements, member forces and member stresses as optimization variables. By introducing more variables into the formulations, the forms of the constraints and their derivatives were changed. Since all functions of the formulations were explicit in terms of the optimization variables, special design sensitivity analysis procedures were no longer needed. Advantages and disadvantages of the formulations were delineated. Although the alternative formulations worked well, the application was limited to small-scale truss structures.

In the present chapter, application of the three SAND formulations given in Chapter 4 is extended to large-scale truss structures, and major issues such as sparsity, storage and implementation of the formulations, are discussed. In large-scale applications, gradients of the functions and Hessian of the Lagrangian are quite sparsely populated; therefore data storage and manipulation becomes important for numerical calculations. Matrix sparsity must be utilized to reduce the storage requirement and improve efficiency of the solution process. A more recent sparse SQP algorithm and associated software are used to take full advantage of the sparsity structure of the alternative formulations. The formulations are evaluated using two relatively large-scale truss problems (Wang and Arora 2006a).

6.2 Sparsity and Storage Requirements

Here we analyze the sparsity of the Jacobians and the Hessians of various functions, and discuss their data storage requirements. Note that the total storage requirement depends on the optimization algorithm used; however, here only the storage requirements for the Jacobian and the Hessian matrices are presented. The following symbols are used: n = number of members; m = number of degrees of freedom of the structure; L = number of loading conditions; d = number of degrees of freedom per member (for plane trusses, $d \le 4$; for space trusses $d \le 6$; for large-scale trusses where there are only a few supports, 4 or 6 is a good approximation for d). For simplicity of presentation, no variable linking is assumed.

6.2.1 T-CF

For T-CF, the gradient $(n \times 1)$ of the cost function (4.3.1), Jacobian matrices ($n \times nL$ and $n \times mL$) of the stress constraints (4.3.2) and displacement constraints (4.3.3), and the Hessian of the Lagrangian $(n \times n)$ are all fully populated.

6.2.2 T-AF1

In T-AF1, the optimization variable vector is $\mathbf{x} = \begin{bmatrix} \mathbf{A}^T & \mathbf{r}^T \end{bmatrix}^T$, where $\mathbf{r}$ $(mL \times 1)$ is the displacement vector for all the loading cases. For the objective function in Eq. (4.3.1), the gradient vector $\frac{\partial f}{\partial \mathbf{x}}$ $((n + mL) \times 1)$ has n non-zero elements, since $\frac{\partial f}{\partial \mathbf{r}} = \mathbf{0}$.

For the equality constraints in Eq. (4.4.1), $\frac{\partial \mathbf{h}^g}{\partial \mathbf{x}} (n + mL) \times mL$ is sparse. For a nodal degree of freedom, only a few members are connected to it. There are approximately *dnL* non-zero elements in $\frac{\partial \mathbf{h}^g}{\partial \mathbf{A}}$. In $\frac{\partial \mathbf{h}^g}{\partial \mathbf{r}}$, the number of non-zero elements can be up to d^2nL. For the inequality constraints in Eq. (4.4.2), $\frac{\partial \mathbf{g}^\sigma}{\partial \mathbf{A}} = \mathbf{0}$, since the member stresses do not explicitly depend on the member cross-sectional areas, $\mathbf{A}$. The maximum number of nodal displacements connected to one member is only six. Therefore, if the member connectivity information of a large-scale truss structure is considered, the number of non-zero elements in $\frac{\partial \mathbf{g}^\sigma}{\partial \mathbf{r}}$ is up to *dnL*.

It is obvious that the Hessians of the objective function (4.3.1), stress constraints (4.4.2), and the simple bound constraints are all null. The equilibrium constraints in Eq. (4.4.1) are bilinear forms of $\mathbf{A}$ and $\mathbf{r}$; therefore, $\left[\frac{\partial^2 h^g}{\partial \mathbf{A}\partial \mathbf{A}}\right]_{n\times n} = \mathbf{0}$, $\left[\frac{\partial^2 h^g}{\partial \mathbf{r}\partial \mathbf{r}}\right]_{mL\times mL} = \mathbf{0}$, and $\left[\frac{\partial^2 h^g}{\partial \mathbf{A}\partial \mathbf{r}}\right]_{n\times mL}$ is quite sparse. The numbers of non-zero elements in $\left[\frac{\partial^2 h^g}{\partial \mathbf{A}\partial \mathbf{r}}\right]$ and $\mathbf{H}$, the Hessian of the Lagrangian, are estimated as *dnL* and *2dnL*, respectively.

6.2.3 T-AF2

For the objective function in Eq. (4.3.1), the gradient vector $\frac{\partial f}{\partial \mathbf{x}}$ $\left((n + mL + nL)\times 1\right)$ contains n non-zero elements since $\frac{\partial f}{\partial \mathbf{r}} = \frac{\partial f}{\partial \mathbf{F}} = \mathbf{0}$, where $\mathbf{x} = \left[\mathbf{A}^T \quad \mathbf{r}^T \quad \mathbf{F}^T\right]^T$, and $\mathbf{r}$ $(mL\times 1)$ and $\mathbf{F}$ $(nL\times 1)$ are the displacement and axial force vectors for all the loading cases, respectively.

For the equality constraints in Eq. (4.5.4), $\frac{\partial \mathbf{h}^g}{\partial \mathbf{A}} = \mathbf{0}$, $\frac{\partial \mathbf{h}^g}{\partial \mathbf{r}} = \mathbf{0}$, and $\frac{\partial \mathbf{h}^g}{\partial \mathbf{F}}$ is sparse, since the global equilibrium of forces for one degree of freedom is only related to the internal forces of members connected to that degree of freedom. The upper bound on the number for non-zero elements is estimated as *dnL*. For the equality constraints in Eq. (4.5.2), $\frac{\partial \mathbf{h}^e}{\partial \mathbf{A}}$ and $\frac{\partial \mathbf{h}^e}{\partial \mathbf{F}}$ are diagonal matrices, and the numbers of non-zero elements are *nL*. $\frac{\partial \mathbf{h}^e}{\partial \mathbf{r}}$ is also quite sparse, with up to *dnL* non-zero elements. The stress constraints in Eq. (4.5.3) can be analyzed similarly. Since the stresses in Eq. (4.5.3) are expressed in terms of areas and internal forces of structural members, $\frac{\partial \mathbf{g}^\sigma}{\partial \mathbf{r}} = \mathbf{0}$, and $\frac{\partial \mathbf{g}^\sigma}{\partial \mathbf{A}}$ and $\frac{\partial \mathbf{g}^\sigma}{\partial \mathbf{F}}$ are diagonal matrices (*nL* non-zero elements).

The Hessians of the objective and the bound constraints are null. Also, the equilibrium constraints in Eq. (4.5.4), and the stress constraints in Eq. (4.5.3) are only linear in terms of variables $\mathbf{F}$, or $\mathbf{A}$ and $\mathbf{F}$, therefore, their Hessians are null. The equality constraints in Eq. (4.5.2) are linear in $\mathbf{F}$, and bilinear in $\mathbf{A}$ and $\mathbf{r}$; therefore, the number of non-zero elements in $\left[\frac{\partial^2 h^e}{\partial \mathbf{A}\partial \mathbf{r}}\right]$ is up to *dnL*, while all other sub-matrices are null. There are up to *2dnL* non-zero elements in $\mathbf{H}$, the Hessian of the Lagrangian.

6.2.4 T-AF3

Similar to T-AF2, there are *n* non-zero elements in $\frac{\partial f}{\partial \mathbf{x}}$ $((n + mL + nL)\times 1)$. Note that $\mathbf{x} = \begin{bmatrix}\mathbf{A}^T & \mathbf{r}^T & \boldsymbol{\sigma}^T\end{bmatrix}^T$, and $\boldsymbol{\sigma}$ $(nL\times 1)$ is the axial stress vector for all the loading cases.

Since Eq. (4.6.3) does not depend explicitly on the nodal displacements,

$\frac{\partial \mathbf{h}^g}{\partial \mathbf{r}} = \mathbf{0}$. There are up to *dnL* non-zero elements in $\frac{\partial \mathbf{h}^g}{\partial \mathbf{A}}$ and $\frac{\partial \mathbf{h}^g}{\partial \boldsymbol{\sigma}}$, respectively. For the equality constraints in Eq. (4.6.2), $\frac{\partial \mathbf{h}^e}{\partial \mathbf{A}} = \mathbf{0}$ and $\frac{\partial \mathbf{h}^e}{\partial \boldsymbol{\sigma}}$ is a diagonal matrix having *nL* non-zero elements. The number of nonzero elements in $\frac{\partial \mathbf{h}^e}{\partial \mathbf{r}}$ can be approximated as *dnL*.

Since the equality constraints in Eq. (4.6.2) are linear in terms of the variables $\mathbf{r}$ and $\boldsymbol{\sigma}$, its Hessian is null. Since the equality constraints in Eq. (4.6.3) are bilinear in $\mathbf{A}$ and $\boldsymbol{\sigma}$, all sub-matrices are null except $\left[\frac{\partial^2 h^g}{\partial \mathbf{A} \partial \boldsymbol{\sigma}}\right]_{n \times nL}$, which has *nL* non-zero elements; therefore, there are up to *2nL* non-zero elements in $\mathbf{H}$.

Note that there are no inequality behavior constraints in the T-AF3, since the stress constraints are simple bound constraints. Table 6.1 lists the approximated numbers of non-zero elements in the gradient vector, and the Jacobian and Hessian matrices for all the four formulations. Note that the numbers of nonzero elements given in Table 6.1 are the upper bound estimates.

For large-scale truss structures, it is interesting to note that the number of non-zero elements in the Jacobians of constraints and Hessian of the Lagrangian in the conventional formulation can be larger than those in the three alternative formulations. T-AF1 has a larger number of non-zero elements in the Jacobian than T-AF2 and T-AF3. Also the numbers of non-zero elements in the Jacobian in T-AF2 and T-AF3 are similar. If sparsity of matrices is not considered, it is obvious that all the alternative formulations have much larger numbers of elements in the Jacobians than the conventional formulation, which means that they require much larger storage and computations than the conventional one. It is also seen that the numbers of non-zero elements in the Hessian of the Lagrangian function for all the alternative formulations depend on their numbers of variables, and therefore L, the number of loading cases. For large-scale structures where there are only a few loading cases, the conventional formulation needs more storage for the Hessian than the alternative formulations.

Table 6.1 Approximate numbers of non-zero elements in gradient vector, Jacobian and Hessian for different formulations

Item		**Formulation**			
		T-CF	**T-AF1**	**T-AF2**	**T-AF3**
Gradient Vector	**Objective Function (4.3.1)**	n	n $[n+mL]$*	n $[n+mL+nL]$	n $[n+mL+nL]$
Jacobian of Constraints	**Equilibrium Constraints (4.4.1), (4.5.4) or (4.6.3)**	-	$(d^2+d)nL$ $[(n+mL)mL]$	dnL $[(n+mL+nL)mL]$	$2dnL$ $[(n+mL+nL)mL]$
	Equality Constraints (4.5.2) or (4.6.2)	-	-	$(d+2)nL$ $[(n+mL+nL)nL]$	$(d+1)nL$ $[(n+mL+nL)nL]$
	Stress Constraints (4.3.2), (4.4.2) or (4.5.3)	n^2L	dnL $[(n+mL)nL]$	$4nL$ $[(n+mL+nL)2nL]$	-
	Displacement Constraints (4.3.3)	mnL	-	-	-
Total of 1st Order Derivatives (Gradient Vector & Jacobian)		$n(nL+mL+1)$	$(d^2+2d)nL+n$ $[(n+mL)(nL+mL+1)]$	$(2d+6)nL+n$ $[(n+mL+nL)(3nL+mL+1)]$	$(3d+1)nL+n$ $[(n+mL+nL)(nL+mL+1)]$
Hessian of Lagrangian		n^2	$2dnL$ $[(n+mL)^2]$	$2dnL$ $[(n+mL+nL)^2]$	$2nL$ $[(n+mL+nL)^2]$

* The expressions in the brackets give the total number of elements when sparsity is not considered.

From Table 6.1, it can also be seen that the Jacobians and Hessians of the alternative formulations are quite sparse; the ratios for the numbers of non-zero elements with sparsity and without sparsity are indeed quite small. Note that for other finite elements, such as beams and plates, similar expressions can be readily derived.

6.3 Implementation

A suitable optimization method is needed to solve problems of the four formulations. In this chapter, an available sparse SQP method in SNOPT (Gill *et al.* 2002) is used to solve alternative formulations and the dense SQP method is used to solve the conventional formulation. To use the algorithm, cost and constraint functions and their gradients need to be calculated. For the alternative formulations, no system of equations is solved to determine structural response. Instead, the constraint functions and their derivatives are evaluated explicitly external to the analysis software ANSYS (2002) using the member stiffness matrices and the member connectivity information. More details of implementation with ANSYS are presented in Chapter 4 and Wang and Arora (2005a).

6.3.1 Optimization algorithm for sparse problems

SNOPT is a general-purpose system for solving large-scale nonlinear programming problems involving many variables and constraints. It minimizes a linear or nonlinear function subject to bounds on the variables and sparse linear or nonlinear constraints.

1. The SQP iteration

SNOPT uses an SQP algorithm that calculates the search direction by solving a quadratic programming (QP) subproblem. Each QP subproblem minimizes a quadratic model of the Lagrangian function subject to linearized constraints. An augmented Lagrangian merit function is reduced along the search direction to ensure convergence from any starting point. The SQP iterations are called major iterations, and they generate a sequence of iterates that converges to a local minimum point.

2. The QP solver

The solution of each QP subproblem is in itself an iterative procedure; iterations within the QP solver are called minor iterations. The QP subproblems are solved using a reduced-Hessian active-set method that takes advantage of the variables appearing linearly in the objective and constraint functions. A null-space method is employed, and a Cholesky factorization of the reduced Hessian is maintained.

3. Large-scale Hessians

If the number of variables is not too large, SNOPT treats the Hessian as a dense matrix and applies the BFGS quasi-Newton updates explicitly with safeguards to maintain a positive definite Hessian approximation. Otherwise, SNOPT uses a limited-memory procedure to update an initial Hessian of the Lagrangian a limited number of times only. The method exploits sparsity in the constraint Jacobian, which is specified by providing the values and positions of non-zero entries. This means that the program requires only the first order information about the problem functions. Although it is not difficult to provide second order derivatives for the alternative formulations, their calculation adds more computational effort and makes the implementation more complex. Thus SNOPT is a good choice for large-scale problems. If exact Hessians are provided, other powerful large-scale sparse NLP codes can also be used, such as the SPRNLP in SOCS package (Betts and Frank 1994).

4. Memory requirement

As noted earlier, the storage requirement of a numerical code also depends on the algorithm used. Unfortunately, it is not easy to precisely quantify SNOPT's memory requirements. SNOPT computes and updates a sparse LU factorization in which the fill-ins in the factors depend upon the current basis matrix, which, in turn, depends on the path taken to the solution. When the basis is factorized, the LU factors are stored at the beginning of large integer and real work arrays. The amount of storage depends on the sparsity pattern of the current basis. In subsequent iterations, as the factors are updated, additional fill-in is added at the end of the work arrays. A re-factorization causes the workspace to be reset to the amount needed for the new factors. Therefore, in SNOPT only a minimum of workspace is required for input. The amount of memory for the LU factors does depend on the sparsity of the constraint Jacobian. Generally speaking, the sparser the Jacobian, the smaller is the memory used by the LU factorization (Gill 2005).

6.3.2 Overall procedure

The detailed solution process starts with the definition of objective function, optimization variables and constraints. Initial values are assigned to the variables. If it is the first iteration, the Jacobian of linear constraints is calculated and stored. Since these are constants, they only need to be calculated once. The Jacobian of the nonlinear constraints is calculated. After obtaining all the functions and their derivatives explicitly at the current optimization variables, a sparse SQP algorithm is used to perform one iteration and compute a new point. Note here that the line search calls the finite element code again to evaluate nonlinear functions.

Note that the sparse SQP code separates the linear and nonlinear constraints, and this feature is considered in calculations. The code also gives the user the option to define linear parts of both the objective function as well as the constraints. In the three alternative formulations presented here, the objective function is a linear function of the variables. There are a lot of linear constraints in the alternative formulations, such as Eqs. (4.4.2), (4.5.3), (4.5.4) and (4.6.2). Therefore, the Jacobian of these constraints is calculated only once. During line search in the SQP algorithm, the finite element code is used to evaluate the nonlinear constraint functions only. The Jacobian of linear constraints is used directly to calculate the linear function.

6.4 Numerical Examples

Two relatively large-scale truss structures are optimized using the conventional and the three alternative formulations. The structural material is steel with Young's modulus E = 68,948 MPa. The allowable stress for each member is 172.375 MPa. Multiple loading conditions are considered. No special techniques are used to obtain an initial design. In all the formulations, the initial cross-sectional areas are 6.4516E-4 m^2 and the initial displacements are 0.0127 m. In T-AF2 and T-AF3, the initial force and stress variables are taken as one. A PC with 2.5 GHz processor and 1 GB RAM is used for running the programs and recording relative CPU times. Very severe stopping criteria are used to obtain precise optimal solutions. With relaxed stopping criteria, smaller computational effort is needed, and solutions close to the true solutions are obtained.

6.4.1 Example 1 – 35-story space tower

This 35-story space tower consists of 1262 members and 936 degrees of freedom (Adeli and Cheng 1994; Adeli and Soegiarso 1999). The base of the structure is in the *X*-*Y* plane, as shown in Figure 6.1. The entire structure consists of three different sections from the top to the bottom. Seventy-two design variables are used to represent the cross-sectional areas of 72 member groups, employing symmetry of the structure. The 72 member groups are given in Adeli and Cheng (1994). The lower and upper bounds on the cross-sectional areas are 6.4516E-4 m^2 and 6.4516E-2 m^2. The loading on the structure consists of downward vertical loads and horizontal loads, as follows:

A. The vertical loads are given as 13.3446 kN at each node in the first section, 26.6892 kN at each node in the second section, and 40.0338 kN at each node in the third section.

B. The horizontal loads are given as 6.6723 kN in the *X* direction at each node on the left side, 4.4482 kN in the *X* direction at each node on the right side.

C. The horizontal loads are given as 4.4482 kN in the *Y* direction at each node on the back side, 4.4482 kN in the *Y* direction at each node on the front side.

The displacement constraints are 0.508 m in the *X*, *Y* and *Z* directions for the four nodes on the top level (about 1/250 of the height). In Case 1, a combination of all the loads A, B and C is applied to the structure. In Case 2, four loading conditions are considered, which consist of different combination of the lateral loads and vertical loads acting on the structure:

1. Loading condition A alone.
2. Loading conditions A and B acting together.
3. Loading conditions A and C acting together.
4. Loading conditions A, B and C acting together

All formulations converge to nearly the same optimum solution (52.0555 m^3) for Case 1. The final optimal values for all the cross-sectional areas are given in Wang and Arora (2006a). Note that Case 1 and Case 2 have the same optimal solutions, which implies that the combination of all the loads A, B and C acting together in fact is the dominant loading condition in Case 2.

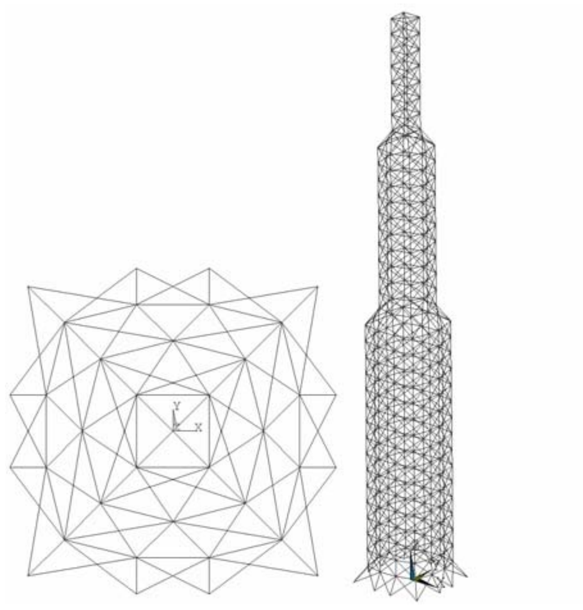

(a) (b)

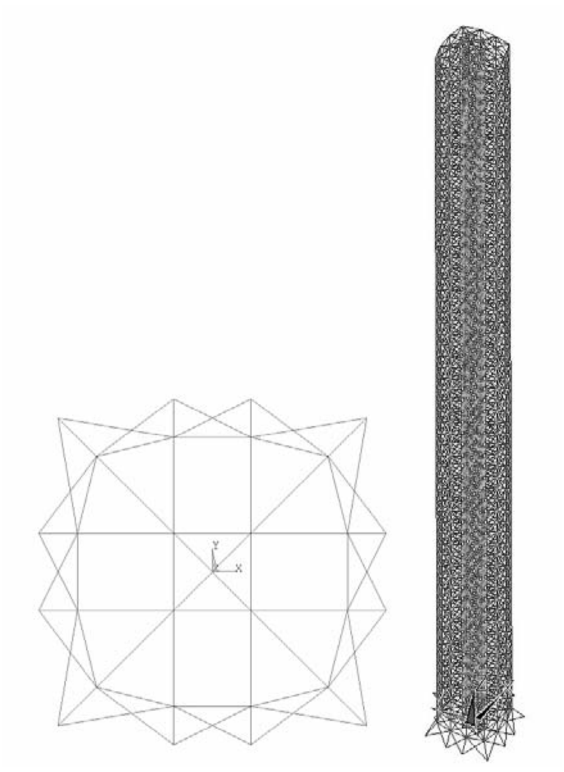
Figure 6.1 Example 1: (a) Top view of 35-story tower; (b) 35-story tower

6.4.2 Example 2 – 62-story space tower

This 62-story space truss tower in Figure 6.2 consisting of 4666 members and 2940 degrees of freedom is a modified structure from Example 1. There are 238 design variables, representing different member groups. The lower and upper bounds on the cross-sectional areas are 6.4516E-4 m^2 and 0.193548 m^2. The loads on the structure are as follows,

A. The vertical loads are given as 26.6892 kN at each node.
B. The horizontal loads are given as 4.4482 kN in the *X* direction at each node on the left side, 4.4482 kN in the *X* direction at each node on the right side.
C. The horizontal loads are given as 4.4482 kN in the *Y* direction at each node on

the back side, 4.4482 kN in the *Y* direction at each node on the front side.



(a) (b)

Figure 6.2 Example2: (a) Top view of 62-story tower; (b) 62-story tower

The displacement constraints are 0.90678 m in the *X*, *Y* and *Z* directions for the four nodes on the top level (about 1/250 of the height). In Case 1, a combination of all the loads A, B and C is applied to the structure. For all the formulations, the initial values for the optimization variables are the same as for Example 1. In Case 2, three loading conditions are considered, which consist of different combination of the lateral loads and vertical loads acting on the structure:

1. Loading condition A alone.
2. Loading conditions A and B acting together.
3. Loading conditions A, B and C acting together

All formulations converge to nearly the same optimum solution (350.486 m^3) for Case 1. The final optimal values for all the cross-sectional areas are given in Wang and Arora (2006a). Note that Case 1 and Case 2 have the same optimal solutions, which implies that the combination of all the loads together in fact is the dominant loading condition in Case 2.

6.5 Discussion of Results

Table 6.2 shows the comparison of sizes for the four formulations for the design examples. The largest problem has 23056 optimization variables and 50814 behavior constraints. Table 6.3 lists the estimated numbers of non-zero elements that need to be stored and used in SNOPT. The table also shows the number of non-zero elements that need to be stored and used if sparsity of the functions is not considered. It is seen that the sparsity of matrices needs to be utilized to solve large-scale design problems efficiently. If the sparsity of matrices is not considered, a standard PC may not have enough memory for the algorithm. Therefore, it is seen that the consideration of sparsity of the formulations is vital to their success. It is interesting to note that the conventional formulation actually requires more storage as compared to the alternative formulations. In order to provide a comparison of memory usage, approximate workspace requirements of SNOPT for all the formulations are recorded. For example, the memory usage reported by the operating system for Case 2 of the 35-story tower for the four formulations are in a range of (30-38) MB, (20-33) MB, (18-23) MB, and (12-15) MB, respectively.

Table 6.2 Number of variables and behavior constraints for different formulations

Example Problem		Formulation			
		T-CF	T-AF1	T-AF2	T-AF3
35-story	Case 1	72/1274	1008/2198	2270/4722	2270/2198
	Case 2	72/5096	3816/8792	8864/18888	8864/8792
62-story	Case 1	238/4678	3178/7606	7844/16938	7844/7606
	Case 2	238/14034	9058/22818	23056/50814	23056/22818

Table 6.3 Numbers of non-zero Jacobian elements for design examples

Example Problem		Formulation			
		T-CF	T-AF1	T-AF2	T-AF3
35-story	Case 1	9.2×10^4	6.1×10^4 [2.2×10^6]*	2.3×10^4 [1.1×10^7]	2.4×10^4 [5.0×10^6]
	Case 2	3.7×10^5	2.4×10^5 [3.4×10^7]	9.1×10^4 [1.7×10^8]	9.6×10^4 [7.8×10^7]
62-story	Case 1	1.1×10^6	2.2×10^5 [2.4×10^7]	8.4×10^4 [1.3×10^8]	8.9×10^4 [6.0×10^7]
	Case 2	3.3×10^6	6.7×10^5 [2.1×10^8]	2.5×10^5 [1.2×10^9]	2.7×10^5 [5.3×10^8]

*The numbers in the brackets give the total number of elements when sparsity is not considered.

All the formulations worked well and converged to the same optimal solutions for four design cases, even with a bad starting point. Table 6.4 lists the numbers of calls to ANSYS and Table 6.5 presents the numbers of iterations for the example problems. All the alternative formulations consume more iterations compared to the conventional formulation, since they have more variables. These numbers are quite reasonable for problems of this size. Table 6.6 shows the CPU times and the average CPU/iteration for all the design cases. It is seen that in the conventional formulation, the computational effort is scalable in terms of the problem size. For example, the CPU times needed are

1009 s for one loading case and 4003 s for four loading cases in Example 1, and 45473 s for one loading case and 125264 s for three loading cases in Example 2. This trend is not seen with the alternative formulations.

Table 6.4 Numbers of calls to ANSYS for design examples

Example Problem		**Formulation**			
		T-CF	**T-AF1**	**T-AF2**	**T-AF3**
35-story	**Case 1**	173 + 171*	308	191	352
	Case 2	172 + 170	495	255	322
62-story	**Case 1**	239 + 237	708	397	302
	Case 2	236 + 234	2531	239	304

*Analysis + Sensitivity Analysis

Table 6.5 Numbers of iterations for design examples

Example Problem		**Formulation**			
		T-CF	**T-AF1**	**T-AF2**	**T-AF3**
35-story	**Case 1**	81	209	166	263
	Case 2	82	313	221	179
62-story	**Case 1**	114	509	331	260
	Case 2	114	1010	205	255

It is seen that when the problem size is relatively small, the computational efforts for the three alternative formulations are smaller than the conventional formulation, with T-AF2 and T-AF3 requiring less CPU effort than T-AF1. This is the same conclusion as the one reached by Chapter 4 and Wang and Arora (2005a). However, for the largest example (Case 2 of Example 2), the conventional formulation uses less CPU time than all the alternative formulations, although the CPU/iteration is smaller for the alternative formulations. The CPU/iteration for the alternative formulations shows some interesting trend. As the size of the optimization problem increases, the CPU/iteration becomes larger for T-AF2 and T-AF3 compared to T-AF1. Apparently, the QP solver for calculation of the search direction becomes less efficient as the numbers of variables and constraints increase. Therefore, better QP solvers for large sparse problems need to be developed.

Table 6.6 Computing efforts for different formulations (CPU, s)

Example Problem		**Formulation**			
		T-CF	**T-AF1**	**T-AF2**	**T-AF3**
35-story	**Case 1**	1009 (13)*	551 (3)	305 (2)	459 (2)
	Case 2	4003 (50)	5526 (18)	2141 (10)	3862 (22)
62-story	**Case 1**	45473 (399)	30989 (61)	29010 (88)	25891 (100)
	Case 2	125264 (1099)	252360 (250)	136834 (668)	217182 (852)

*Average CPU/iteration

Table 6.7 Wall-clock times for different formulations (s)

Example Problem		**Formulation**			
		T-CF	**T-AF1**	**T-AF2**	**T-AF3**
35-story	**Case 1**	8497	3135	1886	3359
	Case 2	26921	11161	5672	7335
62-story	**Case 1**	142609	54053	41611	35407
	Case 2	366448	344108	144931	227530

It is also important to note that the wall-clock times with all the formulations are much larger than the CPU times (refer to Table 6.7). This is because ANSYS is used as an independent program executed on a local area network. Thus, there is considerable overhead time in the use of a commercial analysis program over the local area network. If wall-clock times are used for comparison, the conventional formulation requires more time than the alternative formulations for all the design cases, even for the largest example.

In the alternative SAND formulations, the optimization problem is very large because there are more variables. The size of the problem becomes still larger when the structure is subjected to multiple loading conditions, and it can easily exceed the capacity of current personal computers. However, note that for large-scale applications, the SAND formulations require less memory for the Jacobian and Hessian matrices, and they avoid repeated analyses of the structure. T-AF2 and T-AF3 with forces and stresses as variables have more optimization variables; however, they are more efficient than T-AF1. The reason is that they have many linear constraints, which are more efficient to treat in the solution process.

6.6 Summary

Three alternative formulations for structural optimization were applied to the design of large-scale linearly elastic trusses. Basic structures of the Jacobians of the constraints associated with the formulations were studied and implemented with an existing sparse NLP solver. It is concluded that the exploitation of the sparsity properties of the formulations is critical for storage of data, computational efficiency, and furthermore, success of the alternative formulations. T-AF2 and T-AF3 with forces and stresses as additional optimization variables outperform T-AF1, where only displacements are treated as additional optimization variables. In terms of CPU times, the SAND formulations outperform the conventional formulation, except for the largest example problem. When the problem size is too large, the sparse QP subproblem solver becomes slow to converge, resulting in more computational effort than the conventional formulation. However, in terms of the wall-clock time or the CPU time/iteration, the SAND formulations are more efficient than the conventional formulation, since the number of call to ANSYS is much smaller.

SAND represents a fundamental shift in the way analysis and design problems are currently treated. Further research is suggested to fully study and utilize sparse features of the alternative formulations for large and more complex problems. The exploitation of the sparsity, decomposition to reduce sizes, efficient solution of sparse QP subproblems and parallel algorithms for the alternative SAND formulations need to be further studied, developed and combined for much broader practical applications of these formulations.

CHAPTER 7
TRANSIENT DYNAMIC RESPONSE OPTIMIZATION I

7.1 Introduction

Transient dynamic response optimization problems are difficult to solve because they involve integration of linear or nonlinear differential-algebraic equations (DAEs) or just differential equations (DEs). The most common approach for optimization of such problems has been the one where only the design variables are treated as optimization variables (Afimiwala and Mayne 1974; Haug and Arora 1979; Hsieh and Arora 1984; Grandhi and Haftka 1986; Lim and Arora 1987; Tseng and Arora 1989; Kim and Choi 1998; Arora 1999). All other response quantities, such as displacements, velocities, and accelerations, are treated as implicit functions of the design variables. Therefore, in the optimization process, a system of DEs is integrated to obtain the response (state) variables and to calculate values of various functions of the optimization problem. Then an optimization algorithm is used to update the design. This nested process of solution of DEs and design update, also called the *conventional approach*, is repeated until a stopping criterion is satisfied. This optimization process, however, is difficult to use in practice. The main difficulty is that the response quantities are implicit functions of the design variables, which require special methods for their gradient evaluation, such as the *direct differentiation method* or the *adjoint variable method* (Arora 1999). These methods require integration of additional DEs. Unless finite difference method is used to calculate the gradients, it is difficult to optimize systems with any existing simulation software, especially for multidisciplinary problems requiring use of different discipline-specific analysis software.

Another interesting approach for transient dynamic optimization is the so-called equivalent static load method (Kang *et al.* 2001; Choi and Park 2002; Kang *et al.* 2005), where the problem is transferred to a quasi-static problem. The idea is to find a static load set that can generate the same displacement field as that with the dynamic load at certain times. Therefore, multiple equivalent static load sets obtained at all the time intervals can represent various states of the structure under the dynamic load. However, with this approach, DEs must still be integrated a number of times and design sensitivity analysis must also be performed for the resulting static problems.

To alleviate the difficulties mentioned above, a fundamental shift in the direction of research on optimization of dynamic systems is needed. It will be useful to develop alternative formulations that do not require explicit solution of DEs at each iteration and there is no need for special design sensitivity analysis procedures. By formulating the optimization problem in a mixed space of design and state variables, these two objectives can be met. Thus the purpose of this chapter is to expand upon this idea by proposing alternative formulations for optimization of transient dynamic systems, and evaluate these using extensive numerical experiments. Various state variables, such as the displacements, velocities and accelerations, are treated as independent variables in the formulations. The equations of motion become equality constraints. All constraints of the problem in the proposed formulations are expressed explicitly in terms of the optimization variables. Therefore their gradient evaluations become quite simple. Although the resulting optimization problem is large, it is quite sparse which can be solved using sparse nonlinear programming (NLP) algorithms. Two numerical examples are used to study the alternative approaches and compare solutions with the conventional approach. Advantages and disadvantages of the formulations are also discussed (Wang and Arora 2005b).

This idea of using state and design variables simultaneously in the optimization process has been used in the literature for the static response structural optimization problems. This is known as SAND (Haftka 1985; Kirsch and Rozvany 1994; Orozco and Ghattas 1992a; Arora and Wang 2005). A similar approach has also been used to solve optimal control problems (von Stryk and Bulirsch 1992; Betts and Huffman 1993; Barclay *et al.* 1997; Cervantes and Biegler 1998; Betts 1998, 2000; Cervantes *et al.* 2000), which is called the direct collocation/transcription method. The main applications of the approach are for aerospace trajectory optimization (Betts and Huffman 1993; Betts 2000), and chemical process engineering (Cervantes and Biegler 1998). The basic idea is to discretize the system of first order DEs, and define finite dimensional approximations for the state and control variables. The discretized state equations are treated as equality constraints in the optimization process. The collocation conditions are determined using an implicit formula, such as the implicit Runge-Kutta method, or some polynomial interpolations. Bounds on the state variables are usually enforced at the time grid points, or at all the collocation points in the time domain. The optimization variables are the state variables and their first derivatives at time-grid points, the control variables, and the algebraic variables at all the collocation points. The resulting NLP is usually solved by a sequential quadratic programming (SQP) method (Cervantes and Biegler 1998; Betts 2000) or an interior point method (Cervantes *et al.* 2000), which exploit the sparse and block features of the problem structure. Some detailed formulations for optimal control problems can be found in Barclay *et al.* (1997) and Betts (1998).

7.2 Dynamic Response Optimization Problem

The basic dynamic response optimization problem is to determine design parameters related to stiffness and damping properties of the dynamic system, to achieve certain goals (e.g., minimization of a cost function, such as the maximum displacement or acceleration) while satisfying all the performance requirements. In this section, a general problem for optimization of dynamic systems is presented. In the remaining sections, the conventional and alternative formulations, and numerical results for two example problems are presented and discussed.

7.2.1 Simulation model

Let $\mathbf{x}$ be an m dimensional vector to represent the design variables for the problem, which may include the sizing variables of a structure; or directly mass, stiffness, and damping parameters of the dynamic system. $\mathbf{z}$ is a d dimensional vector that represents the state variables for the problem. The equation of motion for a linear system to determine the state variables can be written as follows (nonlinear problem can be treated similarly as seen later in an example problem):

$$\mathbf{M}(\mathbf{x})\ddot{\mathbf{z}}(t)+\mathbf{C}(\mathbf{x})\dot{\mathbf{z}}(t)+\mathbf{K}(\mathbf{x})\mathbf{z}(t)=\mathbf{F}(\mathbf{x},t) \qquad (7.2.1)$$

with the initial conditions $\mathbf{z}(0)=\mathbf{z}_0$, and $\dot{\mathbf{z}}(0)=\dot{\mathbf{z}}_0$; $\mathbf{M}$, $\mathbf{C}$ and $\mathbf{K}$ represent $(d\times d)$ the generalized mass, damping and stiffness matrices; and $\mathbf{F}(\mathbf{x},t)$ is a $d\times 1$ generalized force vector. Note that Eq. (7.2.1) cannot be solved in a closed form to obtain explicit functional forms for $\ddot{\mathbf{z}}$, $\dot{\mathbf{z}}$ and $\mathbf{z}$ in terms of the design variables $\mathbf{x}$, which causes difficulty in the optimization process. The equations of motion can be solved directly in the second order form (Bathe 1982; Chopra 2001) in Eq. (7.2.1), or by converting it to a first order form, the so-called the *state space representation* of Eq. (7.2.1), as:

$$\dot{\mathbf{y}}(t) = \mathbf{f}(\mathbf{y}(t), \mathbf{x}, t) = \overline{\mathbf{K}}(\mathbf{x})\mathbf{y}(t) + \overline{\mathbf{F}}(\mathbf{x}, t) \qquad (7.2.2)$$

where $\mathbf{y}^T = [\mathbf{z} \quad \dot{\mathbf{z}}]$, $\overline{\mathbf{K}}$ is the system matrix $(2d \times 2d)$, and $\overline{\mathbf{F}}(\mathbf{x},t)$ is $2d \times 1$ vector given as:

$$\overline{\mathbf{K}}(\mathbf{x}) = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{M}^{-1}(\mathbf{x})\mathbf{K}(\mathbf{x}) & -\mathbf{M}^{-1}(\mathbf{x})\mathbf{C}(\mathbf{x}) \end{bmatrix}; \; \overline{\mathbf{F}}(\mathbf{x},t) = \begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1}(\mathbf{x})\mathbf{F}(\mathbf{x},t) \end{bmatrix} \qquad (7.2.3)$$

in which $\mathbf{I}$ is a $d \times d$ identity matrix. It is worthwhile to examine both the second and the first order forms of the dynamics equations.

7.2.2 Cost function and constraints

In general, a cost functional includes the state and design variables, as

$$J = c_0(\mathbf{x}, T) + \int_0^T h_0(\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, t)\, dt \qquad (7.2.4)$$

where T is the total time interval considered. The objective J may be the cost of the system, performance measures, or any other function of the state variables. A time-dependent functional, such as maximum acceleration or displacement, can also be treated as will be seen later in the example problems. Design requirements are imposed mostly as inequality constraints (equality constraints can also be treated). One type of constraints that involves integration over time as

$$\mathbf{g} = \mathbf{c}(\mathbf{x}, T) + \int_0^T \mathbf{h}(\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, t)\, dt \le 0 \qquad (7.2.5)$$

The other type of constraints is the so-called point-wise constraint, which needs to be satisfied at each point of the entire time interval $t \in [0, T]$:

$$\mathbf{g}(\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, t) \le 0 \qquad (7.2.6)$$

These constraints may include the following displacement, velocity and acceleration constraints, and the time-independent constraints on the design variables:

$$\mathbf{z}^L \le \mathbf{z} \le \mathbf{z}^U \,;\;\; \dot{\mathbf{z}}^L \le \dot{\mathbf{z}} \le \dot{\mathbf{z}}^U \,;\;\; \ddot{\mathbf{z}}^L \le \ddot{\mathbf{z}} \le \ddot{\mathbf{z}}^U \tag{7.2.7}$$

$$\mathbf{x}^L \le \mathbf{x} \le \mathbf{x}^U \tag{7.2.8}$$

where $\mathbf{z}^L, \dot{\mathbf{z}}^L, \ddot{\mathbf{z}}^L, \mathbf{x}^L$ and $\mathbf{z}^U, \dot{\mathbf{z}}^U, \ddot{\mathbf{z}}^U, \mathbf{x}^U$ are the lower and upper bounds for the generalized displacements, velocities, accelerations and design variables, respectively. Any other design requirements may also be included in Eqs. (7.2.5) and (7.2.6).

7.3 Conventional Formulation – Only Design Variables as Optimization Variables

In the conventional formulation, optimization is carried out only in the space of design variables. This is the most common way to formulate the transient dynamic response optimization problems (Arora 1999) and it includes the minimum number of optimization variables. The total time interval $[0, T]$ is divided into N intervals (N +1 time grid points), and the equations of motion are integrated to determine the system response and thus calculate various functions of the optimization problem. The optimization problem is to find $\mathbf{x}$ to minimize the cost functional of Eq. (7.2.4) subject to the constraints of Eqs. (7.2.5), (7.2.6) and (7.2.8).

It is important to note that five treatments of the point-wise constraints in Eq. (7.2.6) have been presented and evaluated in the literature (Tseng and Arora 1989; Arora 1999). These include the equivalent integral and critical point methods as well as the conventional approach where the constraints are imposed at each time grid point. It has been concluded that the imposition of constraints at each time grid point is quite effective. The reason is that although the number of constraints becomes very large with this approach, the number of active constraints is quite small (during the iterative process and at the optimum). Therefore the approach works quite well, and is quite easy to implement for numerical calculations. The equivalent integral and critical point constraint methods appear to be quite attractive as well because the number of constraints remains quite small. However, there are some theoretical difficulties with them causing the optimization process to fail sometimes (Hsieh and Arora 1984; Tseng and Arora 1989; Arora 1999). These two methods were implemented for some of the examples presented later. As expected the performance of the equivalent integral approach was not good as compared to the constraint imposed at each time point (conventional approach) in terms of the computational effort as well as convergence to the optimum point. Therefore, in this study, results are presented with only the approach where the constraints are imposed at each time grid point.

Since the constraint functions in this formulation are implicit functions of the design variables, implicit differentiation procedures need to be used to evaluate the gradients. Certainly, the finite difference methods can be used to evaluate the gradients, since they are quite easy to implement. However, accuracy of the gradients is questionable which may affect convergence to the optimal solution. Analytically, the direct differentiation or the adjoint variable method can be used (Hsieh and Arora 1984; Arora 1999). These procedures are difficult to implement with an existing simulation code, because the code needs to be recalled to solve for displacement, velocity and acceleration gradients or the adjoint vectors. Then, the gradients of the response functionals need to be assembled using the adjoint vectors or the displacement, velocity and acceleration gradients. This is one of the main drawbacks of the conventional optimization formulation. In the present work, both the finite difference and direct differentiation approaches were used to solve the example problems. Both the methods worked well and converged to the same solutions. However, only the results requiring least amount of computational effort are reported.

7.4 Alternate Formulation 1 – Design Variables and Displacements as Optimization Variables

In this formulation, the displacements $\mathbf{z}(t)$ are also treated as optimization variables. Thus the problem is to determine $\mathbf{x}$ and $\mathbf{z}(t)$ to minimize the cost functional of Eq. (7.2.4) subject to the constraints of Eqs. (7.2.1), (7.2.5), (7.2.6) and (7.2.8). In the numerical solution process, Eqs. (7.2.1) and (7.2.6) are discretized into *N+1* equations in the entire time interval $[0,T]$, and they represent in fact the constraints that need to be satisfied at each time grid point, as

$$\mathbf{M}(\mathbf{x})\ddot{\mathbf{z}}_i + \mathbf{C}(\mathbf{x})\dot{\mathbf{z}}_i + \mathbf{K}(\mathbf{x})\mathbf{z}_i = \mathbf{F}(\mathbf{x}, t_i), \qquad i = 0, N \tag{7.4.1}$$

$$\mathbf{g}_i(\mathbf{x}, \mathbf{z}_i, \dot{\mathbf{z}}_i, \ddot{\mathbf{z}}_i, t_i) \le \mathbf{0}, \qquad i = 0, N \tag{7.4.2}$$

The optimization variables now become $\mathbf{x}$ and $\mathbf{z}_i$ (i = *-1*, *N+1*). Note that Eqs. (7.4.1) are the equality constraints between the variables representing the equations of motion for the system. It is obvious that in order to make Eqs. (7.4.1) and (7.4.2) explicit with respect to the optimization variables $\mathbf{x}$ and $\mathbf{z}$, the velocity and acceleration vectors $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$ need to be written explicitly in terms of $\mathbf{z}$. One simple way is to use the finite difference approximations as

$$\dot{\mathbf{z}}_i = \dot{\mathbf{z}}_i(t_i) = \frac{\mathbf{z}_{i+1} - \mathbf{z}_{i-1}}{2\Delta t}, \qquad i = 0, N \tag{7.4.3}$$

$$\ddot{\mathbf{z}}_i = \ddot{\mathbf{z}}_i(t_i) = \frac{\mathbf{z}_{i+1} - 2\mathbf{z}_i + \mathbf{z}_{i-1}}{\Delta t^2}, \qquad i = 0, N \tag{7.4.4}$$

where Δt is the time interval defined as $\Delta t = T / N$. Thus in terms of the variables $\mathbf{x}$ and $\mathbf{z}_i$, all the problem functions have an explicit form. Therefore special design sensitivity analysis procedures are not needed to evaluate the gradients. Even the equations of motion are not integrated, but imposed as equality constraints in the optimization process. These equations may not be satisfied at each optimization iteration. They are, however, satisfied at the final solution. Note that Eqs. (7.4.1) are linear in $\mathbf{z}_i$ if the original DEs are linear, and nonlinear if they are nonlinear.

Note also that ideas similar to direct collocation can also be used here (Betts 1998); however, they usually involve more complicated relationships among the state variables as equality constraints. That will make the gradient evaluation as well as computer implementation more complex. These are further studied in Chapter 8.

7.5 Alternate Formulation 2 – Design Variables, Displacements and Velocities as Optimization Variables

In alternate formulation 1, all functions needed to be expressed in terms of the design variables and displacements for numerical calculations. However, if the velocities or accelerations are also treated as variables, it will give choice of expressing some constraints (e.g., the equations of motion) in terms of velocities or accelerations, to simplify their expressions. This may lead to simpler gradient evaluation and computer implementation. To evaluate this idea, velocities and accelerations are also treated as variables in this and the next subsection.

If displacements and velocities are treated as optimization variables, accelerations can be expressed in terms of the velocities. Therefore, another explicit form can be obtained. The problem is to determine $\mathbf{x}$, $\mathbf{z}$ and $\dot{\mathbf{z}}$ to minimize the cost function of Eq. (7.2.4), subject to Eqs. (7.2.1), (7.2.5), (7.2.6) and (7.2.8). The optimization variables in this formulation are $\mathbf{x}$, $\mathbf{z}_i$, and $\dot{\mathbf{z}}_i$ (*i* = *-1*, *N+1*), and this is denoted as Form 1 of the formulation. In fact, this form is similar to the state space representation. If the state space variables $\mathbf{y}$ in Eq. (7.2.2), i.e., displacements $\mathbf{z}$ and velocities $\dot{\mathbf{z}}$ are treated as variables in the optimization formulation, the problem functions become explicit in terms of these variables. The problem is to determine $\mathbf{x}$ and $\mathbf{y}$ to minimize the cost function of Eq. (7.2.4), subject to the first order DEs in Eq. (7.2.2), and the point-wise constraints

$$\mathbf{g}(\mathbf{x}, \mathbf{y}, \dot{\mathbf{y}}, t) \leq \mathbf{0} \tag{7.5.1}$$

Note that Eqs. (7.2.2) and (7.5.1) are discretized into *N+1* equations in the entire time interval $[0, T]$, and they in fact represent the system of constraints as follows:

$$\dot{\mathbf{y}}_i = \overline{\mathbf{K}}(\mathbf{x})\mathbf{y}_i + \overline{\mathbf{F}}_i(\mathbf{x}, t_i), \qquad i = 0, N \tag{7.5.2}$$

$$\mathbf{g}_i(\mathbf{x}, \mathbf{y}_i, \dot{\mathbf{y}}_i, t_i) \leq \mathbf{0}, \qquad i = 0, N \tag{7.5.3}$$

The optimization variables for this formulation are $\mathbf{x}$ and $\mathbf{y}_i$ (*i* = *-1*, *N+1*). In order to make Eqs. (7.5.2) and (7.5.3) explicit with respect to the optimization variables $\mathbf{x}$ and $\mathbf{y}$, vector $\dot{\mathbf{y}}$ needs to be written explicitly in terms of $\mathbf{x}$ and $\mathbf{y}$. Using the finite difference approximations, we get

$$\dot{\mathbf{y}}_i = \dot{\mathbf{y}}_i(t_i) = \frac{\mathbf{y}_{i+1} - \mathbf{y}_{i-1}}{2\Delta t}, \qquad i = 0, N \tag{7.5.4}$$

Note that Eq. (7.5.4) is not as accurate as Eq. (7.4.4), since the error involved here is $O\left((2\Delta t)^2\right)$ (Atkinson 1988), which is larger than that for Eq. (7.4.4). In order to obtain an approximation of the accelerations with higher accuracy, the velocities at the middle of each time step, $\dot{\mathbf{z}}_{\frac{2i-1}{2}}$ (*i = 0, N+1*) are introduced as additional variables, and this is called Form 2 of the formulation. The accelerations are therefore expressed as,

$$\ddot{\mathbf{z}}_i = \frac{\dot{\mathbf{z}}_{\frac{2i+1}{2}} - \dot{\mathbf{z}}_{\frac{2i-1}{2}}}{\Delta t}, \qquad i = 0, N \tag{7.5.5}$$

The optimization variables now are $\mathbf{x}$, $\mathbf{z}_i$ (*i = -1, N+1*), $\dot{\mathbf{z}}_i$ (*i = 0, N*), and $\dot{\mathbf{z}}_{\frac{2i-1}{2}}$ (*i = 0, N+1*). More equality constraints are needed due to the introduction of additional variables. From the finite difference approximation, the velocities are expressed as:

$$\dot{\mathbf{z}}_i = \frac{\mathbf{z}_{i+1} - \mathbf{z}_{i-1}}{2\Delta t}, \qquad i = 0, N \tag{7.5.6}$$

$$\dot{\mathbf{z}}_{\frac{2i-1}{2}} = \frac{\mathbf{z}_i - \mathbf{z}_{i-1}}{\Delta t}, \qquad i = 0, N+1 \tag{7.5.7}$$

7.6 Alternate Formulation 3 – Design Variables, Displacements, Velocities and Accelerations as Optimization Variables

From Eq. (7.2.1), it can be seen that other alternative formulations are possible. If the displacements $\mathbf{z}$, velocities $\dot{\mathbf{z}}$ and accelerations $\ddot{\mathbf{z}}$ are treated as variables simultaneously, another explicit formulation can be obtained. Since the state variables are related to each other, more equality constraints need to be imposed in the formulation, such as Eqs. (7.4.3) and (7.4.4).

The problem is to determine $\mathbf{x}$, $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$ to minimize the cost function of Eq. (7.2.4), subject to Eqs. (7.2.1), (7.2.5), (7.2.6), and (7.2.8). After discretization, the optimization variables in this formulation are $\mathbf{x}$, $\mathbf{z}_i$ (i = -1, N+1), $\dot{\mathbf{z}}_i$ and $\ddot{\mathbf{z}}_i$ (i = 0, N).

7.7 Evaluation of Formulations

Table 7.1 shows the sizes of all the formulations. The following symbols are used in the table: m = number of elements in the design variable vector $\mathbf{x}$; N = number of time intervals (number of grid points = N + 1); e = number of constraints in $\mathbf{g}$; d = number of DEs in Eq. (7.2.1), or the dimension of vector $\mathbf{z}$. Note that some of the inequality constraints g_i in the alternative formulations 1-3 may become simple bounds on variables, as seen later in the examples.

Table 7.1 Number of variables and constraints for different formulations

Item	Conventional	Alternate 1	Alternate 2	Alternate 3
No. of Variables	m	$m+d(N+3)$	$m+d(2N+6)$ or $m+d(3N+6)$	$m+d(3N+5)$
No. of Equality Constraints	0	$d(N+1)$	$d(2N+2)$ or $d(3N+4)$	$d(3N+3)$
No. of Inequality Constraints	$e(N+1)$	$e(N+1)$	$e(N+1)$	$e(N+1)$

It is noted here that since the point-wise constraint is imposed only at the grid points, it is possible that the constraint could be violated between the grid points. This is the usual difficulty whenever a continuum problem is discretized for numerical calculations. In the present case, if the system response is relatively smoother and slowly varying, the constraint violation, if any, between the grid points will be quite small. When the response is varying rapidly, a smaller grid size can reduce the possibility of violation between the grid points at the expense of additional calculations.

In the alternative formulations, since the objective and constraint functions are all explicit in terms of the optimization variables, the gradients of functions can be obtained easily by direct differentiation. Note that the alternative formulations do not require the equations of motion in Eq. (7.2.1) to be satisfied exactly at each optimization iteration. They need to be satisfied only at the final solution point. This has an advantage for the alternative formulations because they avoid possible instabilities or failure of the integration process for the equations of motion. Also, unnecessary simulations of the system are avoided at intermediate designs, where it might be difficult to obtain a solution. However it can be seen that the number of variables and constraints becomes very large in the alternative formulations, and the size depends on the number of grid points. Large-scale NLP solution algorithms with sparse matrix capabilities are required to solve the alternative formulations efficiently. Some aspects of the alternative formulations will be discussed in details in Section 7.8, when particular design problems are presented.

7.8 Numerical Examples

For numerical evaluation, the conventional and three alternative formulations presented in previous sections are applied to solve two examples problems. The alternative formulations are solved using the sparse SQP algorithm in SNOPT (Gill *et al.* 2002), while the conventional formulation is solved using the dense SQP solver in SNOPT since the problem is dense. SNOPT has an option to compute derivatives using a combined forward and central difference method. This option is also used to solve the example problems with the conventional formulation. Results of the examples are listed and compared. Advantages and disadvantages of the formulations are discussed.

For the conventional formulation, two subroutines in the IMSL mathematical library (1997) are used to integrate the first order DEs in all the examples. These are DIVPRK (Runge-Kutta-Verner method), and DIVPAG (Adams-Moulton method or backward differentiation method). These are very good methods that can also solve stiff DEs (DIVPAG) which are encountered in some mechanical system applications. All the foregoing three methods are tried for the example problems presented here and only the most efficient one is selected for optimization. It is noted that these methods automatically adjust the step size internally to maintain stability and accuracy of the solution process. Therefore the computational effort to integrate the equations of motion and sensitivity equations can be substantial. In contrast, the alternative formulations use a fixed time grid in the solution process. To have a fairer comparison of the performance of different formulations, a fixed time step option was tried for integration of the DEs. In some cases, the integration procedure failed to converge causing the optimization process to terminate prematurely. Therefore, it is concluded that a reliable and robust DE integrator is crucial for success of the conventional formulation. In addition, the governing equations are sometimes stiff and include algebraic equations as side constraints. Therefore it is better to use more general approaches for integration of equations of motion. For all the results reported later with the conventional formulation, the time step size was allowed to be adjusted automatically within the integration routine.

A PC with 2.53 GHz processor and 512 MB RAM is used for running the programs and recording the CPU times. Each solution case of the example problems was run several times with different starting point and the shortest time was recorded. Also any differences in the solution points, if any, were noted to observe global optimality of the solution.

7.8.1 Example 1 – 1-DOF impact absorber

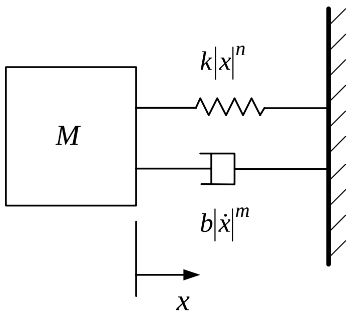


Figure 7.1 Example 1: 1-DOF impact absorber configuration

The problem is to design a nonlinear shock absorber with *n*th order stiffness and *m*th order damping (Afimiwala and Mayne 1974). The objective is to minimize the maximum acceleration of the attached mass during the transient response. The equation of motion is given as

$$M\ddot{x} + b|\dot{x}|^m \operatorname{sgn}\dot{x} + k|x|^n \operatorname{sgn} x = 0 \tag{7.8.1}$$

with initial conditions as $x(0) = 0$ and $\dot{x}(0) = V$, and $\operatorname{sgn} y = y/|y|$. Equation (7.8.1) is normalized by considering transformation of the displacement and the time parameter as $X = x/L$ and $\tau = Vt/L$, where L is the maximum displacement of the mass during the resulting transient motion. Therefore, the normalized equation of motion is

$$\ddot{X} + B\left|\dot{X}\right|^{m} \operatorname{sgn} \dot{X} + K\left|X\right|^{n} \operatorname{sgn} X = 0 \tag{7.8.2}$$

where the "over dots" now represent differentiation with respect to τ; the initial conditions become $X(0) = 0$ and $\dot{X}(0) = 1$; and the two new parameters are defined as

$$B = \frac{bLV^{m-2}}{M}; \; K = \frac{kL^{n+1}}{MV^{2}} \tag{7.8.3}$$

The optimal design problem is to find *B* and *K*, representing the mass, stiffness and damping properties of the system to

$$\underset{B,K}{\text{minimize}} \; \left|\ddot{X}\right|_{\max} \quad \text{subject to} \quad X_{\max} \le 1 \tag{7.8.4}$$

By introducing an artificial variable *E* and discretizing the normalized time interval *T* of 2 ($0 \le \tau \le 2$) into *N* steps ($\Delta t = T / N$), the conventional formulation for the optimal design problem is to find three variables *E*, *B*, and *K* to minimize E subject to

$$\left|\ddot{X}_{i}\right| \le E; \quad i = 0, N \tag{7.8.5}$$

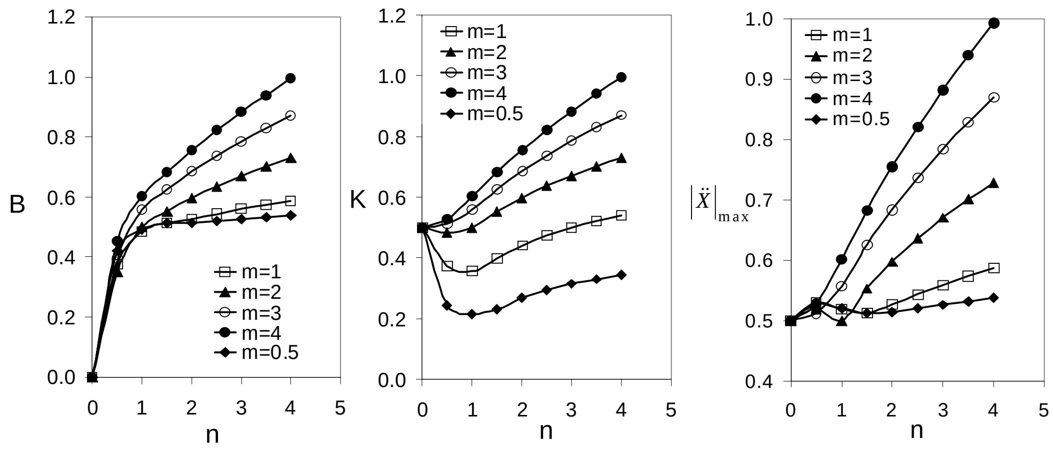
$$X_{i} \le 1; \quad i = 0, N \tag{7.8.6}$$

The numbers of optimization variables, constraints (excluding simple bound constraints) and non-zero elements in the constraint Jacobian are listed in Table 7.2. The initial values of the variables B, K, and E are taken as unity for the conventional formulation and all the three alternative formulations. No special techniques are used to find the starting values of other variables for the three alternative formulations. The starting values of displacements, velocities and accelerations are all taken as unity. Note also that the inequality constraints in Eq. (7.8.6) become simple bound constraints in the alternative formulations that can be treated efficiently in the optimization algorithms.

The optimum solutions obtained by all the formulations are quite similar to those obtained in the literature (Afimiwala and Mayne 1974) and are plotted in Figure 7.2. Figures 7.2 (a) and (b) show the optimum values for non-dimensional damping and stiffness parameters *B* and *K* for various values of *m* and *n* between zero and four. Figure 2 (c) shows the maximum acceleration magnitude $\left|\ddot{X}\right|_{\max}$ for the optimum values of *B* and *K*. The optimum solutions corresponding to the nonlinear system with *m* and *n* taken as 2 and the number of grid points as 50 are listed in Table 7.3.

Table 7.2 Sizes of design example 1

Item	Conventional	Alternate 1	Alternate 2	Alternate 3
No. of Variables	*3*	*N+5*	*2N+7*	*3N+7*
No. of Constraints	*2N+2*	*3N+3*	*4N+4*	*5N+4*
No. of Non-zero Elements in Jacobian	*6N+6*	*13N+7*	*15N+18*	*16N+13*
Total No. of Elements in Jacobian	*6N+6*	$3N^2+18N+15$	$8N^2+36N+28$	$15N^2+47N+28$



(a) (b) (c)

Figure 7.2 Example 1: Optimum values for (a) non-dimensional damping *B*, and (b) stiffness *K*; (c) maximum acceleration magnitude $\left|\ddot{X}\right|_{\max}$ using optimum *B* and *K*

7.8.2 Example 2 – 5-DOF vehicle suspension system

A five degree of freedom vehicle suspension system (Haug and Arora 1979) is shown in Figure 7.3 (the state variables z_i are defined there). The objective is to minimize the extreme acceleration of the driver's seat (mass m_1) for a variety of vehicle speeds and road conditions defined by the functions $f_1(t)$ and $f_2(t)$. The design variables are the spring constants k_1, k_2, and k_3, and the damping constants c_1, c_2, and c_3. The motion of the vehicle is also constrained so that the relative displacements between the chassis and the driver's seat, the chassis and the front and rear axles are within given limits. This is the design problem 1 of Example 5.3 on pages 348-354 in Haug and Arora (1979). The equations of motion for the system are given there.

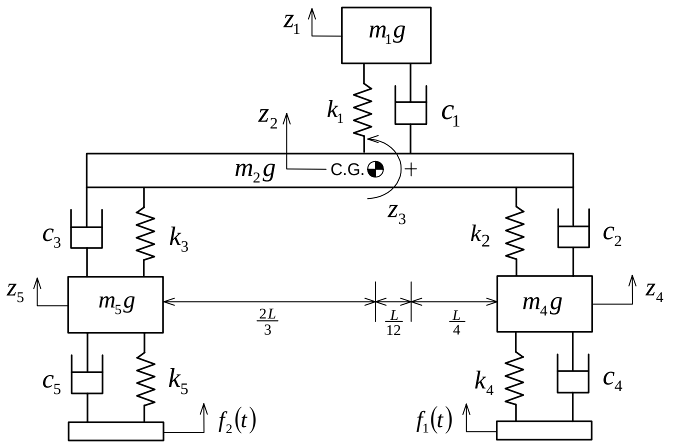


Figure 7.3 Example 2: 5-DOF vehicle suspension system

The optimum design problem is to find k_1, k_2, k_3, c_1, c_2, and c_3 that minimize $\max\limits_{t\in[0,T]}|\ddot{z}_1(t)|$ for the given road profile 1 (Haug and Arora 1979). By introducing an artificial variable E, the reformulated problem is to minimize E subject to the state equations, and the following inequality constraints in the time interval $[0,T]$:

$$|\ddot{z}_1(t)| \le E \tag{7.8.7}$$

$$\left| z_2(t) + \frac{L}{12} z_3(t) - z_1(t) \right| \le 2, \quad 0 \le t \le T \tag{7.8.8}$$

$$\left| z_4(t) - z_2(t) - \frac{L}{3} z_3(t) \right| \le 5, \quad 0 \le t \le T \tag{7.8.9}$$

$$\left| z_5(t) - z_2(t) + \frac{2L}{3} z_3(t) \right| \le 5, \quad 0 \le t \le T \tag{7.8.10}$$

$$| z_4(t) - f_1(t) | \le 2, \quad 0 \le t \le T \tag{7.8.11}$$

$$| z_5(t) - f_2(t) | \le 2, \quad 0 \le t \le T \tag{7.8.12}$$

and simple bounds on the design variables. According to the literature (Haug and Arora 1979), the initial design, lower and upper bounds for the design variable $[k_1, k_2, k_3, c_1, c_2, c_3, E]$ are taken as $[100, 300, 300, 10, 25, 25, 332.6]$, $[50, 200, 200, 25, 5, 1]$ and $[500, 1000, 1000, 50, 80, 80, 500]$, respectively. The total time interval is considered as 2.5 seconds and the number of time steps is set to 300. No special techniques are used to find an initial point for the alternative formulations. The starting values for the displacements, velocities, and accelerations are taken as zero. Note that in all the alternative formulations, Eq. (7.8.7) is treated as a pair of linear inequalities, Eqs. (7.8.8) - (7.8.10) become linear constraints, and Eqs. (7.8.11) and (7.8.12) become simple bounds on the variables. Table 7.4 lists the sizes of the problems for different formulations.

Table 7.3 Optimum designs for examples

1-DOF (N = 50)				5-DOF (N = 300)				
Var.	**CF**	**AF1 & 3**	**AF2**	**Var.**	**CF (Ref. 2)**	**CF**	**AF1**	**AF2 & 3**
E	0.59725	0.59752	0.59735	E	257.40	254.56	254.69	254.38
B	0.59725	0.59752	0.59735	k_1	50.00	50.00	50.00	50.00
K	0.59725	0.59752	0.59735	k_2	200.00	200.00	200.00	200.00
				k_3	241.90	200.00	200.00	200.00
				c_1	12.89	45.45	19.97	45.23
				c_2	77.52	77.35	76.99	77.39
				c_3	80.00	80.00	80.00	80.00

Table 7.4 Sizes of design example 2

Item	**Conventional**	**Alternate 1**	**Alternate 2**	**Alternate 3**
No. of Variables	*7*	*5N+22*	*15N+37*	*15N+32*
No. of Constraints	*6N+6*	*10N+15*	*20N+25*	*20N+20*
No. of Non-zero Elements in Jacobian	*42N+42*	*92N+102*	*111N+126*	*109N+109*
Total No. of Elements in Jacobian	*42N+42*	$50N^2+295N+330$	$300N^2+1115N+925$	$300N^2+940N+640$

Table 7.3 gives the optimum solutions with different formulations. It is seen that alternative formulations 2 and 3 find the same optimum design, while alternate formulation 1 and the conventional formulation converge to a slightly different solution. The optimum solution obtained here is slightly better than that available in the literature.

7.8.3 Discussion of results

1. Number of time steps

It is obvious that the number of time steps used in the numerical solution process can affect the final solution and performance of the formulations. If the step size is too large, the time-dependent constraints may have larger violation between the grid points, and the optimal solution will not be accurate. If the step size is too small, the sizes of the alternative formulations become very large which requires additional calculations and computer storage. To evaluate the performance of various formulations, a few different grid sizes are tried for the two examples and various results are summarized in Tables 7.5, 7.6 and 7.7. Table 7.5 lists the final objective function values, Table 7.6 gives the numbers of iterations, and Table 7.7 gives the CPU efforts for different grid sizes. Table 7-5 shows that all the formulations converged essentially to the same optimum solution for Example 1 with N = 50, 300 and 500. For Example 2, all the formulations converged to a slightly lower objective function value for N = 100 compared to the solution with N = 300 and 500. This indicates that there was some violation of the constraints between the grid points when N = 100 was used. Table 7.7 shows that as the number of grid points is increased the computational effort with all the formulations also increases; the increase being more dramatic for the alternative formulations. It is also observed that for Example 2, Alternate 1 is about 3 times faster with N = 100 and about 2 times faster with N = 300 compared to the Conventional formulation. With N = 500, the conclusion is reversed. Alternate 2 and 3 are more efficient than the Conventional formulation only for N = 100. In any case all the alternative formulations converged to an optimum solution, with Alternate 1 requiring the least computational effort.

Table 7.5 Final objective results for design examples

Example Problem	**No. of Time Intervals (N)**	**Conventional**	**Alternate 1**	**Alternate 2**	**Alternate 3**
1-DOF	50	0.59725	0.59752	0.59735	0.59752
	300	0.59725	0.59726	0.59726	0.59726
	500	0.59725	0.59725	0.59725	0.59725
5-DOF	100	252.81	251.24	252.57	251.24
	300	254.56	254.69	254.38	254.38
	500	254.71	254.74	254.96	254.64

Table 7.6 Numbers of iterations for design examples

Example Problem	No. of Time Intervals (N)	Conventional	Alternate 1	Alternate 2	Alternate 3
1-DOF	50	4	5	16	6
	300	4	8	15	3
	500	4	9	15	4
5-DOF	100	23	23	10	24
	300	29	27	23	20
	500	22	18	23	38

Table 7.7 Computing efforts for different formulations (s)

Example Problem	No. of Time Intervals (N)	Conventional	Alternate 1	Alternate 2	Alternate 3
1-DOF	50	0.04	0.03	0.06	0.05
	300	0.25	0.64	1.35	0.75
	500	0.45	1.62	2.11	2.00
5-DOF	100	13.62	4.30	4.80	10.99
	300	23.19	12.00	45.99	49.31
	500	17.77	74.55	171.26	117.72

2. Scaling of variables

An important point that needs to be noted here is that the alternative formulations include different types of variables, which have different orders of magnitudes. Therefore scaling of some of the variables is necessary to reduce numerical ill-conditioning. In Alternate 2 and 3, velocity and acceleration variables are normalized by positive numbers. These normalizers are comparable to the velocity and acceleration limits. If the generalized displacements have large difference in their orders of magnitude, e.g., rotations and translations, some variables, such as the rotations may also need to be scaled. This is true for the rotational degrees of freedoms in Example 2. The current scaling option in SNOPT works fine only if a good starting point is provided or the problem is not too nonlinear. Efficient automatic scaling procedures need to be developed and incorporated into the alternative formulations.

3. Global solution

The five-DOF example problem has been solved by starting from several different points to determine all the local minimum points. All the starting points converged to one of the two objective function values (254.69 and 254.38). These objective function values are almost the same but have slightly different design variable values, especially the damping parameter c_1 for the driver's seat. Apparently the optimum cost function is insensitive with respect to this parameter. The two solutions can be considered essentially as global optima for the problem.

Table 7.8 Advantages and disadvantages of two formulations

Formulation	Advantages	Disadvantages
Conventional	1. Small optimization problems. 2. Equations of motion are satisfied at each iteration; intermediate solutions are usable. 3. Error in the solution of DEs can be controlled.	1. Equations of motion must be integrated at each iteration; a good DE integrator is needed. 2. Constraints are implicit functions of the variables; design sensitivity analysis must be performed. 3. Implementation is more tedious. 4. The optimization problem is always dense.
Alternative	1. Equations of motion are not integrated at each iteration; no DE integrator is needed. 2. Formulations are explicit in terms of variables; design sensitivity analysis is not needed. 3. Many constraints become linear or simple bounds on variables. 4. Jacobians and Hessians are sparse. 5. Implementation is easier.	1. Intermediate solutions are not usable. 2. Error in the state variables cannot be controlled. 3. Optimization algorithms for large-scale problems must be used. 4. For efficiency, advantage of sparsity of the Jacobians and Hessians must be utilized. 5. Scaling of optimization variables is needed.

4. Advantages and disadvantages of formulations

The advantages and disadvantages of the conventional and alternative formulations are listed in Table 7.8, and the differences between the alternative formulations are discussed in Table 7.9. Note that it is not necessary to include velocities and accelerations as variables to obtain explicit expressions for constraints and Jacobians. Their inclusion gives more optimization variables and constraints, and requires more data storage; however, their inclusion simplifies the constraint expressions and computer implementations. Some constraints become linear or simple bounds on the variables, such as the constraints on the velocities and accelerations. The gradients of the linear constraints in the alternative formulations can be programmed independently and called only once in the solution process, resulting in fewer calculations.

Table 7.9 Advantages and disadvantages of alternative formulations

Formulation	Advantages	Disadvantages
Alternate 1	1. Fewer optimization variables and constraints.	1. Derivative calculation and implementation is slightly more tedious. 2. Denser Jacobians and Hessians.
Alternate 2 & 3	1. Very sparse Jacobians and Hessians. 2. Implementation is very straightforward. 3. Velocity or acceleration constraints become linear or simple bounds.	1. Larger numbers of variables and constraints. 2. Larger number of non-zero elements in Jacobians.

7.9 Summary

Three alternative formulations for optimization of transient dynamic mechanical system were proposed and evaluated. Different state variables were treated as optimization variables in the formulations, i.e., generalized displacements, velocities and accelerations. Therefore the analysis equations (DEs) could be treated as equality constraints in the optimization process. By introducing more variables into the formulations, the forms of the constraints and their derivatives were changed. All functions of the formulations became explicit in terms of the optimization variables. Derivatives of the functions with respect to the variables were computed explicitly. The formulations were implemented with a sparse NLP code for evaluation. Different time steps were tested. The alternative formulations had more variables and constraints, although the constraints had simpler form compared to the conventional formulation. Therefore an optimization algorithm for large numbers of variables and constraints was used to solve the problem. Two example problems were solved extensively to study and evaluate the formulations. The solutions for the sample problems were also compared with those available in the literature.

Based on the current research the following conclusions are drawn:

1. All the proposed alternative formulations using simple finite difference approximations for the state variables worked well and obtained optimum solutions for the example problems.
2. The proposed alternate formulation 1 was more efficient than the formulations 2 and 3.
3. The proposed alternate formulation 1 was competitive and more efficient than the conventional formulation for reasonable number of time grid points.
4. The alternative formulations have potential for further development to optimize practical dynamic systems.

CHAPTER 8
TRANSIENT DYNAMIC RESPONSE OPTIMIZATION II

8.1 Introduction

This chapter is a continuation of the research initiated on the subject of optimization of transient dynamics problems in Chapter 7. Several alternative formulations for transient dynamic response optimization are described, analyzed and compared. A key feature of the formulations is that the state variables, in addition to the real design variables, are treated as independent variables in the optimization process. The state variables include different combinations of generalized displacements, velocities and accelerations. Formulations based on different discretization techniques for first-order and second-order differential equations are presented. Finite difference, Newmark's method and methods based on collocation are all discussed. Similar to the simultaneous analysis and design (SAND) approach used for optimization of structures subjected to static loads and the direct collocation/transcription method for optimal control, the equations of motion are treated as equality constraints. A major advantage of these formulations is that special design sensitivity analysis methods are no longer needed. However, the formulations have larger numbers of variables and constraints. Therefore sparsity of the problem functions must be exploited in all the calculations. Advantages and disadvantages of the formulations are discussed. The numerical results for an example problem and performance features of the formulations are compared. All the formulations converged to the optimum solution for the example problem. They were quite efficient for a smaller number of time grid points; however the computational times increased as the number of grid points was increased.

The major contributions of this chapter are as follows: (1) simultaneous formulations based on the SAND and direct collocation concepts are described, evaluated and compared, (2) some formulations based on the direct discretization of the second order form of the equations of motion (Newmark's method, Hermite and B-spline interpolations) are proposed, which may facilitate use of the existing simulation software in the optimization process, and (3) a modern and powerful optimization algorithm and associated software are used that take full advantage of the sparsity structure of the simultaneous formulations.

8.2 Dynamic Response Optimization Problem

As presented in Chapter 7, the basic dynamic response optimization problem is to determine design parameters related to stiffness and damping properties of the dynamic system, to achieve certain goals (e.g., minimization of a cost function, such as the maximum displacement or acceleration) while satisfying all the performance requirements.

Let $\mathbf{x}$ be an m dimensional vector to represent the design variables for the problem, which may directly include the mass, stiffness and damping parameters of the dynamic system, and $\mathbf{z}$ and $\mathbf{y}^T = [\mathbf{z} \quad \dot{\mathbf{z}}]$ be d and 2d dimensional vectors that represents the state variables for the problem. The equations of motion for a linear system are written in a second order form as Eq. (7.2.1), or a first order form, as of Eq. (7.2.2). It is worthwhile to examine both the second and the first order forms of the dynamics equations. Design requirements are imposed mostly as inequality constraints as in Eq. (7.2.5), or point-wise constraints, which need to be satisfied at each point of the entire time interval $t \in [0, T]$, as

$$\mathbf{g}(\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}, t) \le \mathbf{0} \tag{8.2.1}$$

The optimization problem is to find $\mathbf{x}$ to minimize the cost functional in Eq. (7.2.4) subject to the constraints of Eqs. (7.2.1) and (8.2.1), and other time-independent constraints. In this study, the conventional treatment is used where the time interval is divided into N subintervals (N+1 time grid points) and the constraints are imposed at each time grid point:

$$\mathbf{g}_i(\mathbf{x}, \mathbf{z}_i, \dot{\mathbf{z}}_i, \ddot{\mathbf{z}}, t_i) \le \mathbf{0}, \qquad i = 0, N \tag{8.2.2}$$

where $\mathbf{z}_i = \mathbf{z}(t_i)$, $\dot{\mathbf{z}}_i = \dot{\mathbf{z}}(t_i)$, and $\ddot{\mathbf{z}}_i = \ddot{\mathbf{z}}(t_i)$, and they are the generalized displacement, velocity and acceleration vectors at $t = t_i$. The length of the subinterval is Δt, and is defined as $\Delta t = T / N$. Note that the notation i = 0, N represents i = 0,1,2,…, N. In dynamic systems, the time-dependent inequality constraints in Eq. (8.2.1) may include the displacement, velocity and acceleration constraints.

If the first-order form of the DEs is used, state space variables $\mathbf{y}$ in Eq. (7.2.2), i.e., displacements $\mathbf{z}$ and velocities $\dot{\mathbf{z}}$ are used in the constraint expressions. The problem is to determine $\mathbf{x}$ to minimize the cost function in Eq. (7.2.4), subject to the first order DEs in Eq. (7.2.2), and

$$\mathbf{g}(\mathbf{x}, \mathbf{y}, \dot{\mathbf{y}}, t) \le \mathbf{0} \tag{8.2.3}$$

Note that Eqs. (7.2.2) and (8.2.3) are discretized into N+1 equations in the entire time interval $[0, T]$, and they represent a system of constraints as follows:

$$\mathbf{g}_i(\mathbf{x}, \mathbf{y}_i, \dot{\mathbf{y}}_i, t_i) \le \mathbf{0}, \qquad i = 0, N \tag{8.2.4}$$

where $\mathbf{y}_i = \mathbf{y}(t_i)$, and $\dot{\mathbf{y}}_i = \dot{\mathbf{y}}(t_i)$. If the problem is solved using the conventional formulation, where the optimization is carried out only in the space of design variables, the equations of motion in (7.2.1) and (7.2.2) need to be integrated to determine the system response and thus calculate various functions of the optimization problem. Since the constraint functions in this formulation are implicit functions of the design variables, implicit differentiation procedures need to be used to evaluate the gradients. Analytically, the *direct differentiation* or the *adjoint variable method* can be used (Hsieh and Arora 1984; Arora 1999). These procedures are difficult to implement with an existing simulation code, because the code needs to be recalled to solve for displacement, velocity and acceleration gradients or the adjoint vectors. Then, the gradients of the response functionals need to be assembled using the adjoint vectors or the displacement gradients.

8.3 Simultaneous Formulations Based on First Order DEs

In this section, simultaneous formulations based on the direct collocation/transcription method of optimal control are presented and discussed for optimal design problems. In these formulations, a set of discrete defect equations is derived from the first order DEs and imbedded into the optimization formulation. Starting form a system of first order differential equations, approximations in the time domain for the state variables are set up and collocation is enforced at certain time points. Direct collocation/transcription methods have been used for various optimal control problems, such as trajectory optimization, chemical process engineering, and robotic motion planning.

These formulations are based on different discretization and integration techniques. The first order state differential equations in Eq. (7.2.2) are approximated within each segment using an integration formula. The approximate integration formulation is then transformed to a set of algebraic equations, the so-called defect equations, which need to be set to zero to enforce the DEs. In this section, the following notation is used:

$$\mathbf{f}_i = \mathbf{f}(\mathbf{y}_i, \mathbf{x}, t_i) \tag{8.3.1}$$

where $\mathbf{f}_i$ is the state derivatives in Eq. (7.2.2) at $t = t_i$ and $\mathbf{y}_i = \mathbf{y}(t_i)$. For some formulations,

$$\mathbf{y}_{c,i} = \mathbf{y}(t_{c,i}); \quad \mathbf{f}_{c,i} = \mathbf{f}(\mathbf{y}_{c,i}, \mathbf{x}, t_{c,i}) \tag{8.3.2}$$

in which $t_{c,i} = (t_{i-1} + t_i)/2$, ($i$ = 1, N). $t_{c,i}$ is the center of the time interval, and $\mathbf{y}_{c,i}$ and $\mathbf{f}_{c,i}$ are the corresponding values of the state variables and derivatives.

8.3.1 Simultaneous formulation based on trapezoidal discretization (TR)

This formulation is based on the trapezoidal rule of numerical integration (Atkinson 1988). The general form of integration from time t_{i-1} to t_i is given as

$$\mathbf{y}_i = \mathbf{y}_{i-1} + \int_{t_{i-1}}^{t_i} \dot{\mathbf{y}} dt = \mathbf{y}_{i-1} + \int_{t_{i-1}}^{t_i} \mathbf{f} dt \tag{8.3.3}$$

If the trapezoidal approximation is considered, the defect equations can be set up (Herman and Conway 1995) as

$$\zeta_i = \mathbf{y}_i - \mathbf{y}_{i-1} - \frac{\Delta t}{2}(\mathbf{f}_i + \mathbf{f}_{i-1}), \qquad i = 1, N \tag{8.3.4}$$

where $\mathbf{f}_i$ is defined in Eq. (8.3.1).

In this formulation, the state variables $\mathbf{y}_i$ (i = 0, N) are chosen as the optimization variables. The defect equation in Eq. (8.3.4) needs to be satisfied to enforce the DEs; therefore, this formulation is to minimize the objective function in Eq. (7.2.4), subject to the equality constraints as

$$\mathbf{y}_i - \mathbf{y}_{i-1} - \frac{\Delta t}{2}\left(\mathbf{f}_i + \mathbf{f}_{i-1}\right) = \mathbf{0}, \qquad i = 1, N \tag{8.3.5}$$

and the discretized form of inequality constraints in Eq. (8.2.4). Note that time-dependent constraints on the state variable $\mathbf{y}$ can be imposed at the discrete time grid points, i.e., $\mathbf{y}_i$ (i = 0, N). This formulation is based on a linear approximation of the state gradient $\mathbf{f}$, and therefore a quadratic approximation of the state variables $\mathbf{y}$ in each time subinterval. It can also be derived from the forward difference approximation of the average velocities and accelerations in each time subinterval.

8.3.2 Simultaneous formulation based on compressed Hermite-Simpson discretization (CHS)

If the Simpson rule is used in the general form of numerical integration in Eq. (8.3.3) from time t_{i-1} to t_i, the defect equations are (Hargraves and Paris 1987; von Stryk and Bulirsch 1992):

$$\zeta_i = \mathbf{y}_i - \mathbf{y}_{i-1} - \frac{\Delta t}{6}\left(\mathbf{f}_i + 4\mathbf{f}_{c,i} + \mathbf{f}_{i-1}\right), \qquad i = 1, N \tag{8.3.6}$$

where $\mathbf{f}_i$ and $\mathbf{f}_{c,i}$ are defined in Eqs. (8.3.1) and (8.3.2), respectively, and

$$\mathbf{y}_{c,i} = \frac{1}{2}\left(\mathbf{y}_{i-1} + \mathbf{y}_i\right) + \frac{\Delta t}{8}\left(\mathbf{f}_{i-1} - \mathbf{f}_i\right) \tag{8.3.7}$$

The state variables $\mathbf{y}_i$ (i = 0, N) are chosen as the optimization variables. This formulation is to minimize the objective function defined in Eq. (7.2.4), subject to the inequality constraints in Eq. (8.2.4), and the equality constraints (approximated state equation) as

$$\mathbf{y}_i - \mathbf{y}_{i-1} - \frac{\Delta t}{6}\left(\mathbf{f}_i + 4\mathbf{f}_{c,i} + \mathbf{f}_{i-1}\right) = \mathbf{0}, \qquad i = 1, N \tag{8.3.8}$$

Note that time-dependent constraints on the state variables can be imposed at the discrete time grid points, or both the discrete time grid points and the center of each time subintervals, which is used in this study. In this formulation, the state variables $\mathbf{y}$ are chosen as continuous differentiable functions and defined piecewise as cubic polynomials between $\mathbf{y}_{i-1}$ and $\mathbf{y}_i$. The formulation can also be derived by collocating the first derivatives as equality constraints at the center point of each subinterval (Hargraves and Paris 1987; von Stryk and Bulirsch 1992).

8.3.3 Simultaneous formulation based on separated Hermite-Simpson discretization (SHS)

An alternate form can be obtained, if both the state variables $\mathbf{y}_i$ (i = 0, N) and $\mathbf{y}_{c,i}$ (i = 0, N-1) are treated as optimization variables (Betts and Huffman 1999). In this case, the equality constraints are Eqs. (8.3.8) and

$$\mathbf{y}_{c,i} - \frac{1}{2}\left(\mathbf{y}_{i-1} + \mathbf{y}_i\right) - \frac{\Delta t}{8}\left(\mathbf{f}_{i-1} - \mathbf{f}_i\right) = \mathbf{0}, \qquad i = 1, N \tag{8.3.9}$$

Equation (8.3.9) is derived from Eq. (8.3.7), and contains additional constraints that need to be satisfied. The time-dependent constraints on the state variables can be imposed at the discrete time grid points, or both the discrete time grid points and the center of each time subintervals.

8.4 Simultaneous Formulations Based on Second Order DEs

Although a large number of accurate, high-order or multi-step numerical methods are available to solve the system of first order DEs (Atkinson 1988), the analysis of practical structural and mechanical dynamics also heavily relies on those single-step direct time integration methods that solve the second order DEs directly (Bathe 1982; Chopra 2001). These methods are usually more efficient, and they are widely used in many commercial codes. Therefore, the development of simultaneous formulations combined with the second order form of the DEs will bring benefit to use these available codes directly. In the next section, some of these formulations are presented; they are based on Newmark's method, central difference method, and Hermite and cubic B-spline interpolations.

8.4.1 Simultaneous formulations based on Newmark's method (Newmark)

The Newmark's method is one of the most popular single step direct integration methods for second order DEs. Based on the assumption that the acceleration is linear within each time step, the following equations can be set up:

$$\mathbf{z}_{i+1} = \mathbf{z}_i + \Delta t\dot{\mathbf{z}}_i + \left(\tfrac{1}{2} - \beta\right)\Delta t^2\ddot{\mathbf{z}}_i + \beta\Delta t^2\ddot{\mathbf{z}}_{i+1} \tag{8.4.1}$$

$$\dot{\mathbf{z}}_{i+1} = \dot{\mathbf{z}}_i + (1-\gamma)\Delta t\ddot{\mathbf{z}}_i + \gamma\Delta t\ddot{\mathbf{z}}_{i+1} \tag{8.4.2}$$

Two widely used cases of Eqs. (8.4.1) and (8.4.2) are as follows:

$\gamma = \tfrac{1}{2}$ and $\beta = \tfrac{1}{4}$, it is the average acceleration method.

$\gamma = \tfrac{1}{2}$ and $\beta = \tfrac{1}{6}$, it is the linear acceleration method.

Note that when $\gamma = \frac{1}{2}$ and $\beta = 0$, Eqs. (8.4.1) and (8.4.2) can be reduced to very simple form, and this is in fact the central difference method, also the constant acceleration method. For general cases when $\beta \neq 0$, it is not easy to use the displacements to express the velocities and accelerations. When $\beta = 0$, it is possible to express the velocities and accelerations with respect to the displacements in a simple form. The central difference method will be discussed in the next section. Two possible simultaneous formulations based on Newmark's method in Eqs. (8.4.1) and (8.4.2) are discussed in this section.

1. Design variables, displacements and velocities as optimization variables (Newmark-1)

If displacements and velocities are treated as optimization variables, accelerations can be expressed in terms of the displacements and velocities. Therefore, an explicit form can be obtained for the dynamic optimization problem. The problem is to determine $\mathbf{x}$, $\mathbf{z}$ and $\dot{\mathbf{z}}$ (*i* = *-1*, *N+1*) to minimize the cost function of Eq. (7.2.4), subject to the state equations (7.2.1), and constraint in Eq. (8.2.2). More equality constraints are needed due to the introduction of additional variables. Rearranging the Newmark's equations in Eqs. (8.4.1) and (8.4.2), the accelerations are expressed as follows:

$$\ddot{\mathbf{z}}_i = \frac{1}{\varphi}\left[-\gamma \mathbf{z}_i + \gamma \mathbf{z}_{i+1} - (\gamma - \beta)\Delta t \dot{\mathbf{z}}_i - \beta \Delta t \dot{\mathbf{z}}_{i+1}\right] \tag{8.4.3}$$

$$\ddot{\mathbf{z}}_{i+1} = \frac{1}{\varphi}\left[(1-\gamma)\mathbf{z}_i - (1-\gamma)\mathbf{z}_{i+1} + \left(\tfrac{1}{2} - \gamma + \beta\right)\Delta t \dot{\mathbf{z}}_i + \left(\tfrac{1}{2} - \beta\right)\Delta t \dot{\mathbf{z}}_{i+1}\right] \tag{8.4.4}$$

where $\varphi = \left(\frac{1}{2}\gamma - \beta\right)\Delta t^2$. The equality constraints among the variables $\mathbf{z}_i$, and $\dot{\mathbf{z}}_i$ can be constructed by substituting Eqs. (8.4.3) or (8.4.4) into (8.4.2) or (8.4.3), as

$$\begin{aligned}&(1-\gamma)\mathbf{z}_i + (2\gamma - 1)\mathbf{z}_{i+1} - \gamma \mathbf{z}_{i+2} \\ &+ \left(\tfrac{1}{2} - \gamma + \beta\right)\Delta t \dot{\mathbf{z}}_i + \left(\tfrac{1}{2} - 2\beta + \gamma\right)\Delta t \dot{\mathbf{z}}_{i+1} + \beta \Delta t \dot{\mathbf{z}}_{i+2} = \mathbf{0}\end{aligned} \tag{8.4.5}$$

Note that in Eqs. (8.4.3) and (8.4.4), $\frac{1}{2}\gamma - \beta \neq 0$. This means that the average acceleration method where $\gamma = \frac{1}{2}$ and $\beta = \frac{1}{4}$, is not applicable here. The basic idea of the formulations based on the direct discretization of second order DEs is that the system of DEs is discretized as equality constraints. However, it is not possible to express $\ddot{\mathbf{z}}_i$ in terms of $\mathbf{z}_i$ and $\dot{\mathbf{z}}_i$, from Newmark's equations in Eqs. (8.4.1) and (8.4.2). Therefore, it is not easy to express and evaluate the equality equations in Eq. (7.2.1).

2. Design variables, displacements, velocities and accelerations as optimization variables (Newmark-2)

From Eq. (7.2.1), it is seen that other simultaneous formulations are possible. If the displacements $\mathbf{z}$, velocities $\dot{\mathbf{z}}$ and accelerations $\ddot{\mathbf{z}}$ are treated as variables simultaneously, another explicit form is obtained. The problem is to determine $\mathbf{x}$, $\mathbf{z}$ (*i* = *-1, N+1*), $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$ (*i* = *0, N*) to minimize the cost function of Eq. (7.2.4), subject to the equations of motion (7.2.1), and discretized constraints in Eq. (8.2.2). It is obvious that the equality constraints in Eq. (7.2.1) are explicit with respect to the optimization variables $\mathbf{x}$, $\mathbf{z}$, $\dot{\mathbf{z}}$ and $\ddot{\mathbf{z}}$. However, more equality constraints are needed, since the state variables are related to each other by the Newmark's equations in Eqs. (8.4.1) and (8.4.2). These need to be imposed in the formulation.

8.4.2 Simultaneous formulations based on central difference method (CD)

Three simultaneous formulations based on the central difference approximation have been presented and evaluated in Chapter 7. Details of these three formulations are not presented here. However, these formulations include displacement only (CD-1), displacement and velocity (CD-2), and displacement, velocity and acceleration (CD-3) as additional optimization variables, respectively. Finite difference relationships (Eqs. (7.4.3) and (7.4.4)) are used as equality constraints in the formulations, which makes the implementation of the algorithm quite simple.

8.4.3 Simultaneous formulation based on piecewise Hermite interpolation (Hermite)

The basic discretization scheme is as follows: the state variables $\mathbf{z}$ are chosen as continuous differentiable functions and defined as piecewise cubic polynomials between $\mathbf{z}_i$ and $\mathbf{z}_{i+1}$, with the state equations (7.2.1) satisfied at t_i and t_{i+1}. For $t \in [t_i, t_{i+1}]$, using a parameter $u \in [0,1]$, such that $t = t_i + u\Delta t$, the approximation of state variables $\mathbf{z}$ is

$$\mathbf{z}_a(t) = \sum_{k=0}^{3} \mathbf{c}_k^i u^k \tag{8.4.6}$$

where

$$\begin{aligned} \mathbf{c}_0^i &= \mathbf{z}_i \\ \mathbf{c}_1^i &= \Delta t \dot{\mathbf{z}}_i \\ \mathbf{c}_2^i &= -3\mathbf{z}_i - 2\Delta t \dot{\mathbf{z}}_i + 3\mathbf{z}_{i+1} - \Delta t \dot{\mathbf{z}}_{i+1} \\ \mathbf{c}_3^i &= 2\mathbf{z}_i + \Delta t \dot{\mathbf{z}}_i - 2\mathbf{z}_{i+1} + \Delta t \dot{\mathbf{z}}_{i+1} \end{aligned} \tag{8.4.7}$$

where $\Delta t = t_{i+1} - t_i$. The approximation function (8.4.6) of the state variables must satisfy the DEs (7.2.1) at the grid point t_i ($i = 0, N$).

The optimization problem is to determine $\mathbf{x}$, $\mathbf{z}$, and $\dot{\mathbf{z}}$ to minimize the cost function of Eq. (7.2.4), subject to the equality constraints in Eq. (7.2.1) at the discrete time grid points, and the discretized inequality constraints in Eq. (8.2.2) at the time grid point and mid-point of each interval, $t_{c,i} = (t_i + t_{i+1})/2$ (*i* = *0*, *N*-*1*). From Eq. (8.4.6),

$$\dot{\mathbf{z}}_a(t) = \sum_{k=1}^{3} \frac{k}{\Delta t} \mathbf{c}_k^i u^{k-1}; \qquad \ddot{\mathbf{z}}_a(t) = \sum_{k=2}^{3} \frac{k(k-1)}{\Delta t^2} \mathbf{c}_k^i u^{k-2} \tag{8.4.8}$$

At the grid point $t = t_i$,

$$\ddot{\mathbf{z}}_i = \ddot{\mathbf{z}}_a(t_i) = \frac{2}{\Delta t^2}\left(-3\mathbf{z}_i + 3\mathbf{z}_{i+1} - 2\Delta t \dot{\mathbf{z}}_i - \Delta t \dot{\mathbf{z}}_{i+1}\right) \tag{8.4.9}$$

A system of equality constraints is obtained at each time grid point, as

$$\mathbf{M}(\mathbf{x})\left[\frac{2}{\Delta t^2}\left(-3\mathbf{z}_i + 3\mathbf{z}_{i+1} - 2\Delta t \dot{\mathbf{z}}_i - \Delta t \dot{\mathbf{z}}_{i+1}\right)\right] + \mathbf{C}(\mathbf{x})\dot{\mathbf{z}}_i + \mathbf{K}(\mathbf{x})\mathbf{z}_i = \mathbf{F}_i(\mathbf{x}, t_i) \tag{8.4.10}$$

The relationships between the displacement and velocity variables are obtained from Eq. (8.4.9), as

$$3\mathbf{z}_i - 3\mathbf{z}_{i+2} + \Delta t\left(\dot{\mathbf{z}}_i + 4\dot{\mathbf{z}}_{i+1} + \dot{\mathbf{z}}_{i+2}\right) = \mathbf{0} \tag{8.4.11}$$

8.4.4 Simultaneous formulation based on cubic B-spline interpolation (B-spline)

If the final state variable history needs to be very smooth, cubic B-spline interpolation (Mortenson 1985) can be used. B-splines have many important properties such as continuity, differentiability, and local control. There are a number of ways to define the B-spline basis functions, and here the uniform B-spline is used. Let $T = \{t_0, t_1, .., t_{nk}\}$ be a non-decreasing sequence of real numbers, i.e., $t_i \le t_{i+1},\ i = 0, nk\text{-}1$. The t_i are called *knots*, and they are evenly spaced for a uniform B-spline. A cubic B-spline is defined as

$$z(t) = \sum_{j=0}^{nc} N_{j,4}(t) P_j; \qquad 0 \le t \le T \tag{8.4.12}$$

where the $\{P_j\}$, $j = 0, nc$ are the $(nc+1)$ *control points*, and the $\{N_{j,4}(t)\}$ are the cubic B-spline basis functions defined on the non-periodic knot vector ($(nk+1)$ knots). Using a parameter $u \in [0,1]$ defined as such that $t = t_i + u\Delta t$, the basis functions of a cubic B-spline are as follows

$$\begin{aligned}
N_{j,4}(u) &= \tfrac{1}{6}\left(-u^3 + 3u^2 - 3u + 1\right) \\
N_{j+1,4}(u) &= \tfrac{1}{6}\left(3u^3 - 6u^2 + 4\right) \\
N_{j+2,4}(u) &= \tfrac{1}{6}\left(-3u^3 + 3u^2 + 3u + 1\right) \\
N_{j+3,4}(u) &= \tfrac{1}{6}\left(u^3\right)
\end{aligned} \tag{8.4.13}$$

Since each segment of the curve is defined by four control points, for $t_i \le t_{i+1}$, Eq. (8.4.12) can be simplified as

$$z(u) = N_{i,4}(u)P_i + N_{i+1,4}(u)P_{i+1} + N_{i+2,4}(u)P_{i+2} + N_{i+3,4}(u)P_{i+3} \tag{8.4.14}$$

First and second order derivatives of the displacement are needed in the optimization procedure. The *k*th derivatives of a cubic B-spine curve can be easily obtained from Eq. (8.4.14), since only the basis functions are functions of time.

In this formulation, the control point vector $\mathbf{P}$ for each DOF is chosen as the optimization variables. This formulation is to minimize the objective function in Eq. (7.2.4), subject to the inequality constraints in Eq. (8.2.1), as

$$\mathbf{g}(\mathbf{x}, \mathbf{P}, t) \le \mathbf{0} \tag{8.4.15}$$

Note that the equilibrium equations in Eq. (7.2.1) can be expressed similarly in terms of $\mathbf{P}$ and treated as equality constraints in the formulation. The time-dependent constraints on the state variables can be imposed on the discrete time grid points.

8.5 Evaluation of Formulations

Table 8.1 shows the sizes of all the formulations. The following symbols are used in the table: m = number of elements in the design variable vector $\mathbf{x}$; N = number of time intervals (number of time grid points = $N + 1$); n = number of control points in a cubic B-spline; d = number of DEs in Eq. (7.2.1), or the dimension of vector $\mathbf{z}$; e = number of constraints in $\mathbf{g}$; however, some of the inequality constraints in some alternative formulations become simple bounds on the variables.

Table 8.1 Number of variables and constraints for different formulations

Formulations		Variables	No. of Variables	No. of Equality Constraints	No. of Inequality Constraints
Conventional		$\mathbf{x}$	m	0	$e(N+1)$
1st order DEs	**TR**	$\mathbf{x}, \mathbf{y}$	$m+2dN+2d$	$2dN$	$e(N+1)$
	CHS	$\mathbf{x}, \mathbf{y}$	$m+2dN+2d$	$2dN$	$e(2N+1)$
	SHS	$\mathbf{x}, \mathbf{y}, \mathbf{y}_c$	$m+4dN+2d$	$4dN$	$e(2N+1)$
2nd order DEs	**Newmark-1**	$\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}$	$m+2dN+6d$	$2dN+2d$	$e(N+1)$
	Newmark-2	$\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}$	$m+3dN+9d$	$3dN+5d$	$e(N+1)$
	CD-1	$\mathbf{x}, \mathbf{z}$	$m+dN+3d$	$dN+d$	$e(N+1)$
	CD-2	$\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}$	$m+3dN+6d$	$3dN+4d$	$e(N+1)$
	CD-3	$\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}$	$m+3dN+5d$	$3dN+3d$	$e(N+1)$
	Hermite	$\mathbf{x}, \mathbf{z}, \dot{\mathbf{z}}$	$m+2dN+4d$	$2dN+d$	$e(2N+1)$
	B-spline	$\mathbf{x}, \mathbf{P}$	$m+nd$	$dN+d$	$e(N+1)$

Note that the alternative formulations do not require the equations of motion in Eq. (7.2.1) to be satisfied exactly at each iteration of the optimization process. They need to be satisfied only at the final solution point. This has an advantage if instabilities occur or no solution exists for certain designs in the design space. Also, unnecessary simulations of the system are avoided at intermediate designs, where it might be difficult to obtain a solution.

Table 8.2 Advantages and disadvantages of different formulations

Formulations		Advantages	Disadvantages
Conventional		• Smaller NLP problems. • DEs are satisfied at each iteration; intermediate solutions are usable. • Error in the solution of DEs can be controlled.	• Design sensitivity analysis must be performed. • Dense Jacobian and Hessian matrices. • DEs must be integrated; a good DEs integrator is needed.
1st order DEs	**TR**	• Implementation is straightforward. • Velocity constraints become linear or simple bounds.	• Larger number of variables and constraints. • Acceleration constraints become complex.
	HS	• Good smoothness • Smaller NLP problems • Velocity constraints become linear or simple bounds.	• Implementation is not straightforward. • Acceleration constraints become complex.
2nd order DEs	**CD**	• Very sparse Jacobians and Hessian. • Implementation is very straightforward. • Velocity or acceleration constraints become linear or simple bounds.	• Larger number of variables and constraints. • Larger number of non-zero elements in Jacobians.
	Newmark	• Very sparse Jacobians and Hessian. • Implementation is very straightforward. • Velocity or acceleration constraints become linear or simple bounds. • Very general	• Larger number of variables and constraints. • Larger number of non-zero elements in Jacobians.
	Hermite	• Good smoothness. • Smaller NLP problems • Velocity or acceleration constraints become linear or simple bounds.	• Implementation is not straightforward.
	B-spline	• Good smoothness. • Smaller NLP problems • Displacement, velocity or acceleration constraints are linear.	• Implementation is not straightforward. • Displacement, velocity or acceleration constraints are not simple bounds.

The differences between the simultaneous formulations are discussed in Table 8.2. For the simultaneous formulations based on the first order DEs, the number of time grid points is usually not very large; therefore, the resulting NLP may not need to be too large. Since most of the methods are based on polynomial interpolations between grid points, only a reasonable number of grid points are needed. Moreover, these methods can provide more accurate solutions, provided that multi-step or higher order methods are used. As can be seen that the defect equations (i.e., Eqs. (8.3.5) and (8.3.8)) involve repeated substitutions of the equations of motion (7.2.2), if multi-step methods of discretization or higher order polynomial interpolations are used. Thus implementation becomes tedious. Sparsity patterns of those formulations are hard to identify and define. For the constraints involving accelerations, the expressions for some cases become complex, since the equations of motion are embedded in the expressions for accelerations.

With the second order DEs, since the equations are directly discretized and treated as equality constraints, the complexity of the constraint equations is well defined. The implementation and sparsity pattern are very straightforward. The methods treat the acceleration constraints more efficiently. For the finite difference-based methods, e.g., central difference or Newmark’s methods, since a small time step is used to guarantee convergence and stability, the number of grid points *N* is usually very large, resulting in large numbers of variables and constraints in the formulations. Large-scale NLP solution algorithms with sparse matrix capabilities are required to solve the simultaneous formulations efficiently. These methods are more suitable for dynamic response that may not necessarily be smooth. Some aspects of the simultaneous formulations are discussed in details in Section 8.6, when a design example is solved.

8.6 A Numerical Example

All the simultaneous formulations developed in Sections 8.3 and 8.4 are applied to a dynamic mechanical example for evaluation. All the simultaneous formulations are solved using the sparse SQP algorithm in SNOPT (Gill *et al.* 2002), while the conventional formulation is solved using the dense SQP solver in SNOPT. A Dell PC with 2.53 GHz P4 processor and 1.0 GB RAM is used for running the programs and recording the relative CPU times. Each solution case of the example problem was run several times with different starting point and the shortest time was recorded. Results of the examples are listed and compared. Advantages and disadvantages of the formulations are discussed.

8.6.1 5-DOF vehicle suspension system

The five degree of freedom vehicle suspension system (Haug and Arora 1979) is presented in Chapter 7 and shown in Figure 7.3. The equations of motion and design requirements for the system are given there.

Table 8.3 Sizes of the design problem for different formulations

Formulations		No. of Variables	No. of Constraints	No. of Non-zero Elements in Jacobian (w. Sparsity)	No. of Non-zero Elements in Jacobian (w/o Sparsity)
Conventional		*7*	*6N+6*	*42N+42*	*42N+42*
1st order DEs	**TR**	*10N+17*	*15N+5*	*141N+27*	$150N^2+305N+85$
	CHS	*10N+17*	*22N+5*	*349N+27*	$220N^2+424N+85$
	SHS	*20N+17*	*30N+5*	*335N+27*	$600N^2+610N+85$
2nd order DEs	**Newmark-1**	*10N+37*	*15N+15*	*115N+115*	$150N^2+705N+555$
	Newmark-2	*15N+52*	*20N+30*	*119N+164*	$300N^2+1490N+1560$
	CD-1	*5N+22*	*10N+15*	*92N+102*	$50N^2+295N+330$
	CD-2	*15N+37*	*20N+25*	*111N+126*	$300N^2+1115N+925$
	CD-3	*15N+32*	*20N+20*	*109N+109*	$300N^2+940N+640$
	Hermite	*10N+27*	*22N+10*	*160N+85*	$220N^2+694N+270$
	B-spline	*5N+22*	*12N+22*	*148N+178*	$60N^2+374N+484$

8.6.2 Discussion of results

1. Sizes of optimization problem

Table 8.3 lists the sizes of the problems for different formulations. For the conventional formulation, the size of the design problem is well defined having 7 design variables. This is a quite small design problem in terms of numerical optimization. In the alternative formulations, there are large numbers of variables, depending on the number of time grid points chosen. If the number of grid points is large, the resulting optimization problem may include hundreds and even thousands of variables, which may make the problem ill-conditioned sometimes. This is the major disadvantage of these alternative formulations. Even though the alternative formulations are quite sparse, the storage requirement is still larger than that for the conventional formulation for this example, as seen in Tables 8.3.

Table 8.4 Final results of design example (N = 300)

Formulations		E	k_1	k_2	k_3	c_1	c_2	c_3
Conventional (Haug and Arora 1979)		257.40	50.00	200.00	241.90	12.89	77.52	80.00
	Conventional	254.56	50.00	200.00	200.00	45.45	77.35	80.00
1st order DEs	**TR**	254.91	50.00	200.00	200.00	20.10	76.95	80.00
	CHS	254.82	50.00	200.00	200.00	50.00	77.39	80.00
	SHS	254.74	50.00	200.00	200.00	31.42	77.19	80.00
2nd order DEs	**CD-1**	254.69	50.00	200.00	200.00	19.97	76.99	80.00
	CD-2	254.38	50.00	200.00	200.00	45.23	77.39	80.00
	CD-3	254.38	50.00	200.00	200.00	45.23	77.39	80.00
	Newmark-1 (Const. ac.)	254.69	50.00	200.00	200.00	19.97	76.99	80.00
	Newmark-1 (Linear ac.)	254.84	50.00	200.00	200.00	20.06	76.96	80.00
	Newmark-2 (Const. ac.)	254.69	50.00	200.00	200.00	19.97	76.99	80.00
	Newmark-2 (Linear ac.)	254.56	50.00	200.00	200.00	45.61	77.36	80.00
	Newmark-2 (Ave. ac.)	254.91	50.00	200.00	200.00	20.10	76.95	80.00
	Hermite	254.55	50.00	200.00	200.00	28.44	77.15	80.00
	B-spline	254.65	50.00	200.00	200.00	50.00	77.41	80.00

2. Number of time steps

It is obvious that the number of time steps used in the numerical solution process can affect the final solution and performance of the formulations. If the time step size is too large, the time-dependent constraints may have larger violations between the grid points, and the optimal solution will not be accurate. If the step size is too small, the sizes of the alternative formulations become very large which requires additional calculations and data storage. To evaluate the performance of various formulations, a few different grid sizes (N = 100, 300, and 500) are tried for the example and the final optimization results are summarized in Tables 8.4 and 8.5, and the numbers of iterations and CPU efforts are given in Table 8.6.

Table 8.5 Final objective values of design example

	Formulations	***N*=100**	***N*=300**	***N*=500**
	Conventional	252.81	254.56	254.71
1st order DEs	**TR**	253.09	254.91	254.73
	CHS	255.00	254.81	254.78
	SHS	254.31	254.74	254.85
2nd order DEs	**CD-1**	251.24	254.69	254.74
	CD-2	252.57	254.38	254.96
	CD-3	251.24	254.38	254.64
	Newmark-1 (Const. ac.)	251.24	254.69	254.64
	Newmark-1 (Linear ac.)	252.45	254.84	255.01
	Newmark-2 (Const. ac.)	251.24	254.69	254.96
	Newmark-2 (Linear ac.)	252.45	254.56	254.80
	Newmark-2 (Ave. ac.)	253.09	254.91	254.73
	Hermite	252.51	254.55	255.02
	B-spline	251.95	254.65	254.95

Table 8.4 gives the final optimum solutions with different formulations for N = 300. The final objective values of the design example are listed in Table 8.5. It is seen that various simultaneous formulations work well and optimal solutions are obtained. The optimum solutions obtained by the conventional and simultaneous formulations in this study are slightly better than those available in the literature. Most of the formulations converged to a slightly lower objective function value for N = 100 compared to the solution with N = 300 and 500. This indicates that there was some violation of the constraints between the grid points when N = 100 was used.

Table 8.6 Numbers of iterations and computing efforts for different formulations

Formulations		Numbers of Iterations			CPU Time (s)		
		N=100	N=300	N=500	N=100	N=300	N=500
Conventional		23	29	22	13.6	23.2	17.8
1st order DEs	**TR**	21	32	43	4.8	51.3	117.5
	CHS	34	190	18	24.6	323.9	465.3
	SHS	20	27	38	32.4	288.0	539.5
2nd order DEs	**CD-1**	23	27	18	4.3	12.1	74.4
	CD-2	10	23	26	4.9	46.4	172.8
	CD-3	24	20	38	10.9	49.1	116.8
	Newmark-1 (Const. ac.)	22	30	35	5.9	38.4	180.3
	Newmark-1 (Linear ac.)	26	42	31	7.3	51.8	227.8
	Newmark-2 (Const. ac.)	48	22	12	8.4	54.8	64.5
	Newmark-2 (Linear ac.)	30	37	21	7.8	84.5	142.4
	Newmark-2 (Ave. ac.)	20	21	29	9.0	58.0	171.7
	Hermite	37	28	17	8.3	211.5	406.3
	B-spline	30	26	24	6.1	72.0	209.3

Since different approximations of the state variables are introduced in the simultaneous formulations, the quality of the final solutions is different. Note that the quality of the final solution depends on the approximation made between the state variables. In this work, the finite difference based methods, such as CDs and Newmarks provide similar performance in terms of the optimal solutions, numbers of iterations and computing efforts. In order to have good results, the number of grid points usually needs to be large. The second order Hermite and Bspline formulations do not provide better accuracy with a small number of time grid points, because the equations of motion are only imposed at the discrete time grid points.

For the simultaneous formulations based on the first order DEs, such as CHS and SHS, the optimal solutions with N = 100 are already very good. This can be explained by the piecewise cubic polynomial approximations for both the generalized displacements and velocities, and the imposing of constraints at the center of each time step. These provide accurate approximations for the state variables; therefore, for CHS and SHS to reach similar accuracy of optimal solutions as other alternative formulations, a smaller number of time steps *N* is needed. Although SHS is easier to implement than CHS, due to the inclusion of more variables, the overall performances of CHS and SHS are similar. TR usually performs better than CHS or SHS with respect to the computational efforts for various *N*, because of simpler expressions of constraints and the exclusion of constraints or variables at the center of time steps. However since the approximation is not as good as those by CHS or SHS, the optimal solution obtained by TR for *N* = 100 is not as good, either.

Table 8.6 contains comparison of the numbers of iterations and CPU efforts for different grid sizes. It shows that as the number of grid points is increased the computational effort with all the alternative formulations increases. Although the numbers of iterations needed for the alternative formulations are mostly in the range of 20~40, the CPU/iteration is quite large, and becomes larger as the size of the optimization problem increases. However, the CPU effort for the conventional formulation remains relatively constant for various *N*. It is seen that the alternative formulations are more efficient and require less CPU effort than the conventional formulation when the number of time grid points is smaller (*N* = 100), except for CHS and SHS. For *N* = 300, only CD-1 is more efficient than the conventional formulation, and with *N* = 500, the conventional formulation is the most efficient one among all the formulations. This can be explained by the sizes of the formulations. When the number of time grid points becomes large, there are much more optimization variables and constraints in the alternative formulations which makes the convergence slower. Also, the number of computations for each iteration increases. Apparently, the QP solver for calculation of the search direction becomes less efficient as the numbers of variables and constraints increase. Therefore, better QP solvers for large sparse problems need to be developed.

3. Advantages and disadvantages of formulations

The main advantages of the simultaneous formulations for dynamic systems are as follows. (i) The equations of motion for the system need not be integrated explicitly. They need to be satisfied only at the final solution point. This has advantage if instabilities occur or no solution exists for certain designs in the design space. Also, unnecessary simulations of the system are avoided at intermediate designs, where it might be difficult to obtain a solution. (ii) Design sensitivity analysis of the systems (which is quite tedious and difficult to implement) is not needed since all the problem functions are explicit in terms of the variables. (iii) The inclusion of more state variables in the formulations simplifies the constraint expressions and computer implementations. Some constraints may become linear or simple bounds on the variables, such as the constraints on the displacements, velocities and accelerations. The gradients of the linear constraints in the simultaneous formulations can be programmed independently and calculated only once in the solution process.

A major disadvantage of the simultaneous formulations is that the optimization problem is very large and large-scale sparse NLP methods are needed. Also, since the simultaneous formulations include different types of variables, which have different orders of magnitudes, scaling of some of the variables is necessary to reduce numerical difficulties.

4. Scaling of variables

In most alternative formulations, velocity and acceleration variables are normalized by positive numbers. These normalizers are comparable to the velocity and acceleration limits. If the generalized displacements have large difference in their orders of magnitude, e.g., rotations and translations, some variables, such as the rotations may also need to be scaled. This is true for the rotational degrees of freedoms in the example. The current scaling option in SNOPT works fine only if a good starting point is provided or the problem is not too nonlinear. Efficient automatic scaling procedures need to be developed and incorporated into the alternative formulations.

5. Other formulations

It is clear that other simultaneous formulations are possible. These are based on different discretization techniques of the first or second-order DEs, such as the Runge-Kutta formula for numerical solution of first order DEs (Enright and Conway 1992; Betts 1998; Betts and Huffman 1999), and piecewise higher degree polynomial approximations of state variables for first order DEs (Herman and Conway 1995). However, these multi-step methods or higher degree of polynomials may result in significant complexity of numerical implementation for the simultaneous formulations, which is not desired. The application of these methods for transient dynamic optimization needs further evaluation.

8.7 Summary

Simultaneous formulations for optimization of transient dynamic mechanical systems were described and evaluated. Different state variables or their parametric approximations were treated as optimization variables in the formulations, i.e., generalized displacements, velocities and accelerations. Therefore the discretized state equations (DEs), either in the first or second order forms could be treated as equality constraints in the optimization process. By introducing more variables into the formulations, the forms of the constraints and their derivatives were changed. The formulations were implemented with a sparse NLP code for evaluation. The simultaneous formulations had more variables and constraints, although the constraints had simpler form compared to the conventional formulation. Therefore an optimization algorithm for large numbers of variables and constraints was used to solve the problem. The solutions for one sample problem were obtained and compared. Based on this work, the following conclusions are drawn:

1. All the proposed alternative formulations obtained optimum solutions for the example problem.
2. Formulations based on Hermite-Simpson discretization of first order DEs gave better solutions with the smaller number of time grid points.
3. Finite difference based methods were easier to implement than those based on first order DEs; However, they usually required a larger number of time grid points for better solutions.
4. In terms of CPU times, most alternative SAND formulations outperform the conventional formulation for a smaller number of time grid points.
5. When the problem size is too large, the sparse QP subproblem solver became slower to converge, resulting in much more computational effort than the conventional formulation.
6. More efficient solution methods of sparse QP subproblems and parallel algorithms for the alternative formulations need to be studied, developed and

combined for broader practical applications of these formulations.

CHAPTER 9
DIGITAL HUMAN DYNAMIC MOTION PREDICTION

9.1 Introduction

Virtual human modeling and simulation has attracted considerable attention in recent years. Predicting a human gait motion is to solve a non-trivial dynamics problem, since joint rotations and torque profiles as well as ground reaction forces are all unknowns. In reality, humans can walk in an infinite variety of ways; therefore, there is no unique solution to the prediction of human gait motion. Several different research avenues have been explored in the literature. In the robotics field, a fast solution of the dynamics problem is needed to facilitate real-time motion and control. Thus, the basic information that needs to be generated is the motion trajectories of various segments and the control torques. In the area of biomechanics, more natural and realistic human motions with complex musculoskeletal models have been studied and analyzed. Muscle tendon force, stress, fatigue, and injury are all active areas of research.

The motion planning methods in robotics can be broadly categorized into those that are based on stability only without optimization of any performance measure, and those that use optimization to solve for optimal trajectories and torques. Motion capture, zero moment point (ZMP)-based trajectory generation, and inverted pendulum methods all belong to the first category. The motion capture approach is experiment-based; the motion of a subject is recorded by identifying the marker positions in the Cartesian coordinates. Then the joint angle trajectories are generated by using the recorded data and inverse kinematics. This approach is limited by the accuracy of the experimental data.

Also, the simulated motion is subject-specific. Statistical methods are developed based on the pre-established motion database and they do not involve equations of motion (Furusho and Masubuchi 1986; Faraway *et al.* 1999). The ZMP-based trajectory generation method enforces the stability of the mechanism in which leg motion of the biped and location of the ZMP are pre-planned (Yamaguchi *et al.* 1999; Nishiwaki and Kagami 2002). Upper-body motion is generated to satisfy the ZMP trajectory; however, this may result in excessive upper-body motion. The key point of this approach is that the dynamics equations are used only to formulate the stability (ZMP) constraint condition rather than generation of the entire trajectory directly. Since additional equality constraints on gait parameters, such as hip/limb motion or the ZMP trajectory as a function of time are imposed, it is recommended to minimize the number of artificial constraints used since they may be too restrictive to adapt to changes in the mission goals, the anthropometric data, or the environment conditions. The inverted pendulum model is often used to solve the walking problem because biped walking can be treated as an inverted pendulum. The advantages of this method are its simplicity and fast solvable dynamics equations (Park and Kim 1998). However, it also suffers from an inadequate dynamics model that cannot generate natural and realistic human motion.

Optimization-based trajectory generation is aimed at more realistic and natural humanoid motion. Many human-featured criteria can be simultaneously considered rather than only the stability. For digital human simulations, the objective functions represent human performance measures, and optimization methods are used to solve for the feasible joint motion profiles that optimize the objective functions and satisfy the necessary constraints (Chevallereau and Aoustin 2001; Bessonnet *et al.* 2002, 2005;

Saidouni and Bessonnet 2003). Lo *et al.* (2002) used quasi-Newton nonlinear programming techniques to determine the human motion that minimizes the actuating joint torques. The design variables were the control points for the cubic B-spline approximation of joint angle profiles. Chevallereau and Aoustin (2001) planned a walking and running motion using the Pontryagin Maximum Principle to determine the coefficients of a polynomial approximation for profiles of the pelvis translations and joint angle rotations. Saidouni and Bessonet (2003) solved for cyclic, symmetric gait motion of a nine degree of freedom (DOF) model that moves in the sagittal plane. The control points for the B-spline curves along with the time durations for the gait stages are optimized to minimize the actuating torque energy. By adopting the time durations as design variables both the motion for the single support and for the double support are simultaneously optimized.

Skeletal models are used quite naturally in the robotics area. They have also been used in human locomotion modeling due to their relative simplicity and computational efficiency (Chevallereau and Aoustin 2001; Bessonnet *et al.* 2002, 2005; Saidouni and Bessonnet 2003). In biomechanics literature, optimization-based methods have been used to simulate human motion with complex musculoskeletal models (Yamaguchi and Zajac 1990; Pandy *et al.* 1992; Anderson and Pandy 2001). Muscle groups are included in the model using Hill-type elements. The examples with musculoskeletal models include an 8 DOF model by Yamaguchi and Zajac (1990) to restore unassisted natural gait to paraplegics and a model with 23 DOF and 54 muscles for normal symmetric walking on level ground by Anderson and Pandy (2001). Thelen *et al.* proposed a muscle control algorithm that used a static optimization method with feed-forward and feedback controls to obtain the desired kinematics trajectories of a musculoskeletal model (Thelen *et al.* 2003). The resulting simulations matched well with the patterns of body-segmental displacements, ground-reaction forces, and muscle activations obtained from experiments. The forward dynamics optimization problem with such musculoskeletal models is typically posed to minimize the metabolic energy expenditure per unit distance traveled. A set of terminal posture constraints are often imposed to ensure repeatability of the gait cycle.

In the current chapter, the Denavit-Hartenberg (DH) method (Denavit and Hartenberg 1955) is used for modeling human skeletal links and dynamics. Meanwhile, various constraints are imposed to ensure that the optimal motion satisfies maximum joint angle limit, no ground penetration, dynamic stability, and foot-point locations. The human body dynamics is based on a recursive formulation that can also be used to calculate gradients efficiently. The equations of motion (EOMs) are discretized and finite dimensional approximation or parametric representation for the joint angles variables are defined, converting the simulation problem into a nonlinear programming (NLP) problem. Different formulations and discretization techniques are available. These techniques include finite difference, and piecewise polynomial and spline interpolations. In the current work, explicit integration of the equations of motion is avoided, which is very advantageous for large-scale problems. With these formulations, all the optimization constraints, i.e., limits on joint angles can be expressed explicitly in terms of the optimization variables. Therefore their gradient evaluations become simple. Note that digital human motion prediction and simulation is an active and vast area of research and this current chapter can by no means present all the relevant materials. Instead the focus of this research is to compare various formulations of optimization-based motion prediction and provide clues on how best to formulate the problem for practical applications. Some key features of the present work include: (1) three formulations based on different discretization methods are evaluated and compared for digital human gait simulations, (2) a recursive form of the equations of motion is discretized directly, and no EOMs are integrated. The present chapter also uses a more recent SQP algorithm and associated software. One numerical example is optimized and its solutions are compared. Advantages and disadvantages of the formulations are discussed.

9.2 Human Motion Modeling

A skeletal model is used in this study. Human limbs are modeled as a series of linkages. The movements are generated by muscle forces, which act on the skeletal bones through lever arms thereby generating torques on joints. Before setting up the optimization formulations, kinematics and dynamics analyses of the system need to be carried out. The DH method and recursive formulation are adopted for kinematics and dynamics analyses, respectively (Hollerbach 1980; Toogood 1989).

9.2.1 The DH method and recursive kinematics formulation

The DH method is an approach for relating the position of a point in one coordinate system to another, by using transformation matrices (Denavit and Hartenberg 1955). In order to obtain a systematic representation of a serial kinematics chain, $\mathbf{q} \in R^d$ is defined as the vector of d-generalized coordinates, the joint angles. The position vector of a point of interest in the Cartesian space can be written in terms of the joint variables as $\mathbf{X} = \mathbf{X}(\mathbf{q})$, where $\mathbf{X}(\mathbf{q})$ can be obtained from the multiplication of the 4×4 homogeneous transformation matrices ${}^{i-1}\mathbf{T}_i$ relating coordinate frames i and i-1, represented by four parameters θ_i, d_i, α_i, and a_i, as shown in Figure 9.1.

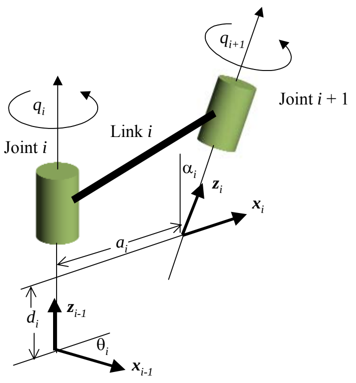


Figure 9.1 Joint coordinate systems

Let us define augmented 4×1 vectors ${}^0\mathbf{r}_d$ and $\mathbf{r}_d$ using the global Cartesian vector $\mathbf{X}(\mathbf{q})$ and the local Cartesian vector $\mathbf{X}_d$ as:

$$ {}^0\mathbf{r}_d = \begin{bmatrix} \mathbf{X}(\mathbf{q}) \\ 1 \end{bmatrix}; \; \mathbf{r}_d = \begin{bmatrix} \mathbf{X}_d \\ 1 \end{bmatrix} \tag{9.2.1} $$

where $\mathbf{X}_d$ is the position of the point with respect to the dth coordinate system. Using these vectors, ${}^0\mathbf{r}_d$ can be related to $\mathbf{r}_d$ (i.e., the global Cartesian vector $\mathbf{X}(\mathbf{q})$ can be expressed in terms of the local Cartesian vector $\mathbf{X}_d$) as:

$$ {}^0\mathbf{r}_d = {}^0\mathbf{T}_d(\mathbf{q})\mathbf{r}_d \tag{9.2.2} $$

where

$$ {}^{0}\mathbf{T}_d(\mathbf{q}) = \prod_{i=1}^{d} {}^{i-1}\mathbf{T}_i = {}^{0}\mathbf{T}_1(q_1)\,{}^{1}\mathbf{T}_2(q_2)\cdots{}^{d-1}\mathbf{T}_d(q_d) \tag{9.2.3} $$

According to the above analysis, we can define matrices $\mathbf{A}_j$, $\mathbf{B}_j$, and $\mathbf{C}_j$ as recursive position, velocity, and acceleration transformation matrices for the *j*th joint, respectively. Given the link transformation matrix ($\mathbf{T}_j$) and the kinematics state of each joint variable ($q_j, \dot{q}_j$ and $\ddot{q}_j$), for $j = 1, d$ (i.e., an d degree of freedom chain), we have:

$$ \mathbf{A}_j = \mathbf{T}_1\mathbf{T}_2\mathbf{T}_3\cdots\mathbf{T}_j = \mathbf{A}_{j-1}\mathbf{T}_j \tag{9.2.4} $$

$$ \mathbf{B}_j = \dot{\mathbf{A}}_j = \mathbf{B}_{j-1}\mathbf{T}_j + \mathbf{A}_{j-1}\frac{\partial \mathbf{T}_j}{\partial q_j}\dot{q}_j \tag{9.2.5} $$

$$ \mathbf{C}_j = \dot{\mathbf{B}}_j = \ddot{\mathbf{A}}_j = \mathbf{C}_{j-1}\mathbf{T}_j + 2\mathbf{B}_{j-1}\frac{\partial \mathbf{T}_j}{\partial q_j}\dot{q}_j + \mathbf{A}_{j-1}\frac{\partial^2 \mathbf{T}_j}{\partial q_j^2}\dot{q}_j^2 + \mathbf{A}_{j-1}\frac{\partial \mathbf{T}_j}{\partial q_j}\ddot{q}_j \tag{9.2.6} $$

where $\mathbf{A}_0 = [\mathbf{I}]$ and $\mathbf{B}_0 = \mathbf{C}_0 = [\mathbf{0}]$. After obtaining all the transformation matrices $\mathbf{A}_j$, $\mathbf{B}_j$, and $\mathbf{C}_j$, the global position, velocity, and acceleration of a point in the Cartesian coordinates can be calculated as (Xiang *et al.* 2006)

$$ {}^{0}\mathbf{r}_d = \mathbf{A}_d\mathbf{r}_d \tag{9.2.7} $$

$$ {}^{0}\dot{\mathbf{r}}_d = \mathbf{B}_d\mathbf{r}_d \tag{9.2.8} $$

$$ {}^{0}\ddot{\mathbf{r}}_d = \mathbf{C}_d\mathbf{r}_d \tag{9.2.9} $$

where $\mathbf{r}_d$ is the local coordinates of the point in the dth coordinate system.

9.2.2 Dynamics (Recursive Lagrangian equations)

The general form of dynamic equations of motion is derived from the energy principle. Based on the recursive kinematics analysis of the previous section, the backward recursion for the dynamic analysis is accomplished by defining transformation matrices $\mathbf{D}$ and $\mathbf{E}$ as follows (Xiang *et al.* 2006). Given the mass and inertia properties of each link, the joint actuation forces/torques, τ_i, are computed for $i = d$ to 1 using

$$\tau_i = tr\left[\frac{\partial \mathbf{A}_i}{\partial q_i}\mathbf{D}_i\right] - \mathbf{g}^T \frac{\partial \mathbf{A}_i}{\partial q_i}\mathbf{E}_i \tag{9.2.10}$$

$$\mathbf{D}_i = \mathbf{J}_i \mathbf{C}_i^T + \mathbf{T}_{i+1}\mathbf{D}_{i+1} \tag{9.2.11}$$

$$\mathbf{E}_i = m_i\, {}^i\mathbf{r}_i + \mathbf{T}_{i+1}\mathbf{E}_{i+1} \tag{9.2.12}$$

where $\mathbf{D}_{d+1} = \mathbf{E}_{d+1} = [\mathbf{0}]$; $\mathbf{J}_i$ = inertia matrix for link *i;* m_i = mass of link *i;* $\mathbf{g}$ = gravity vector; ${}^i\mathbf{r}_i$ = location of the center of mass of link *i* in link *i* frame. $tr[\cdots]$ and $(\cdots)^T$ denote the matrix trace and transpose operations, respectively. The first term in the torque expression denotes inertia and Coriolis torques; the second term denotes the torque due to gravity. Gradients information for all transformation matrices and torques with respect to state variables can be also evaluated in a recursive way as shown in Eqs. (9.2.4) to (9.2.12).

9.3 Human Motion Prediction as an Optimization Problem

It has been shown that task-based human motion prediction is in fact a numerical optimal control problem (OCP) (Wang *et al.* 2005). The basic optimal control problem is to determine unknown quantities such as joint angles and torques, to achieve certain goals (e.g., minimization of a performance measure function) while satisfying all the performance requirements or constraints. Although the formulations are applicable to various types of human motions, gait motion is considered in this study.

9.3.1 Objective function

A general objective function (performance index PI) for optimal control problem is defined as:

$$J = c_0\left(\mathbf{q}(T), T\right) + \int_0^T h_0\left(\boldsymbol{\tau}, \mathbf{q}, t\right) dt \tag{9.3.1}$$

where T is the total time interval considered. Note that the definition of the OCP problem contains a wide variety of control problems, such as minimum time, minimum control effort, trajectory tracking, and response constraints. The joint angles $\mathbf{q}$ and torques $\boldsymbol{\tau}$ are called *state variables* and *control variables* in the OCP problem. For digit human gait simulation, the objective functions represent human performance measures. Various performance measures of digital human have been developed in the literature (Pandy *et al.* 1992; Lo *et al.* 2002; Saidouni and Bessonnet 2003; Yang *et al.* 2004; Xiang *et al.* 2006).

9.3.2 Optimization constraints

The optimization constraints consist of the equations of motion in Eqs. (9.2.10) - (9.2.12), and other time-dependent requirements, as:

$$\mathbf{g}\left(\boldsymbol{\tau}, \mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, t\right) \le \mathbf{0} \tag{9.3.2}$$

This type of constraints is the so-called dynamic or *point-wise* constraint, which needs to be satisfied at each point of the entire time interval $t \in [0, T]$. The other type of constraints are not functions of t; therefore, they can be treated easily in the optimization process. These constraints include the initial and final motion constraints. Five treatments of the point-wise constraints in Eq. (9.3.2) have been discussed in the literature (Arora 1999). A reasonable treatment of time-dependent constraints is to impose them at the discrete time grid points. In dynamic gait simulations, the time-dependent inequality constraints in Eq. (9.3.2) may include the following basic constraints: (1) foot ground penetration, (2) foot strike position, (3) ZMP stability condition, (4) joint angle and torque limits.

1. Foot ground penetration

Foot ground penetration in each phase (controls height of foot points as shown in Figure 9.2). For foot points without contact (triangular points in Figure 9.2), it is required that $x \geq 0$; for foot points with contact condition (circular points in Figure 9.2), the condition is $x = 0$, where x is the vertical coordinate of any point on a foot. It is seen that the number of foot ground penetration constraints depends on the total number of points on the two feet.

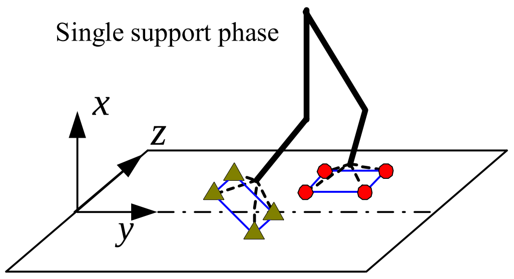


Figure 9.2 Foot ground penetration constraints

2. Foot strike position

Foot strike position can be used to determine walking direction and step length. As shown in Figure 9.3, the strike coordinates can be imposed as equality constraints.

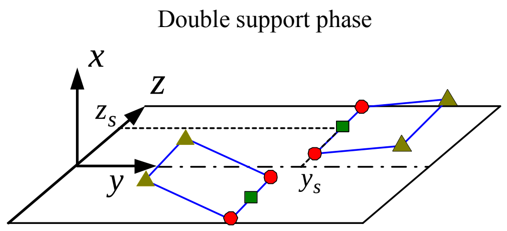


Figure 9.3 Foot strike position constraints

3. ZMP stability

The dynamic stability that refers to dynamic equilibrium of all the forces on a walking biped, is imposed by restricting the ZMP location within the boundary of foot support region (FSR). The ZMP concept dates back thirty-five years and has been re-examined by recent literature either for clarification or extension (Goswami 1999; Vukobratović and Borovac 2004). The ZMP coordinates and stability requirements are:

$$y_{ZMP} = \frac{\sum_{i=1}^{ndof} m_i(\ddot{x}_i + g)y_i - m_i\ddot{y}_i x_i + I\ddot{\theta}_{iz}}{\sum_{i=1}^{ndof} m_i(\ddot{x}_i + g)} \in FSR \tag{9.3.3}$$

$$z_{ZMP} = \frac{\sum_{i=1}^{ndof} m_i(\ddot{x}_i + g)z_i - m_i\ddot{z}_i x_i + I\ddot{\theta}_{iy}}{\sum_{i=1}^{ndof} m_i(\ddot{x}_i + g)} \in FSR \tag{9.3.4}$$

where x_i, y_i, and z_i are the coordinates of the center of mass for *i*th link; m_i is the mass, I_i is the global inertia, and $\ddot{\theta}_i$ is the global angular acceleration of the *i*th link, respectively.

4. Joint angle and torque limits

The constraints on the joint angle profiles and torques are

$$\mathbf{q}^L \le \mathbf{q} \le \mathbf{q}^U \,;\; \boldsymbol{\tau}^L \le \boldsymbol{\tau} \le \boldsymbol{\tau}^U \qquad (9.3.5)$$

where $\mathbf{q}^L, \mathbf{q}^U$ and $\boldsymbol{\tau}^L, \boldsymbol{\tau}^U$ are the lower and upper bounds for the joint displacements and torques, respectively. Note that the joint torque limits $\boldsymbol{\tau}^L$ and $\boldsymbol{\tau}^U$ are sometimes difficult to obtain; therefore, a viable way is to treat them as a part of the objective function. This constitutes a so-called minimum control effort problem, which has no particular difficulty for solution.

9.4 Solution Techniques of Optimal Control Problems

The optimal control problems find the best solution that achieves user-specified objective through forward or inverse dynamics combined with optimization. Several different ways based on numerical optimization to solve the OCP problem are available, namely conventional state variable elimination (forward dynamics) method, direct collocation, and differential inclusion (inverse dynamics). Direct collocation and differential inclusion do not need the integration of the equations of motion; they are regarded as alternative formulations for optimal control in this chapter.

9.4.1 State variable elimination

This is the conventional method to solve numerical optimal control problems. Forward dynamics starts with initial conditions and known forces and solves for unknown joint displacements by numerical integration (Neptune and Hull 1998; Roussel *et al.* 2001; Anderson and Pandy 2001). The process of integrating forces over time intervals to obtain walking motions can itself be computationally intensive. Design sensitivity analysis is also needed in this formulation, since the joint angle, velocities and accelerations are implicit functions of the optimization variables, the joint torques (Pandy *et al.* 1999; Arora 1992). One way to deal with the computational intensity of forward dynamics formulations has been to use massively parallel algorithms and processing scheme (Pandy *et al.* 1999).

9.4.2 Direct collocation

This is the simultaneous formulation where both the joint torques and the joint profiles are treated as optimization variables. Therefore the equations of motion in Eqs. (9.2.10) to (9.2.12) are treated as equality constraints in the formulation. The direct collocation approach has been used in other engineering fields (Enright and Conway 1992; von Stryk and Bulirsch 1992; Betts 1998; Hull 2003; Schulz 2004), and robotic or human motion planning (Kaplan and Heegaard 2001). Although there are a large number of variables in this formulation, the equations of motion are not required to be satisfied at each iteration of the optimization process. They only need to be satisfied at the final optimum point of the problem. There are two main advantages of this formulation for dynamic systems: (i) the equations of motion for the system need not be integrated explicitly, (ii) design sensitivity analysis of the systems is not needed since all the problem functions are explicit in terms of the variables. With these formulations, the optimization problem becomes large; i.e., the numbers of variables and constraints are large. However, the problem functions are quite sparse; i.e., each function depends on only a few variables. These sparse properties of the functions can be exploited in the optimization process.

9.4.3 Differential inclusion

The inverse dynamics method, on the other hand, calculates unknown forces from joint displacement histories. The joint displacement histories associated with locomotion are determined using optimization methods (Chevallereau and Aoustin 2001; Bessonnet *et al.* 2002). This is a unique class of alternative formulation, which does not exist in optimization of structures subjected to static loads. Two important issues in such inverse dynamics frameworks are the human performance criteria and methods for approximating the joint trajectories. The work of Lo *et al.* (2002) provides a thorough description of an inverse dynamics framework for predicting human motions, although it deals with human motions other than the locomotion. Note that in this formulation, the equations of motion in Eqs. (9.2.10) to (9.2.12) are not integrated nor treated as equality constraints; they are automatically satisfied in the optimization process.

9.5 Discretization Techniques of Equations of Motion

Different discretization techniques for the equations of motion in Eqs. (9.2.10) to (9.2.12) are available. The general idea is to transfer the EOMs to an algebraic system of equations, the so-called defect equations, which need to be set to zero to enforce the EOMs. In the next section, some of these formulations are presented, and they are based on finite difference method, and Hermite and B-spline interpolations.

9.5.1 Central difference (CD)

This is the perhaps the easiest way to discretize the system of dynamic equations. Some common methods include forward, backward difference, and central difference. In the central difference method, the joint velocity and acceleration vectors $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are written explicitly with respect to the joint angle vector $\mathbf{q}$, as follows::

$$\dot{\mathbf{q}}_i = \dot{\mathbf{q}}_i(t_i) = \frac{\mathbf{q}_{i+1} - \mathbf{q}_{i-1}}{2\Delta t}, \quad i = 0,N \qquad (9.5.1)$$

$$\ddot{\mathbf{q}}_i = \ddot{\mathbf{q}}_i(t_i) = \frac{\mathbf{q}_{i+1} - 2\mathbf{q}_i + \mathbf{q}_{i-1}}{\Delta t^2}, \quad i = 0,N \tag{9.5.2}$$

where Δt is the time interval ($\Delta t = T / N$). If some of the state variables, such as the joint angles $\mathbf{q}$, are treated as variables in the optimization formulation, the implicit problem becomes explicit. Therefore special design sensitivity analysis procedures are not needed, when the direct collocation or inverse dynamic methods are employed. The finite difference approximations are used to replace the joint velocities and accelerations in terms of the angles. Note that Eqs. (9.5.1) and (9.5.2) are the additional equality constraints between the variables.

Note that the quality of the final solution depends on the approximation made between the state variables. In this research, the central finite difference method is used to approximate the relationship between joint angles, velocities and accelerations. In order to have good results, the number of grid points usually needs to be large. However, the finite difference approximation is simple and easy to implement. This is the major advantage of the formulation. It is clear that other alternative formulations are possible. If the joint velocities $\dot{\mathbf{q}}$ or accelerations $\ddot{\mathbf{q}}$ are also treated as variables, it will give choice of expressing some constraints, e.g., the equations of motion, in terms of velocities or accelerations, to simplify their expressions. This may lead to simpler gradient evaluation and computer implementation (Wang and Arora 2005b).

9.5.2 Piecewise Hermite interpolation (Hermite)

The basic discretization scheme is as follows: the state variables $\mathbf{q}$ are chosen as continuous differentiable functions and piecewise defined as cubic polynomials between $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$, with the EOMs (9.2.10) to (9.2.12) satisfied at t_i and t_{i+1}. For $t \in [t_i, t_{i+1}]$, the approximation of state variables $\mathbf{q}$ is:

$$\mathbf{q}_a(t) = \sum_{k=0}^{3} \mathbf{c}_k^i \left(\frac{t - t_i}{\Delta t} \right)^k \tag{9.5.3}$$

where

$$\begin{aligned} \mathbf{c}_0^i &= \mathbf{q}_i \\ \mathbf{c}_1^i &= \Delta t \dot{\mathbf{q}}_i \\ \mathbf{c}_2^i &= -3\mathbf{q}_i - 2\Delta t \dot{\mathbf{q}}_i + 3\mathbf{q}_{i+1} - \Delta t \dot{\mathbf{q}}_{i+1} \\ \mathbf{c}_3^i &= 2\mathbf{q}_i + \Delta t \dot{\mathbf{q}}_i - 2\mathbf{q}_{i+1} + \Delta t \dot{\mathbf{q}}_{i+1} \end{aligned} \tag{9.5.4}$$

where $\Delta t = t_{i+1} - t_i$. The approximation function (9.5.3) of the state variables must satisfy the EOMs (9.2.10) - (9.2.12) at the grid point t_i ($i = 0, N$). The time dependent constraints are satisfied at the grid points. The optimization problem is to determine $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\boldsymbol{\tau}$ to minimize the cost function of Eq. (9.3.1), subject to the discretized inequality constraints in Eq. (9.3.2) at the time grid point. Note that at the grid point $t = t_i$,

$$\ddot{\mathbf{q}}_i = \ddot{\mathbf{q}}_a(t_i) = \frac{2}{\Delta t^2}\left(-3\mathbf{q}_i + 3\mathbf{q}_{i+1} - 2\Delta t \dot{\mathbf{q}}_i - \Delta t \dot{\mathbf{q}}_{i+1}\right) \tag{9.5.5}$$

9.5.3 Cubic B-spline interpolation (B-spline)

Since human motion trajectories are usually very smooth, cubic B-spline interpolation can be used. For the optimization problem, the entire time domain is discretized by B-spline curves, which are defined by a set of control points $\mathbf{P}$ and time grid points (knots) $\mathbf{t}$. B-spline is a numerical interpolation method that has many important properties, such as continuity, differentiability, and local control (Piegl and Tiller 1997). These properties, especially differentiability and local control, make B-splines competent to represent joint angle trajectories, which require smoothness and flexibility. There are a number of ways to define the B-spline basis functions, and it is preferred to have an explicit polynomial form rather than in a recursive form. Let $T = \{ t_0, t_1, .., t_{nk} \}$ be a non-decreasing sequence of real numbers, i.e., $t_i \le t_{i+1},\ i = 0, nk\text{-}1$. The t_i are called *knots*, and they are non-decreasingly spaced. A cubic B-spline is defined as

$$q(t) = \sum_{j=0}^{nc} N_{j,4}(t) P_j; \qquad 0 \le t \le T \tag{9.5.6}$$

where the $\{P_j\}$, $j = 0, nc$ are the $(nc+1)$ control points, and the $\{N_{j,4}(t)\}$ are the cubic B-spline basis functions defined on the knot vector ($(nk+1)$ knots). In reality each segment of the curve is defined by four control points. The *k*th derivatives of a cubic B-spine curve can be easily obtained from Eq. (9.5.6), since only the basis functions are functions of time. In this formulation, the control point vector $\mathbf{P}$ for each DOF is chosen as the optimization variables. This formulation is to minimize the objective function in Eq. (9.3.1), subject to the inequality constraints in Eq. (9.3.2), as

$$\mathbf{g}(\boldsymbol{\tau}, \mathbf{P}, t) \le \mathbf{0} \tag{9.5.7}$$

9.6 Discussion of Formulations

Advantages and disadvantages of different formulations are listed in Table 9.1. Since the objective and constraint functions are all explicit in terms of the optimization variables in the direct collocation or differential inclusion formulations, the gradients of functions can be obtained easier than the state variable elimination method. Starting form a system of differential equations, approximations in the time domain for the state variables are set up and collocation can be enforced on certain time points in the direct collocation formulations. The equations of motion in Eqs. (9.2.10) - (9.2.12) do not need to be satisfied exactly at each iteration of the optimization process. They need to be satisfied only at the final solution point. This has advantage if instabilities occur or no solution exists for DEs for certain points in the design space. Also, unnecessary simulations of the system are avoided at intermediate designs, where it might be difficult to obtain a solution. The differential inclusion formulation does not need the integration of the dynamic equations, either. In these formulations, the system of DEs is directly discretized and imbedded into the optimization formulation. However, the error in the solution of DEs in state variable elimination formulations can be easily controlled, which is not an easy task for the other two formulations. Differential inclusion formulation includes fewer variables and needs less storage than direct collocation; therefore it has special advantage. Note that differential inclusion formulation is only available in optimal control problems; there is no similar formulation in optimal design of systems subjected to static loads.

In terms of the discretization techniques, several different ways can be used: direct discretization by finite differences, piece-wise polynomials and splines of various orders. Table 9.2 lists the advantages and disadvantages of different discretization techniques. For the piece-wise polynomial and spline interpolations, the number of variables is usually not very large; therefore, the resulting NLP is not too large. Sparsity can be utilized in the formulations, but not necessary. These formulations provide good smoothness for the final solution; therefore, they are well suitable for digital human motion simulation. However, the drawback of these formulations is that the implementation is not straightforward. They are sometimes too restrictive; therefore no solution or no good solution may be obtained for the optimal control problem. The finite difference method usually requires large number of grid points N; therefore large-scale NLP algorithms with sparse matrix capabilities are required to solve the problems efficiently.

Table 9.1 Advantages and disadvantages of different formulations

Formulations	Variables	Advantages	Disadvantages
State Variable Elimination	$\boldsymbol{\tau}$	1. Small optimization problems. 2. Equations of motion are satisfied at each iteration; intermediate solutions may be usable. 3. Error in the solution of DEs can be controlled.	1. Equations of motion must be integrated at each iteration, which is expensive. 2. A good DEs integrator is needed. 3. Objective or constraints involving $\mathbf{q}$ are implicit functions of the variables; their evaluation requires solution of the equations of motion. 4. Design sensitivity analysis must be performed, and its implementation is tedious. 5. Dense constraint Jacobian and Hessian matrices; difficult to treat large-scale problems.
Direct Collocation	$\mathbf{q}, \boldsymbol{\tau}$	1. Formulations are explicit in terms of variables. 2. Equations of motion are not integrated at each iteration. 3. Constraint Jacobians and Hessian are sparse. 4. Design sensitivity analysis is not needed. 5. Constraints on $\mathbf{q}$ and $\boldsymbol{\tau}$ can be treated efficiently.	1. Numbers of variables and constraints are large. 2. Intermediate solutions may not be usable. 3. Optimization algorithms for large-scale problems must be used. 4. For efficiency, advantage of sparsity of the constraint Jacobians and Hessians must be utilized. 5. Optimization variables sometimes need to be normalized.
Differential Inclusion	$\mathbf{q}$	1. Smaller number of optimization variables. 2. Formulations are explicit in terms of variables. 3. Equations of motion are not integrated, but are satisfied at each iteration. 4. Intermediate solutions may be usable. 5. Design sensitivity analysis is not needed. 6. Constraints on $\mathbf{q}$ can be treated efficiently.	1. Objective or constraints involving $\boldsymbol{\tau}$ need evaluation of inverse dynamics. 2. Implementation sometimes is not straightforward.

Table 9.3 shows the approximated numbers of non-zero elements in the gradient vector and Jacobian of constraint functions for all the formulations. The following symbols are used in the table: d = number of degree of freedoms (DOFs) in the human model; N = number of time intervals (number of time grid points = $N+1$); n = number

of control points in a cubic B-spline; N_g = number of foot ground penetration constraints; N_s = number of foot strike position constraints; N_Z = number of ZMP stability constraints. This approximation in fact provides upper bounds for the numbers of non-zero elements in the vectors and matrices. Only the storage of the input information for the algorithm, such as the Jacobian matrix, is discussed and compared.

Table 9.2 Advantages and disadvantages of different discretization techniques

Formulation	Advantages	Disadvantages
Finite Difference	• Very sparse constraint Jacobians and Hessian. • Implementation is very straightforward. • State or control variable constraints become simple bounds in most cases.	• Larger number of variables and constraints. • Optimization algorithms for large-scale sparse problems must be used.
Piece-wise Polynomial Interpolation	• Good smoothness. • Smaller NLP problems. • State or control variable constraints become simple bounds or linear in most cases	• The required curve profile may be too restrictive. • Implementation sometimes is not straightforward.

Table 9.3 Numbers of non-zeros in gradient vector and Jacobian for formulations

Item		**Formulation**		
		CD	**Hermite**	**B-spline**
Gradient Vector	**Objective Function**	$d(N+3)$	$2d(N+2)$	dn
Jacobian of Constraints	**Foot Ground Penetration Constraints**	$3dN_p(N+1)$ $[dN_p(N+3)(N+1)]$*	$4dN_p(N+1)$ $[2dN_p(N+2)(N+1)]$	$4dN_p(N+1)$ $[dnN_p(N+1)]$
	Foot Strike Position Constraints	$3dN_s(N+1)$ $[dN_s(N+3)(N+1)]$	$4dN_s(N+1)$ $[2dN_s(N+2)(N+1)]$	$4dN_s(N+1)$ $[dnN_s(N+1)]$
	ZMP Stability Constraints	$3dN_Z(N+1)$ $[dN_Z(N+3)(N+1)]$	$4dN_Z(N+1)$ $[2dN_Z(N+2)(N+1)]$	$4dN_Z(N+1)$ $[dnN_Z(N+1)]$
	Joint Angle Constraints	-	-	$4d(N+1)$ $[d^2n(N+1)]$
Total of 1st Order Derivatives (Gradient Vector & Jacobian)		$3d(N_p+N_s+N_Z)(N+1)+$ $d(N+3)$ $[d(N+3)\cdot$ $((N_p+N_s+N_Z)(N+1)+1)]$	$4d(N_p+N_s+N_Z)(N+1)+$ $2d(N+2)$ $[2d(N+2)\cdot$ $((N_p+N_s+N_Z)(N+1)+1)]$	$4d(N_p+N_s+N_Z+1)(N+1)+$ dn $[dn\cdot$ $((N_p+N_s+N_Z+d)(N+1)+1)]$

*The expressions in the brackets give the total number of elements when sparsity is not considered.

9.7 Numerical Examples

All the formulations developed in Section 9.5 are applied to a gait simulation example for evaluation. All the formulations are solved using the sparse SQP algorithm in SNOPT (Gill *et al.* 2002). A PC with 3.2 GHz processor and 1.0 GB RAM is used for running the programs and recording the relative CPU times. Each solution case of the example problem was run several times and the shortest time was recorded. Results of the examples are listed and compared. Advantages and disadvantages of the formulations are discussed.

9.7.1 Example 1 – 2-DOF manipulator arm

A two-link rigid manipulator arm is considered and solved using the formulations developed in the previous section. The reason to use this example is to present basic ideas and to validate the numerical results, since the solutions of this well-studied example are readily available. The two links are considered rigid and the joint coordinates are selected as independent generalized coordinates. The manipulator under study consists of two links whose lengths are L_1 and L_2 and moments of inertia I_1 and I_2. q_1 and q_2 are the relative joint angels that are controlled by the joint actuator torques τ_1 and τ_2. The manipulator is assumed to lie in the horizontal plane, and the gravity effects are neglected in writing the equations of motion. The equations of motion for the two-link manipulator are given by Dissanayake *et al.* (1991), Goh *et al.* (1993), Kota (1996) and Furukawa (2002). The parameters of the manipulator are given as $L_1 = L_2 = 0.4$ m, $m_1 = m_2 = 0.5$ kg, and $I_1 = I_2 = 0.1$ kg-m^2. The upper and lower bounds for control torques are ± 10 Nm. Two situations are considered in this example.

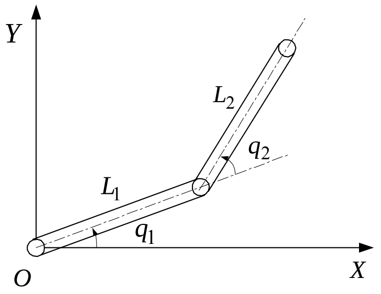


Figure 9.4 Example 1: two-link arm model

Table 9.4 Example 1-Case 1

Formulations		No. of variables	No. of Constraints	No. of Non-zeros in Constraint Jacobian	No. of Iteration	T (s)	CPU (s)
State Variable Elimination		$2N+3$	4	$8N+12$	21	0.393	6.72
Direct Collocation	CD	$4N+9$	$2N+6$	$16N+24$ $[8N^2+42N+36]$*	10	0.392	0.05
	Hermite	$6N+11$	$4N+2$	$32N+20$ $[24N^2+56N+22]$	37	0.393	0.57
	B-spline	$2n+2N+3$	$2N+10$	$20N+44$ $[4N^2+4Nn+26N+20n+30]$	22	0.394	0.19
Differential Inclusion	CD	$2N+7$	$2N+6$	$14N+22$ $[4N^2+16N+42]$	15	0.392	0.09
	Hermite	$4N+9$	$4N+2$	$30N+18$ $[16N^2+44N+18]$	316	0.393	3.81
	B-spline	$2n+1$	$2N+10$	$18N+42$ $[4Nn+2N+20n+10]$	16	0.394	0.16

*The expressions in the brackets give the total number of elements when sparsity is not considered.

Case 1: The OCP is formulated such that the end point of the manipulator is desired to move form one location to another in minimum time. Note that the joint actuator torques τ_1 and τ_2 must be determined such that the end point moves from a position (q_{10}, q_{20}) in the horizontal plane at time $t=0$ to a point (q_{1T}, q_{2T}) at time $t=T$. T is the final time that needs to be minimized. The OCP is subjected to the following initial and final conditions: $q_{10} = 0.0, q_{20} = -2.0, \dot{q}_{10} = 0.0, \dot{q}_{20} = 0.0$; $q_{1T} = 1.0, q_{2T} = -1.0, \dot{q}_{1T} = 0.0, \dot{q}_{2T} = 0.0$. where q_{10}, q_{20} , q_{1T} and q_{2T} are the initial and final values of the joint angles that specify the orientation of the links. The terminal velocities are taken as zero. Central difference, Hermite interpolation and B-spline are used in this example. The number of variables, constraints and the final minimum time are listed in Table 9.4, where N is the number of time intervals considered (N = 20 is used for all formulations). $\mathbf{P}$ contains the unknown control points, and n is the number of control points, respectively. The minimum time is similar to that reported in the literature (0.3945 s by Furukawa (2002)).

Table 9.5 Example 1-Case 2

Formulations		No. of variables	No. of Constraints	No. of Non-zeros in Constraint Jacobian	No. of Iteration	T (s)	CPU (s)
State Variable Elimination		$2N+3$	$N+4$	$2N^2+11N+12$	78	0.405	22.56
Direct Collocation	CD	$4N+9$	$2N+6$	$16N+24$ $[8N^2+42N+36]$*	15	0.404	0.08
	Hermite	$6N+11$	$4N+2$	$32N+20$ $[24N^2+56N+22]$	50	0.405	0.78
	B-spline	$2n+2N+3$	$3N+11$	$24N+48$ $[6N^2+6Nn+31N+22n+33]$	17	0.406	0.16
Differential Inclusion	CD	$2N+7$	$2N+6$	$14N+22$ $[4N^2+16N+42]$	26	0.404	0.14
	Hermite	$4N+9$	$4N+2$	$30N+18$ $[16N^2+44N+18]$	150	0.405	2.22
	B-spline	$2n+1$	$3N+11$	$22N+46$ $[6Nn+3N+22n+11]$	18	0.406	0.23

*The expressions in the brackets give the total number of elements when sparsity is not considered.

Case 2: Additional point-wise constraints on the state variables are considered, as $g = |q_2| - 2.0 \leq 0.0$. The inclusion of point-wise state constraints brings more nonlinear constraints in the state elimination formulation, and more linear constraints in the B-spline based formulations. However, in others formulations these constraints are simple bounds, as shown in Table 9.5. The minimum time is better than that reported in the literature (0.424 s by Kota (1996)).

It is seen that various formulations work for the example problem and similar optimal solutions are obtained. It is seen from Tables 9.4 and 9.5 that it generally takes more time for the state variable elimination formulation to find the optimal point, since repeated integration of the equations of motion is required. Formulations based on direct collocation and differential inclusion generally require less CPU effort, since the integration of the equations of motion is not needed. Compared to direct collocation method, differential inclusion formulation has smaller numbers of variables and non-zero elements in constraint Jacobian, which is advantageous for large-scale simulation problems. The inclusion of point-wise state constraints, such as the joint displacement limits brings more constraints in the state elimination formulation, and more linear constraints in the B-spline based formulations. However, it shows no special difficulty for others formulations based on direct collocation and differential inclusion. For the formulations based on direct collocation and differential inclusion methods, it is seen that the problems are quite sparse, with large numbers of zeros in the constraint Jacobian. The consideration of matrix sparsity can in general reduce the data storage by an order of magnitude. The sparse NLP code works quite well for these formulations, and turns out to be the key to solve large-scale problems. In the next section, various differential inclusion (inverse dynamics) formulations will be applied to a large-scale human gait model.

9.7.2 Example 2 – 18-DOF human lower-body gait model

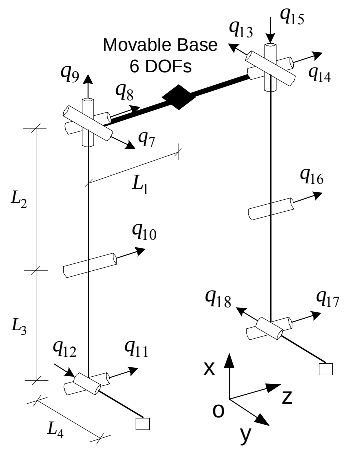


Figure 9.5 Example 2: 18-DOF human lower-body gait model

Table 9.6 Anthropometric Data of example 2

Item	Pelvis	Thigh	Shank	Foot
Length (m)	0.0085	0.383	0.395	0.180
Mass (kg)	33.55	9.70	4.50	2.80
Inertia (kgm^2)	$\begin{bmatrix} 0.3550 & 0 & 0 \\ 0 & 0.2890 & 0 \\ 0 & 0 & 0.4120 \end{bmatrix}$	$\begin{bmatrix} 0.4008 & 0 & 0 \\ 0 & 0.1381 & 0 \\ 0 & 0 & 0.5116 \end{bmatrix}$	$\begin{bmatrix} 0.0046 & 0 & 0 \\ 0 & 0.2269 & 0 \\ 0 & 0 & 0.2269 \end{bmatrix}$	$\begin{bmatrix} 0.0128 & 0 & 0 \\ 0 & 0.0044 & 0 \\ 0 & 0 & 0.0164 \end{bmatrix}$

An 18-DOF three-dimensional digital human lower-body model is considered as shown in Figure 9.5. In this model, the hip has 6 global DOFs, including 3 translations and 3 rotations. The pelvis has 3 rotational DOFs, and the knee has 1 rotational DOF. The ankle is represented by 2 orthogonal rotational joints. The two legs are exactly symmetric. The physical data for thigh, shank, and foot are listed in Table 9.6 (Xiang *et al.* 2006). The number of foot ground penetration constraints in this example is 8, which is the total number of points on the two feet. One foot strike position constraint and four ZMP stability constraints are used in this study. Note also that since the ground reaction forces are not considered in the current formulation, the torques are only due to gravity, inertia, and Coriolis effects.

The total motion time is considered as 2.442 seconds. No special techniques are used to find an initial point for the formulations. The starting values for the optimization variables are taken as zero. Table 9.7 lists the sizes of the problem for different formulations. Table 9.8 gives the numbers of iterations and CPU (s) for different formulations.

9.7.3 Discussion of results

1. Number of time steps

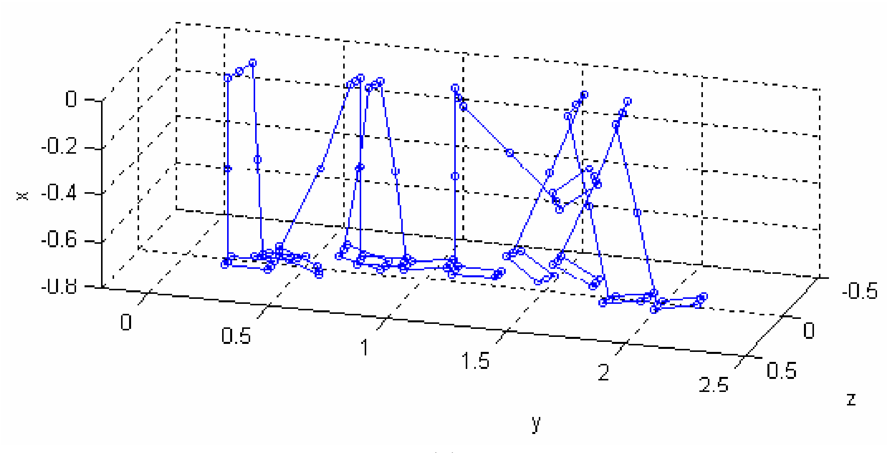
It is obvious that the number of time steps used in the numerical solution process can affect the performance of the formulations. In general if the step size is large, the size of the optimization problems is small. The numbers of iterations and CPU times to find an optimal solution are small, and vise versa. If the step size is too small, the sizes of the alternative formulations become very large which requires additional calculations and computer storage. To evaluate the performance of various formulations, a few different grid sizes are tried for the example and various data and results are summarized in Tables 9.7 and 9.8. The numbers of variables, constraints, elements in Jacobian for different formulations are listed Table 9.7. It is seen that for the same number of time steps, there are more variables in the formulation based on Hermite interpolation, and more constraints in the B-spline formulation. All the three formulations are indeed very sparse, with small numbers of non-zero elements in the constraint Jacobian. When the number of time step is larger, the density of the Jacobian matrices (percentage of non-zero elements to all) becomes smaller, and vise versa. Note that the formulation based on B-spline interpolation is the sparsest one, although the difference among all the formulations is small. Table 9.8 shows that as the number of time steps is increased the computational effort with all the formulations also increases. It is also observed that the CD and B-spline formulations require less computational efforts compared to the formulation based on Hermite interpolation. For smaller numbers of time steps, the CD formulation is very efficient. When the number of time steps becomes larger, CD and B-spline formulations require similar computational efforts.

Table 9.7 Sizes of example 2

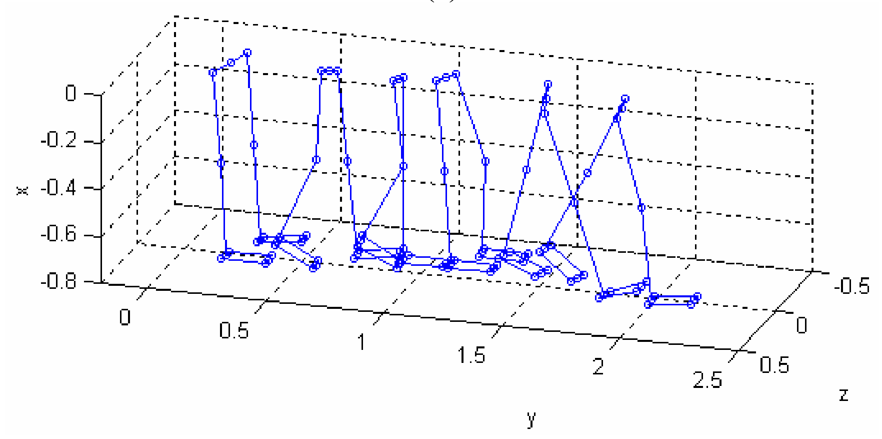
Formulation		**No. of Variables**	**No. of Constraints**	**No. of Non-zero Elements in Jacobian**	**Total No. of Elements in Jacobian**	**Density**
CD	$N = 12$	270	170	9396	45900	20.5 %
	$N = 36$	702	482	26676	338364	7.9 %
	$N = 72$	1350	950	52596	1282500	4.1 %
Hermite	$N = 12$	504	170	12672	85680	14.8 %
	$N = 36$	1368	482	36000	659376	5.5 %
	$N = 72$	2664	950	70992	2530800	2.8 %
B-spline	$N = 12$	270	404	13374	109080	12.3 %
	$N = 36$	702	1148	37998	805896	4.7 %
	$N = 72$	1350	2264	74934	3056400	2.5 %

Table 9.8 Numbers of iterations and CPU (s) for different formulations

No. of Time Intervals (*N*)	**Numbers of Iterations**			**CPU (s)**		
	CD	**Hermite**	**B-spline**	**CD**	**Hermite**	**B-spline**
12	30	31	43	11	20	28
36	29	32	37	65	328	246
72	96	49	51	1398	2047	1503



(a)



(b)

Figure 9.6 Gait motions of a 18-DOF human lower-body model (a) kinematically feasible gait where torques are not considered; (b) torques are considered

2. Gait simulations

Figures 9.6 show the lower-body gait simulations. For simplicity of illustration, the axis in the walking direction (*y*) is extended. It is seen that all the constraints presented in Section 9.3, including the foot ground penetration and prescribed strike locations are all satisfied. Figure (9.6a) shows a kinematically feasible walking motion, while (9.6b) illustrates an optimal gait motion with the absolute torque values are minimized. Note that since the joint torques are not considered in (9.6a), some motions such as the high kick of the right leg in Fig. (9.6a) are not realistic. Figure (9.6b) shows a more realistic gait motion.

3. Advantages and disadvantages of formulations

The main advantages of the differential inclusion inverse-dynamics based formulations for dynamic systems are summarized as follows: (i) the equations of motion for the system need not be integrated explicitly; therefore, unnecessary simulations of the system are avoided at intermediate designs; (ii) design sensitivity analysis of the systems is not needed since all the problem functions are explicit in terms of the variables. The major disadvantage of the formulation based on B-spline interpolation is that some constraints are linear instead of simple bounds, such as the joint angle constraints.

4. Other formulations

It is clear that other simultaneous formulations are possible. These are based on different discretization techniques of the first order or second order EOMs (Betts 1998), and piecewise higher degree polynomial approximations of state variables (Herman and Conway 1995). However, as discussed in Chapter 8, these multi-step methods or higher degree of polynomials may result in significant complexity of numerical implementation for the formulations, which is not desired. The application of these methods for digital human motion prediction needs further evaluation.

5. Future research

Optimization-based gait prediction reveals great insights into real human gait. Intuitively, simulation of a natural gait will involve multiple objectives. This is an important topic for future consideration in simulation of human gait. A full-size three-dimensional gait model including human upper body and arm movements will be very useful for understanding of upper body motion. Human running and jumping motions are of special interest to certain types of applications, such as sports. Some further research in human motion prediction and simulation also includes comparison with other models now being used, and the consideration of collision avoidance in the formulations. A general framework for collision avoidance of human motions has been recently presented (Yang *et al*. 2006).

9.8 Summary

The task-based motion prediction of digital humans was shown to be an optimal control problem and therefore could be solved numerically by powerful sparse NLP techniques. In order to provide clues on how best to formulate the problem for practical applications, various solution techniques for optimization-based optimal control were presented and compared. Different discretization techniques were discussed and their strength and weakness compared. Based on this work, the following observations are made:

1. The direct collocation and differential inclusion methods did not require integration of the equations of motion. All functions of the formulations became explicit in terms of the optimization variables.
2. Differential inclusion formulation is unique in the sense that it is only available in optimal control problems. Compared to direct collocation method, differential inclusion formulation has special advantage and great potential for large-scale digital human motion prediction, because the size of the optimization problem is smaller.
3. Formulations based on central difference and B-spline required less

computational efforts compared to the formulation based on Hermite interpolation for the same number of time steps.

4. When the number of time steps becomes larger, CD and B-spline formulations require similar computational efforts. The differential inclusion formulation based on B-spline is currently being implemented in the virtual human environment SANTOS^{TM} (Yang *et al*. 2006).

More work is still needed to fully develop nonlinear optimization-based control techniques, such as differential inclusion for realistic human motion prediction, with the validation of motion tracking data. More justification should be provided for the selection of these methods for human motion simulation. The current control torque does not include ground reaction forces, so the torque in the stance leg is not real; the ground reaction forces need to be included in future work.

CHAPTER 10
CONCLUSIONS AND FUTURE RESEARCH

10.1 Discussion and Conclusions

Alternative formulations for optimization of structural and mechanical systems subjected to static and dynamic loads based on the concept of simultaneous analysis and design (SAND) were studied in this research. As sample application areas, optimal design of trusses, frames, and dynamic mechanical system as well as digital human motion simulation problems were considered. The major focuses and contributions of this research include: (1) review of the formulations presented in diverse fields with the objective of possible cross fertilization of ideas that can lead to better formulations and solution procedures for optimization of complex systems; (2) comparison and evaluation of various SAND formulations for structural optimization; (3) study of the sparse features and implementation with various alternative formulations for structural and mechanical system optimization; (4) integration of existing simulation software with alternative formulations; and (5) extension of the SAND approach originally developed for optimization of structures subjected to static loads to broader applications, such as transient dynamic response optimization, and simulation and control problems.

Some alternative SAND formulations for optimal design of trusses and frames were presented, analyzed and evaluated. The formulations were implemented using a sparse sequential quadratic programming (SQP) algorithm and a commercial structural analysis program. For the selected example problems, the alternative formulations worked quite well and converged to better optimal solutions than the known solutions in the literature. Advantages and disadvantages of various formulations were discussed. Implementation issues of the formulations with the commercial structural analysis program were described. A key conclusion is that the alternative formulations were easier to implement with the commercial structural analysis programs and had potential for practical applications. Other observations based on the present work areas follows:

1. Alternative formulations that do not require assembly of the structural stiffness matrix (i.e., Jacobian matrix of the equilibrium equations in terms of displacements) are more attractive than others.
2. Implementations with the SAND formulations with the existing analysis codes is simpler compared to the conventional formulation in the sense that no system of equations needs to be solved for gradient evaluations. This is highly advantageous for complex applications where the simulation code may be using iterative or approximate procedures to solve the governing equations.
3. Normalization (scaling) of the variables is needed in the alternative formulations. More effective automatic scaling procedures need to be developed to further improve efficiency of the formulations.
4. Sparsity of the problem functions must be utilized for efficiency and effectiveness of the alternative formulations.

Simultaneous formulations for optimization of transient dynamic mechanical systems were also proposed and evaluated. Similar to the SAND approach used for optimization of structures subjected to static loads and the direct collocation/transcription method for optimal control, different state variables or their parametric approximations were treated as optimization variables in the formulations, i.e., generalized displacements, velocities and accelerations. Therefore the discretized state equations (DEs), either in the first or second order forms could be treated as equality constraints in the optimization process. By introducing more variables into the formulations, the forms of the constraints and their derivatives were changed. The formulations were implemented with a sparse NLP code for evaluation. Different time steps were tested. The simultaneous formulations had more variables and constraints, although the constraints had simpler form compared to the conventional formulation. Therefore an optimization algorithm for large numbers of variables and constraints was used to solve the problem. The solutions for sample problems were obtained and compared. Based on this work, the following conclusions are drawn:

1. Formulations based on Hermite-Simpson discretization of first order DEs gave better solutions with the smaller number of time grid points.
2. Finite difference based methods were easier to implement than those based on first order DEs; However, they usually required a larger number of time grid points for better solutions.
3. In terms of CPU times, most alternative SAND formulations outperform the conventional formulation for a smaller number of time grid points.
4. When the problem size is very large, the sparse QP subproblem solver became slower to converge, resulting in much more computational effort than the conventional formulation.
5. More efficient solution methods of sparse QP subproblems and parallel algorithms for the alternative formulations need to be studied, developed and combined for broader practical applications of these formulations.

10.2 Some Future Research Topics

Further research work and developments worthy of investigation are described as follows:

1. Formulation of framed structure design for practical applications.
2. Application to problems with frequency constraints.
3. Application to nonlinear problems.
4. Application to topological and shape design problems.
5. Application to other types of structures, such as plates and shells.
6. Testing of other efficient optimization algorithms, such as augmented Lagrangian approaches.
7. Extension to other fields, including heat transfer, fluid dynamics, magnetics, electrical fields, etc.
8. Application to multidisciplinary design optimization (MDO).

SAND represents a fundamental shift in the way analysis and design problems are currently treated. Further research is suggested to fully study and utilize sparse features of the alternative formulations for large and more complex problems. The exploitation of the sparsity, decomposition to reduce sizes, efficient solution of sparse QP subproblems, and parallel algorithms for the alternative SAND formulations need to be further studied, developed and combined for much broader practical applications of these formulations.

REFERENCES

Achtziger, W. (1999). “Local stability of trusses in the contex of topology optimization. Part I: exact modelling.” *Structural and Multidisciplinary Optimization*, 17(4), 235–246.

Adeli, H., and Cheng, N.-T. (1994). “Concurrent genetic algorithms for optimization of large structures.” *Journal of Aerospace Engineering, ASCE*, 7(3), 276-296.

Adeli, H., and Soegiarso, R. (1999). *High-Performance Computing in Structural Engineering*, CRC Press, Boca Raton, Florida.

Afimiwala, K.A., and Mayne, R.W. (1974). “Optimum design of an impact absorber.” *Journal of Engineering for Industry, Transactions of the ASME*, 96(1), 124–130.

AISC. (2001). *Manual of steel construction-load and resistance factor design*, 3rd Ed., Chicago, IL.

Anderson, F.C., and Pandy, M.G. (2001). “Dynamic optimization of human walking.” *Journal of Biomechanical Engineering*, *Transactions of the ASME,* 123(5), 381-390.

ANSYS 7.0 User’s Manual (2002). Swanson Analysis Systems, Inc., Canonsburg, PA.

Arora, J.S. (1989). *Introduction to Optimum Design*, McGraw Hill, New York.

Arora, J.S. (1995). “Structural design sensitivity analysis: continuum and discrete approaches.” in *Advances in Structural Optimization*, edited by J. Herskovits, Kluwer Academic Publishers, Boston, MA, pp. 47-70.

Arora, J.S. (1999). “Optimization of structures subjected to dynamic loads.” *Structural Dynamic Systems Computational Techniques and Optimization*: *Optimization Techniques*, edited by C.T. Leondes, Gordon and Breach Science, New York, pp. 1–72.

Arora, J.S. (2002). “Methods for discrete variable structural optimization.” in *Recent Advances in Optimal Structural Design*, edited by S. Burns, Structural Engineering Institute, ASCE, 1801 Alexander Bell Drive, Reston, VA 20191-4400, pp. 1-40.

Arora, J.S., and Wang, Q. (2005). “Review of formulations for structural and mechanical system optimization.” *Structural and Multidisciplinary Optimization*, 30(4), 251-272.

Atkinson, K.E. (1988). *An Introduction to Numerical Analysis*, Wiley, New York.

Balling, R.J., and Sobieszczanski-Sobieski, J. (1996). "Optimization of coupled systems: A critical overview of approaches." *AIAA Journal*, 34(1), 6-17.

Balling, R.J., and Wilkinson, C.A. (1997). "Execution of multidisciplinary design optimization approaches on common test problems." *AIAA Journal*, 35(1), 178-186.

Barclay, A., Gill, P.E., and Rosen, J.B. (1997). *SQP Methods and Their Application to Numerical Optimal Control*, Department of Mathematics, Report NA 97-3, University of California, San Diego, CA.

Bathe, K.-J. (1982). *Finite Element Procedures in Engineering Analysis,* Prentice-Hall, Englewood Cliffs, NJ.

Bendsøe, M.P., and Sigmund, O. (2003). *Topology Optimization, Theory, Methods and Applications*, Springer-Verlag, Berlin.

Ben-Tal, A., and Bendsøe, M.P. (1993). "A new method for optimal truss topology design." *SIAM Journal on Optimization*, 3(2), 322–358.

Ben-Tal, A., and Nemirovski, A. (1997). "Robust truss topology design via semidefinite programming." *SIAM Journal on Optimization*, 7(4), 991–1016.

Ben-Tal, A., Kočvara, M., Nemirovski, A., and Zowe, J. (2000). "Free material design via semidefinite programming: The multiload case with contact conditions." *SIAM Review*, 42(4), 695–715.

Bessonnet, G., Sardain, P., and Chesse, S. (2002). "Optimal motion synthesis- dynamic modeling and numerical solving aspects." *Multibody System Dynamics*, 8(3), 257-278.

Bessonnet, G., Seguin, P., and Sardain, P. (2005). "A parametric optimization approach to walking pattern synthesis." *The International Journal of Robotics Research*, 24(7), 523-536.

Betts, J.T., and Huffman, W.P. (1991). "Trajectory optimization on a parallel processor." *Journal of Guidance, Control, and Dynamics*, 14(2), 431-439.

Betts, J.T., and Huffman,W.P. (1993). "Application of sparse nonlinear programming to trajectory optimization." *Journal of Guidance, Control, and Dynamics*, 15(1), 198–206.

Betts, J.T., and Frank, P.D. (1994). “A sparse nonlinear optimization algorithm.” *Journal of Optimization Theory and Applications*, 82(3), 519-541.

Betts, J.T. (1998). “Survey of numerical methods for trajectory optimization.” *Journal of Guidance, Control, and Dynamics*, 21(2), 193–207.

Betts, J.T., and Huffman, W.P. (1999). “Exploiting sparsity in the direct transcription method for optimal control.” *Computational Optimization and Applications*, 14(2), 179-201.

Betts, J.T. (2000). “Very low-thrust trajectory optimization using a direct SQP method.” *Journal of Computational and Applied Mathematics*, 120(1), 27–40.

Biegler, L.T. (1988). “On the simultaneous solution and optimization of large scale engineering systems.” *Computers & Chemical Engineering*, 12(5), 357-369.

Biegler, L.T., Ghattas, O., Heinkenschloss, M., and Bloemen Waanders, B.v. (Eds.). (2003). *Large-Scale PDE-Constrained Optimization*, Lecture Notes in Computational Science and Engineering, Vol. 30, Springer, Berlin, 1–13.

Bottasso, C., and Croce, A. (2004). “Optimal control of multibody systems using an energy preserving direct transcript method.” *Multibody System Dynamics*, 12(1), 17-45.

Bracken, J., and McGill, J.T. (1973). “Mathematical programs with optimization problems in the constraints.” *Operations Research*, 21(1), 37-44.

Burns, S. (2002). *Recent Advances in Optimal Structural Design*, Structural Engineering Institute, ASCE, 1801 Alexander Bell Drive, Reston, VA 20191-4400.

Cervantes, A., and Biegler, L.T. (1998). “Large-scale DAE optimization using a simultaneous NLP formulation.” *AIChE Journal*, 44(5), 1038–1050.

Cervantes, A.M., Wächter, A., Tütüncü, R.H., and Biegler, L.T. (2000). “A reduced space interior point strategy for optimization of differential algebraic systems.” *Computers and Chemical Engineering*, 24(1), 39–51.

Chan, C.-M., Grierson, D.E., and Sherbourne, A.N. (1995). “Automatic optimal design of tall steel building frameworks.” *Journal of Structural Engineering, ASCE*, 121(5), 838-847.

Chevallereau, C., and Aoustin, Y. (2001). “Optimal reference trajectories for walking and running of a biped robot.” *Robotica*, 19(5), 557-569.

Choi, W.S., and Park, G.J. (2002). “Structural optimization using equivalent static loads at all time intervals.” *Computational Methods in Applied Mechanics and Engineering*, 191(19–20), 2077–2094.

Chopra, A. (2001). *Dynamics of Structures: Theory and Applications to Earthquake Engineering*, Prentice-Hall, Upper Saddle River, NJ.

Cramer, E.J., Dennis, J.E., Jr., Frank, P.D., Lewis, R.M., and Shubin, G.R. (1994). “Problem formulation for multidisciplinary optimization.” *SIAM Journal on Optimization*, 4(4), 754-776.

Cuthrell, J.E., and Biegler, L.T. (1986). “Simultaneous solution and optimization of process flow-sheets with differential equation models.” *Chemical Engineering Research and Design*, 64(5), 341-346.

Denavit, J., and Hartenberg, R.S. (1955). “A kinematic notation for lower-pair mechanisms based on matrices.” *ASME Journal of Applied Mechanics*, 22(2), 215–221.

Dissanayake, M.W.M.G., Goh, C.J., and Phan-Thien, N. (1991). “Time-optimal trajectories for robot manipulators.” *Robotica*, 9(2), 131-138.

Enright, P.J., and Conway, B.A. (1992). “Discrete approximation to optimal trajectories using direct transcription and nonlinear programming.” *Journal of Guidance, Control, and Dynamics,* 15(4), 994-1002.

Faraway, J.J., Zhang, X., and Chaffin, D.B. (1999). “Rectifying postures reconstructed from joint angles to meet constraints.” *Journal of Biomechanics*, 32(7), 733-736.

Frank, P.D., and Shubin, G.R. (1992). “A comparison of optimization-based approaches for a model computational aerodynamics design problem, *Journal of Computational Physics*, 98(1), 74-89.

Fuchs, M.B. (1982). “Explicit optimum design.” *International Journal of Solids and Structures*, 18(1), 13–22.

Fuchs, M.B. (1983). “Explicit optimum design technique for linear elastic trusses.” *Engineering Optimization*, 6(4), 213-218.

Furukawa, T. (2002). “Time-subminimal trajectory planning for discrete non-linear systems.” *Engineering Optimization*, 34(3), 219-243.

Furusho, J., and Masubuchi, M. (1986). “Control of a dynamical biped locomotion system for steady walking.” *Journal of Dynamic System, Measurement, and Control*, 108(2), 111-118.

Gill, P.E., Murray, W., and Saunders, M.A. (2002). "SNOPT: an SQP algorithm for large-scale constrained optimization." *SIAM Journal on Optimization*, 12(4), 979–1006.

Gill, P.E. (2005). personal communication.

Goh, C.J., Edwards, N.J., and Zomaya, A.Y. (1993). "Feedback control of minimum-time optimal control problems using neural networks." *Optimal control applications and methods*, 14(1), 1-16.

Goswami, A. (1999). "Postural stability of biped robots and the foot rotation indicator point." *International Journal of Robotics Research*, 18(6), 523-533.

Grandhi, R.V., Haftka, R.T., and Watson, L.T. (1986). "Design-oriented identification of critical times in transient Response." *AIAA Journal*, 24(4), 649–656.

Gu, W., Gürdal, Z., and Missoum, S. (2002). "Elastoplastic truss design using a displacement-based optimization." *Computer Methods in Applied Mechanics and Engineering*, 191(27), 2907-2924.

Haftka, R.T. (1985). "Simultaneous analysis and design." *AIAA Journal*, 23(7), 1099-1103.

Haftka, R.T., and Kamat, M.P. (1989). "Simultaneous nonlinear structural analysis and design." *Computational Mechanics*, 4(6), 409–416.

Haftka, R.T., and Gürdal, Z. (1992). *Elements of Structural Optimization,* Third Revised and Expanded Edition, Solid Mechanics and Its Applications, Kluwer Academic, Boston, MA.

Haftka, R.T., Sobieszczanski-Sobieski, J. and Padula, S.L. (1992). "On options for interdisciplinary analysis and design optimization." *Structural Optimization*, 4(2). 65-74.

Hargraves, C.R., and Paris, S.W. (1987). "Direct trajectory optimization using nonlinear programming and collocation." *Journal of Guidance, Control, and Dynamics,* 10(4), 338-342.

Haug, E.J., and Arora, J. S. (1979). *Applied Optimum Design, Mechanical and Structural Systems*, John Wiley & Sons, Inc., New York, N.Y.

Haug, E.J., Choi, K.K., and Komkov, V. (1986). *Design Sensitivity Analysis of Structural Systems (Mathematics in Science and Engineering Series, Vol. 177),* Elsevier Science & Technology Books.

Herman, A.L., and Conway, B.A. (1995). "Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules." *Journal of Guidance, Control, and Dynamics,* 19(3), 592-599.

Herskovits, J., Mappa, P., and Juillen, L. (2001). "FAIPA SAND: An interior point algorithm for simultaneous analysis and design optimization." *WCSMO4, Fourth World Congress of Structural and Multidisciplinary Optimization* (held in Dalian, China).

Hilding, D., Klarbring, A., and Petersson, J. (1999). "Optimization of structures in unilateral contact." *Applied Mechanics Review,* 52(4), 139-160.

Hollerbach, J.M. (1980). "A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity." *IEEE Transactions on Systems, Man, and Cybernetics*. SMC-10 (11), 730-736.

Hoppe, R.H.W., Petrova, S.I., and Schulz, V. (2002). "Primal-dual Newton-type interior-point method for topology optimization." *Journal of Optimization Theory and Applications*, 114(3), 545–571.

Hrymak, A.N., McRae, G.J., and Westerberg, A.W. (1985). "Combined analysis and optimization of extended heat transfer surfaces." *Journal of Heat Transfer*, 107(3), 527-532.

Hsieh, C.C., and Arora, J.S. (1984). "Design sensitivity analysis and optimization of dynamic response." *Applied Mathematics and Optimization*, 43(2), 195–219.

Hull, D.G. (1997). Conversion of optimal control problems into parameter optimization problems." *Journal of Guidance, Control, and Dynamics*, 20(1), 57-60.

Hull, D.G. (2003). *Optimal Control Theory for Applications*, Springer-Verlag, New York.

IMSL® Math/Library 3.0 Manual (1997). Visual Numerical, Inc., Houston, TX.

Jarre, F., Kočvara, M., and Zowe, J. (1998). "Optimal truss design by interior-point methods." *SIAM Journal on Optimization*, 8(4), 1084–1107.

Kang, B.S., Choi,W.S., and Park, G.J. (2001). "Structural optimization under equivalent static loads transformed from dynamic loads based on displacement." *Computers and Structures*, 79(2), 145–154.

Kang, B.S., Park, G.J., and Arora, J.S. (2005). "Optimization of flexible multibody dynamic systems using the equivalent static load method." *AIAA Journal,* 43(4), 846-852.

Kang, B.S., Park, G.J., and Arora, J.S. (2006). “A review of optimization of structures subjected to transient loads.” *Structural and Multidisciplinary Optimization,* 31(2), 81-95.

Kaplan, M.L., and Heegaard J.H. (2001). “Predictive algorithms for neuromuscular control of human locomotion.” *Journal of Biomechanics*, 34(8), 1077-1083.

Kaplan, M.L., and Heegaard, J.H. (2002). “Second-order optimal control algorithm for complex systems.” *International Journal for Numerical Methods in Engineering*, 53(9), 2043-2060.

Khan, M.R., Willmert, K.D., and Thronton, W.A. (1978). “A new optimality criterion method for large-scale structures.” *Proceedings, AIAA/ASME 19th Structures, Structural Dynamics, and Materials Conference,* Bethesda, MD, April 1978, 47-50.

Khan, M.R. (1984). “Optimality criterion techniques applied to frames having general cross-sectional relationships.” *AIAA Journal*, 22(5), 669-676.

Kim, M.-S., and Choi, D.-H. (1998). “Min–max dynamic response optimization of mechanical systems using approximate augmented Lagrangian.” *International Journal for Numerical Methods in Engineering*, 43(3), 549–564.

Kirsch, U. (1993). *Structural Optimization: Fundamentals and Applications*, Springer-Verlag, Germany.

Kirsch, U., and Rozvany, G.I.N. (1994). “Alternative formulations of structural optimization.” *Structural and Multidisciplinary Optimization*, 7(1-2), 32-41.

Kirsch, U. (2000). “Combined approximations - a general reanalysis approach for structural optimization.” *Structural and Multidisciplinary Optimization*, 20(2), 97-106.

Kirsch, U., and Papalambros, Y. (2001). “Accurate displacement derivatives for structural optimization using approximate reanalysis.” *Computer Methods in Applied Mechanics and Engineering*, 190(31), 3945-3956.

Kirsch, U. (2002). *Design-Oriented Analysis of Structures*, Kluwer Academic Publishers, Dordrecht.

Kota, and N.N. (1996). *Computational Methods for Nonlinear Optimal Control*. Ph.D. Dissertation, Department of Mechanical Engineering, University of Iowa, Iowa City 52242.

Krishnamurthy, T. (2003). "Response surface approximation with augmented and compactly supported radial basis functions." *Proc. 44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conf.* (held in Norfolk, VA), Paper AIAA 2003-1748.

Leineweber, D.B., Bauer, I., Bock, H.G., and Schloder, J.P. (2003). "An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization. Part 1: Theoretical aspects." *Computers & Chemical Engineering*, 27(2), 157-166.

Lim, O.K., and Arora, J.S. (1987). "Dynamic response optimization using an active set RQP algorithm." *International Journal for Numerical Methods in Engineering*, 24(10), 1827–1840.

Lo, J., Huang, G., and Metaxas, D. (2002). "Human motion prediction based on recursive dynamics and optimal control techniques." *Multibody System Dynamics*, 8(4), 433-458.

Luo, Z.-Q. Pang, J.-S., and Ralph, D. (1996). *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, Cambridge.

Maar, B., and Schulz, V. (2000). "Interior point multigrid methods for topology optimization." *Structural and Multidisciplinary Optimization,* 19(3), 214-224.

McKeown, J.J. (1977). "Optimal composite structures by deflection variable programming." *Computer Methods in Applied Mechanics and Engineering*, 12(2), 155-179.

McKeown, J.J. (1989). "The design of optimal trusses via sequence of optimal fixed displacement structures." *Engineering Optimization*, 14(2), 159-178.

McKeown, J.J. (1998). "Growing optimal pin-jointed frames." *Structural and Multidisciplinary Optimization*, 15(2), 92-100.

Missoum, S., and Gürdal, Z. (2002). "Displacement-based optimization for truss structures subjected to static and dynamic constraints." *AIAA Journal,* 40(1), 154-161.

Missoum, S., Gürdal, Z., and Gu, W. (2002a). "Optimization of nonlinear trusses using a displacement-based approach." *Structural and Multidisciplinary Optimization*, 23(3), 214-221.

Missoum, S., Gürdal, Z., and Watson, L.T. (2002b). "A displacement based optimization method for geometrically nonlinear frame structures." *Structural and Multidisciplinary Optimization*, 24(3), 195-204.

Mortenson, M.E. (1985). *Geometric Modelling*, John Wiley & Sons, Inc., New York, NY.

Myers, R.H., and Montgomery, D.C. (2002). *Response Surface Methodology*, *Second Edition*, Wiley-Interscience, New York.

Neptune, R.R., and Hull, M.L. (1998). "Evaluation of performance criteria for simulation of submaximal steady-state cycling using a forward dynamic model." *Journal of Biomechanical Engineering*, 120(3), 334-341.

Nishiwaki, K., and Kagami, S. (2002). "Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired ZMP." *IEEE International Conference on Intelligent Robots and Systems*, 3, 2684-2689.

Orozco, C.E., and Ghattas, O.N. (1992a). "Sparse approach to simultaneous analysis and design of geometrically nonlinear structures." *AIAA Journal*, 30(7), 1877-1885.

Orozco, C.E., and Ghattas, O.N. (1992b). "Massively parallel aerodynamic shape optimization." *Computing Systems in Engineering*, 1-4, 311-320.

Orozco, C.E., and Ghattas, O.N. (1996). "Infeasible path design methods with application to aerodynamic shape optimization." *AIAA Journal*, 34(2), 217-224.

Orozco, C.E., and Ghattas, O.N. (1997). "A reduced SAND method for optimal design of nonlinear structures." *International Journal for Numerical Methods in Engineering*, 40(15), 2759-2774.

Outrata, J. V. Kočvara, M., and Zowe, J. (1998). *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints: Theory, Applications and Numerical Results*, Kluwer Academic Publishers, Dordrecht.

Pandy, M.G., Anderson, F.C., and Hull, D.G. (1992). ''A parameter optimization approach for the optimal control of large-scale musculoskeletal systems,'' *Journal of Biomechanical Engineering, Transactions of the ASME*, 114(4), 450-460.

Park, J., and Kim, K. (1998). "Biped robot walking using gravity-compensated inverted pendulum mode and computed torque control." *IEEE International Conference on Robotics and Automation*, 4, 3528-3533.

Pezeshk, S., Camp, C.V., and Chen, D. (2000). "Design of nonlinear framed structures using genetic optimization." *Journal of Structural Engineering, ASCE*, 126(3), 387-388.

Piegl, L., and Tiller, W. (1997). *The NURBS Book, 2nd Edition*, Springer-Verlag, New York, NY.

Ringertz, U.T. (1989). “Optimization of structures with nonlinear response.” *Engineering Optimization*, 14(3), 179-188.

Ringertz, U.T. (1992). “Numerical methods for optimization of nonlinear shell structures.” *Structural Optimization*, 4(3-4), 193-198.

Ringertz, U.T. (1995). “An algorithm for optimization of nonlinear shell structures.” *International Journal for Numerical Methods in Engineering*, 38(2), 299-314.

Roussel, L., Canudas-de-Wit, C., and Goswami, A. (2001). “Generation of energy optimal complete gait cycles for biped robots.” *Proc. of IEEE International Conference on Robotics and Automation*. (held in Leuven, Belgium), 3, 2036-2041.

Sadek, E.A. (1992). “Optimization of structures having general cross-sectional relationships using an optimality criterion method.” *Computers and Structures*, 43(5), 959-969.

Saidouni, T., and Bessonnet, G. (2003). “Generating globally optimized sagittal gait cycles of a biped robot.” *Robotica*, 21(2), 199-210.

Saka, M.P. (1980). “Optimum design of rigidly jointed frames.” *Computers and Structures*, 11(5), 411-419.

Schmit, L.A., and Fox, R.L. (1965). “An integrated approach to structural synthesis and analysis.” *AIAA Journal*, 3(6), 1104-1112.

Schulz, V. 2004: Simultaneous solution approaches for large optimization problems *Journal of Computational and Applied Mathematics*, 164(1), 629-641.

Sedaghati, R., and Esmailzadeh, E. (2003). “Optimum design of structures with stress and displacement constraints using the force method.” *International Journal of Mechanical Science*, 45(8), 1369-1389.

Seywald, H. (1994). “Trajectory optimization based on differential inclusion.” *Journal of Guidance, Control, and Dynamics*, 17(3), 480-487.

Shin, Y.S., Haftka, R.T., and Plaut, R.H. (1988). “Simultaneous analysis and design for eigenvalue maximization.” *AIAA Journal*, 26(6), 738-744.

Shubin, G.R. (1995). "Application of alternative multidisciplinary optimization formulation to a model problem for static aeroelasticity." *Journal of Computational Physics*, 118(1), 73-85.

Stackelberg, G. (1952). *The Theory of Market Economy*, Oxford University Press, Oxford, U.K.

Stope, M. (2003). *On Models and Methods for Global Optimization of Structural Topology*. Ph.D. Dissertation, Department of Mathematics, Royal Institute of Technology, KTH, Stockholm, 3–14.

Stope, M., and Svanberg, K. (2003). "A note on stress-constrained truss topology optimization." *Structural and Multidisciplinary Optimization*, 25(1), 62–64.

Stope, M., and Svanberg, K. (2004). "A stress-constrained truss-topology and material-selection problem that can be solved by linear programming." *Structural and Multidisciplinary Optimization*, 27(1-2), 126–129.

Thelen, D.G., Anderson, F.C., and Delp, S.L. (2003). "Generating dynamic simulations of movement using computed muscle control." *Journal of Biomechanics*, 36(3), 321-328.

Toogood, R.W. (1989). "Efficient robot inverse and direct dynamics algorithms using micro-computer based symbolic generation." *Proceedings of IEEE International Conference on Robotics and Automation*, 3, 1827-1832.

Tseng, C.H., and Arora, J.S. (1989). "Optimum design of systems for dynamics and controls using sequential quadratic programming." *AIAA Journal*, 27(12), 1793–1800.

Venkayya, V.B. (1971). "Design of optimum structures." *Computers and Structures*, 1(1-2), 265-309.

von Stryk, O., and Bulirsch, R. (1992). "Direct and indirect methods for trajectory optimization." *Annals of Operations Research*, 37(1-4), 357–373.

Vukobratović, M., and Borovac, B. (2004). "Zero-moment point – thirty five years of its life." *International Journal of Humanoid Robotics*, 1(1), 157-173.

Wang, Q., and Arora, J.S. (2005a). "Alternative formulations for structural optimization: application to trusses." *AIAA Journal,* 43(10), 2202-2209.

Wang, Q., and Arora, J.S. (2005b). "Alternative formulations for transient dynamic response optimization." *AIAA Journal,* 43(10), 2188-2195.

Wang, Q., Xiang, Y.-J., Kim, H.-J., Arora, J.S., and Abdel-Malek, K. (2005). "Alternative formulations for optimization-based digital human motion prediction." Paper 2005-01-2691, *2005 Digital Human Modeling for Design and Engineering Symposium*, Iowa City, IA, June 14-16, 2005.

Wang, Q., and Arora, J.S. (2006a). "Optimization of large-scale structural systems using sparse SAND formulations." *International Journal for Numerical Methods in Engineering,* to appear.

Wang, Q., and Arora, J.S. (2006b). "Alternative formulations for structural optimization: an evaluation using frames." *Journal of Structural Engineering, ASCE,* to appear.

Xiang, Y.-J., Chung, H.-J., Arora, J.S., and Abdel-Malek, K. (2006). *Digital Human Modeling and Virtual Reality for FCS - Simulation of Human Gait*, Technical Report No. VSR-06.01, CCAD, University of Iowa, Iowa City, IA 52242.

Yamaguchi, J., Soga, E., Inoue, S., and Takanishi, A. (1999). "Development of a bipdal humanoid robot control method of whole body cooperative dynamic biped walking." *IEEE International Conference on Robotics and Automation*, 1, 368-374.

Yamaguchi, G.T., and Zajac, F.E. (1990). "Restoring unassisted natural gait to paraplegics via functional neuromuscular stimulation: a computer simulation study." *IEEE Transactions on Biomedical Engineering*, 37(9), 886–902.

Yang, J., Marler, R.T., Kim, H., Arora, J., and Abdel-Malek, K. (2004). "Multi-objective optimization for upper body posture prediction." *Proc. 10th AIAA/ISSMO Multi-disciplinary Analysis and Optimization Conf.*, August, Albany, NY.

Yang, J., Marler, T., Beck, S., Kim, J., Wang, Q., Zhou, X., Pena Pitarch, E., Farrell, K., Patrick, A., Potratz, J., Abdel-Malek, K., Arora, J.S., and Nebel, K. (2006). "New capabilities for the virtual-human Santos^{TM}." *SAE 2006 World Congress*, Detroit, MI, April 3-6, 2006.

Zhou, M., and Rozvany, G.I.N. (1993). "DCOC: an optimality criteria method for large systems Part II: algorithm." *Structural Optimization*, 6(4), 250-262.