An Optimization Code for Nonlinear Transient Problems of a Large Scale Multidisciplinary Mathematical Model

By Koichi TAKASAKI

Aerospace Research and Development Directorate, Japan Aerospace Exploration Agency, Chofu, Tokyo, Japan

(Received April 25th, 2008)

This paper presents a program for the multidisciplinary optimization and identification problem of the nonlinear model of large aerospace vehicle structures. The program constructs the global matrix of the dynamic system in the time direction by the p-version finite element method (pFEM), and the basic matrix for each pFEM node in the time direction is described by a sparse matrix similar to the static finite element problem. The algorithm used by the program does not require the Hessian matrix of the objective function and so has low memory requirements. It also has a relatively low computational cost, and is suited to parallel computation. The program was integrated as a solver module of the multidisciplinary analysis system CUMuLOUS (Computational Utility for Multidisciplinary Large scale Optimization of Undense System) which is under development by the Aerospace Research and Development Directorate (ARD) of the Japan Aerospace Exploration Agency (JAXA).

Key Words: Large Scale Systems, Nonlinear Optimization, Dynamic Models, Nonlinear Constraints, Sparse Matrix

1. Introduction

1.1. Background

Thermal testing of a thermal protection system (TPS) for atmospheric re-entry vehicles has been conducted \(^1\) \(^2\) by JAXA’s Aerospace Research and Development Directorate and its predecessor organizations the JAXA Institute of Aerospace Technology (IAT) thermal structures section and the National Aerospace Laboratory thermal structures laboratory. In relation to these tests, several codes were also developed for identification and optimization analysis \(^3\) \(^4\). Based on these research results, development of this system began as a basic thermal analysis program in 2003.

There are various methods for solving the transient optimization problem. Nakamichi and Washizu \(^5\) solved nonlinear trajectory optimization by using finite elements in the time axis. Warner and Hodges \(^6\) solved optimal control problems using p-version finite element in the time axis. Other nonlinear trajectory optimization methods have also been developed using Hamilton’s weak principle \(^7\), the direct method using SQP \(^8\), and gradient projection \(^9\).

If we have to solve the optimization problem of a transient system which contains a relatively large structural model, specialized data structures and a solver algorithm are needed to reduce memory requirements and computational cost.

1.2. Overview of total analysis system

The system is designed to be flexible to allow it to accommodate future advances in analytical theory, structural concepts and operation procedures.

The current version of CUMuLOUS contains 84 modules and has about 52,000 lines of code. The program is constructed as an inverted tree of modules. The main module is located in the top of the system. Element modules, solver modules and sparse matrix operation modules are located under the main module. Basic data processing modules and the basic mathematical operation module are arranged in the bottom layer of the module tree structure. The version of each module is managed independently.

Target applications of this system are:

- TPS conceptual design: Quick parametric studies.
- Real-time thermal control of experimental equipment: Portable code will enable the entire finite element model to also be implemented in control systems.
- Conceptual design of lightweight, flexible thermal structures: Analysis of close-coupled aerothermodynamic / control / structural problems.

To be useful for these applications, the final system is planned to have the following functions:

- Basic thermal / structural analysis functions (static, transient, modal).
- Nonlinear objective function and constraint functions.
- Large-scale structural dynamics model coupled to multidisciplinary problems (with flight dynamics, fluid dynamics).
- Data structures and algorithms suitable for parallel computing.
- Ability to incorporate user-defined element modules and material characteristic modules that can be linked to the core system.

In order to implement these functions, the program was written considering the following points:

- The program modules are based on the same structural analysis system which forms the basis of the entire CUMuLOUS system.
- Both data structures and algorithms are designed for reduced memory requirements and parallel computing:
  - Hierarchical and sparse data structures for global matrices.
- The solver algorithm uses an iterative method that corresponds to the data structures.
  - Written in FORTRAN 90, a language suitable for matrix manipulation and parallel computing and widely used in this field.

CUMuLOUS adopts the following basic policies for element matrix construction:

1. Elements provide all constraints.
2. Nodes contain all variables.

Therefore, if the user defines an element, its application is not limited to the structural model but it can be used for boundary conditions (heat flux, pressure) and other constraint functions. For instance, the constraint for the analysis of flight dynamics is written in CUMuLOUS in the form of an element module. Due to the second policy above, all of the variables and parameters are stored in nodes, even shell element thickness or the sectional area of a rod element. This policy enables parametric studies of all variables in the global analysis model to be conducted simultaneously.

In consideration of the need to solve problems involving large-scale finite element structural models using parallel computing, the system uses sparse matrices and iterative solvers. The sparse matrices are composed of block sub-matrices related to each element. In other words, memory is allocated only for each element’s block matrices.

The optimization algorithm used in this program module has the following features:

- Simple correction / gradient phases both using the LSQR algorithm\(^{[10]}\). These are suitable for large structural systems and parallel computing.
- Introduction of pFEM in the time axis for transient analysis. Various types of transient problem can be solved, and analysis precision management is also possible.
- Each global iteration of the program yields a realistic solution, unlike the SQP method and other codes which solve the optimization problem and constraint simultaneously.

To implement this algorithm, a new solver module and meta-data operation modules are added to CUMuLOUS. The algorithm is described below.

2. Optimization Method Algorithm

2.1. Definition of nonlinear constraint equation in matrix form

The fundamental constraint equation of the transient problem is defined by the following matrix formula:

\[
M(t, p, x)\dot{x} + C(t, p, x)x + K(t, p, x)x = F(t, p, x) \quad (1)
\]

where

- \(M\) : Mass matrix
- \(C\) : Damping matrix
- \(K\) : Stiffness matrix
- \(F\) : Force vector
- \(t\) : Time
- \(p\) : vector that is independent from \(t\)
- \(x\) : vector that depends on \(t\)

To solve this transient problem, an one-dimensional p-version finite element between two points on the time axis is introduced (Fig. 1). Each element on the time axis has information on the variable \(x\) at time step \(\Delta t\). This element on the time axis is referred to as a ’meta element’ in this paper.

![Fig. 1. Basic meta element on the time axis.](image)

Within a time step of a meta element, the variable \(x\) is expressed by a superposition of higher order shape functions of the meta element.

The fundamental equation is transformed to the finite element formulation in the time axis:

\[
\int_{t_0}^t \left[ -\frac{dN_I}{dt} \sum_{J=1}^N \left( M_{I,J} \frac{dN_J}{dt} x_J \right) + N_I \left( C_{I,I} \sum_{J=1}^N \frac{dN_J}{dt} x_J + K_{I,I} \sum_{J=1}^N N_J x_J - F_I \right) \right] |J|\ d\tau = 0 \quad (2)
\]

where

- \(N_I\) : Shape function of the meta element
- \(x_J\) : \(x\) of each node of number \(I\) of meta element
- \(|J|\) : Determinant of the Jacobian of meta element
- \(\Delta t\) : Time step of the meta element
- \(\Delta t\) : Time step of the meta element

Note that each meta element has its own time axis. A user can connect elements in the time axis like building a model railway track, and apply them to various problems (Fig. 2).

![Fig. 2. Example of connection of meta elements.](image)
global vectors as follows:
\[ [h_{\Delta T}, h_p, h_X] \Delta T^T, p^T, X^T]_T - h_0 = 0 \] (3)
where
\[ \Delta T = [\Delta t_1, .., \Delta t_L]^T \]
\[ X = ... \]
where \( \hat{h} = [h^T, h^T_{\text{ext}}]^T \); \( h_{\text{ext}} \) is an additional constraint function which is defined by the user if required.

The sub-matrix of the global matrices \( h_{\Delta T}, h_p, h_X \) and the global vector \( h_0 \) for each meta element along the time axis has been found as follow:
\[ h_{\Delta T} = M_{\Delta T} + C_{\Delta T} + K_{\Delta T} \] (5)
\[ h_p = M_p + C_p + K_p - F_p \] (6)
\[ h_{\Delta t} = M_{\Delta t} + C_{\Delta t} - F_{\Delta t} \] (7)
\[ h_p = \left[ -N_j \sum_{j=1}^{N} M_{t,j} \frac{dN_j}{d\tau} x_j[J] \right] \left[ -1 \right] = \left[ \int_{-1}^{1} N_j F_t[J] d\tau \right] \] (8)

where
\[ M_{\Delta T} = \int_{-1}^{1} \left( -\frac{dN_j}{d\tau} M_{t,j} \frac{dN_j}{d\tau}[J]^{-1} \right) d\tau \] (9)
\[ C_{\Delta T} = \int_{-1}^{1} \left( N_j C_{t,j} \frac{dN_j}{d\tau} \right) d\tau \] (10)
\[ K_{\Delta T} = \int_{-1}^{1} (N_j K_{t,j} N_j[J]) d\tau \] (11)
\[ M_p = \left[ N_j \sum_{j=1}^{N} M_{t,j} \frac{dN_j}{d\tau} x_j[J] \right]^{-1} \] (12)
\[ C_p = \int_{-1}^{1} \left( N_j C_{t,j} \frac{dN_j}{d\tau} x_j \right) d\tau \] (13)
\[ K_p = \int_{-1}^{1} \left( N_j \sum_{j=1}^{N} K_{t,j} N_j x_j[J] \right) d\tau \] (14)
\[ F_p = \int_{-1}^{1} \left( N_j F_t[J] \right) d\tau \] (15)
\[ M_{\Delta t} = \int_{-1}^{1} \left\{ -\frac{dN_j}{d\tau} M_{t,j} \frac{dN_j}{d\tau} x_j(-\frac{1}{2}[J]^{-2}) \right\} d\tau \] (16)
\[ + \left[ N_j \sum_{j=1}^{N} M_{t,j} \frac{dN_j}{d\tau} x_j(-\frac{1}{2}[J]^{-2}) \right] \left[ -1 \right] \] (17)
\[ K_{\Delta t} = \int_{-1}^{1} \sum_{j=1}^{N} \left( \frac{1}{2} N_j K_{t,j} N_j x_j \right) d\tau \] (18)
\[ F_{\Delta t} = \int_{-1}^{1} \frac{1}{2} F_t d\tau \] (19)

2.2. Matrix data structure of constraint equation

A schematic of the data structures used to store the global matrices in Eq. (3) is shown in Fig. 3. In this schematic, each box filled with slanted lines denotes a non-zero submatrix of an element in the global matrices. The row of matrices \( h_{\Delta T}, h_p, h_X \) and \( h_0 \) is related to the residual of the constraint vector, and the columns of the matrices correspond to the variables \( \Delta t, p, x \) that influence the constraints (Fig. 3). \( h_{\Delta T} \) is a square matrix while \( h_p \) and \( h_{\Delta t} \) are rectangular matrices. To solve an optimization problem, the total row length of the global matrix must be less than the column length.

The sub-matrix of \( h_p \) related to each shape function in a meta element is referred to as a ‘Basic matrix’. A basic matrix constructed by superposition of element matrices using a technique similar to that of the usual finite element method. A basic matrix that relates to a different shape function on the time axis often becomes a near-zero matrix, especially the problem is linear. The reason is that the Legendre polynomial is employed as a shape function, which is an orthogonal function. Using this characteristic, the matrix norm of each basic matrix is examined whenever a total basic matrix is constructed, and the following multiplication in the iterative solver is not performed if the value of the norm is below a user-specified threshold.

Using p-version finite element along the time axis reduces the number of unknown variables much more than subdivision in the time direction by a first-order shape function, and accuracy management is also easy. Of course, analysis can be performed by setting the polynomial order of the shape function to unity. In this case the analysis is similar to the Crank-Nicolson method.

As shown in Fig. 2, there are no limits on meta element connections and constraints. In other words, constraints may be set not only for the initial condition but also at the terminal or mid-points of the time axis. Potential applications of this algorithm include analysis of orbital transfer, rocket stage separation and launch abort.

2.3. Optimization procedure

Under the nonlinear constraints defined in the previous subsection, the optimization problem is defined as:
\[ \text{minimize} \quad f(\Delta T, p, X) \] (19)
subject to \[ \hat{h}(\Delta T, p, X) = 0 \] (20)
where \( \hat{h} = [h^T, h^T_{\text{ext}}]^T \); \( h_{\text{ext}} \) is an additional constraint function which is defined by the user if required.
Optimization analysis is performed according to the following steps:

0. Initialize all variables.
1. Correction phase.
2. Calculate the projected gradient.
3. Terminate the analysis if the norm of the projected gradient is less than a given value.
4. Compute the search direction vector using the projected gradient vectors from the past iterations.
5. Perform a linear search along the search direction vector.
6. Return to step 1.

The equations used in steps 1 and 2 are shown below.

\[ \begin{bmatrix} I & \dot{h}_X \\ \dot{h}_X^T & O \end{bmatrix} \begin{bmatrix} r \\ \Delta \dot{h}_X \end{bmatrix} = \begin{bmatrix} \dot{h}_0 \\ 0 \end{bmatrix} : \text{step 1} \quad (21) \]

\[ \begin{bmatrix} I & \dot{h}_X \\ \dot{h}_X^T & O \end{bmatrix} \begin{bmatrix} g(\dot{X}) \\ -\lambda \end{bmatrix} = \begin{bmatrix} f_X \\ 0 \end{bmatrix} : \text{step 2} \quad (22) \]

where

\[ \dot{X} = [\Delta T^T, p^T, X^T]^T \quad (23) \]

\[ r : \text{residual of Eq.}(3) \]

\[ g : \text{projected gradient of } f \]

\[ \lambda : \text{Lagrange multiplier vector} \]

These two equations are of same form except that the matrix on the left hand side is transposed. Both are solved by the LSQR algorithm. In step 2, the matrix on the left hand side is in a similar form to the first-order optimality condition, which is also known as the KKT(Karush-Kuhn-Tucker) condition, except that the Hessian matrix is replaced by the identity matrix. Therefore, a large memory space to store a Hessian matrix is not required.

Because the multiplication procedure of the LSQR algorithm is performed for each submatrix of \( \dot{h}_X \), parallel computing is easily applied to this procedure.

3. Numerical examples

3.1. Example 1: Optimization problem of lumped mass

To verify the algorithm’s optimization analysis capability, a simple one-dimensional dynamics model of a lumped mass with a control input \( u \) was examined. The analytical model is as follows:

\[
\text{minimize} \quad \int_0^t u^2 \, dt \\
\text{under differential equation} \quad \begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u
\]

and boundary conditions

\[
\begin{bmatrix} x \\ v \end{bmatrix} \bigg|_{t=0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
\begin{bmatrix} x \\ v \end{bmatrix} \bigg|_{t=1} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}
\]

where

\[ x : \text{position} \]

\[ v : \text{velocity} \]

The theoretical solution of this problem is

\[ u = 6(1 - 2t) \quad (28) \]

The meta elements were connected in a series as shown in Fig. 4. The polynomial order of the state vector \( [x, v] \) of all of meta elements was set as two, while the polynomial order of control input \( u \) was constrained to unity (linear). The time step \( \Delta t \) within all meta elements was set to 0.1.

Fig. 4. Connection of meta elements.

Comparisons of the computational results with the theoretical solutions for the control input \( u \) and state vector \( [x, v] \) are shown in Figs. 5, 6 and 7 respectively.

3.2. Example 2: Optimization problem of truss structure

We now present an optimal control problem of a two-dimensional truss structure. The same meta element connections as in Example 1 were employed (Fig. 4). \( \Delta t \) for each meta element was set to 0.5sec, and the polynomial order in the time axis was set as follows:

from meta element 1 to meta element 5 : 3
from meta element 6 to meta element 10 : 4

The finite element structural analysis model is shown in Fig. 8. This structural model is constructed from two-
node, two-dimensional rod elements, which is in the CUMuLOUS element library. The mass matrix of each element is diagonalized. The following values were set in all elements:

Sectional area: 0.001 \( [m^2] \)
Young’s modulus: 1,000 \( [Pa] \)
Material density: 1,000 \( [kg/m^3] \)

The optimization problem was defined as follows:

\[
\text{minimize} \quad \int_0^5 f^2 dt \quad (29)
\]

under the following boundary conditions

- At node 1 in Fig. 4 (initial condition):
  \[
  u = 0 \quad \text{for all nodes} \quad (30)
  \]
  \[
  \frac{du}{dt} = 0 \quad \text{for all nodes} \quad (31)
  \]

- At node 9 in Fig. 4 (terminal condition; \( t=5 \)):
  \[
  z = 1 \at \text{point B} \quad (32)
  \]

- At node 11 in Fig. 4 (terminal condition; \( t=5 \)):
  \[
  z = 0 \at \text{point B} \quad (33)
  \]

where

\[
f: \text{control force along x-axis at point A}
\]
\[
u = [x, z]: \text{displacement at nodes}
\]

The shape function of the control force time history was also limited to linear. The convergence criterion of the total optimization problem is to have the projected gradient of the function defined by Eq. (23) as follows:

\[
\|g\| < 0.001 \quad (34)
\]

A total of 22 iterations were carried out to solve this problem. The computational results of the control force history and z-axis displacement of point B is shown in Figs. 9 and 10, respectively.

4. Summary

An optimization / identification algorithm has been developed for transient and nonlinear models of aerospace vehicle structures, and it has been implemented in computer code.

In this algorithm, in addition to a dynamic problem of an analysis model described by the finite element method,
a finite element formulation is applied to the time axis. p-version finite element is used within each time step in elements along the time axis.

Using this formulation, a user of the system can handle various transient problems. By setting an adequate polynomial order of the time-axis finite element, the accuracy in the time-domain can be set as required.

The entire matrix generated with this procedure is expressed by a first-order Taylor expansion form to the entire non-linear constraint vector by all variables.

The optimization problem is defined by repeating two phases: a constraint phase applied to the present estimated value of a variable, and a gradient phase which is a one-dimensional search along the projected gradient vector given by the gradient of the objective function and the constraint condition. The whole matrix forms of both phases are expressed by similar simple forms only to transpose the whole matrix, and the LSQR algorithm is applied to both phases.

In the second phase, the whole matrix form becomes similar to the KKT condition except that the Hessian matrix is replaced by the identity matrix. This means that the algorithm does not require a large memory space to store a Hessian matrix. Since LSQR is an iterative solver, and the algorithm uses only matrix-vector multiplication, the algorithm is suitable for parallel computing.

To validate the algorithm, a test problem was solved for a relatively simple structural model with one-dimensional dynamics, and very good agreement was found between the theoretical solutions and the computational results. Optimization analysis of a flexible truss structure with multiple constraint conditions applied at arbitrarily specified times was also conducted, and it was confirmed that this problem could be solved.

References

7) Hodges, D. H. and Bless, R. R.: Weak Hamiltonian Finite Element Method for Optimal Control Problems, J. Guid-