The Nonlinear Finite Element Toolbox (NLFET)

Matthew W. Roberts

April 13, 2002

Chapter 1

NLFET: An Open-Source Finite Element Toolkit for Matlab

1.1 Overview

The unique information requirements needed to implement a control strategy are unavailable using commercial, proprietary structural analysis or finite element codes (Balmès 1999). Because of this, researchers are typically required to develop custom analysis software when working in structural control. NLFET, an open-source, finite element code for MATLAB (1999), has been developed in conjunction with this research in order to provide a common framework and a set of software tools for structural control research and applications. MATLAB was chosen as the development platform because its scripting language is particularly adapted to numerical computation, allowing fast implementation, and because there are many powerful routines available for control [e.g., Control System Toolbox (1998), μ-Analysis and Synthesis Toolbox (2001)] and nonlinear analysis (Shampine and Reichelt 1997).

This chapter constitutes the documentation for NLFET and is released under the provisions of the GNU Free Documentation License which is reproduced in Appendix A. The actual code is released under the GNU Public License (GPL) which is reproduced in Appendix B. The GPL has been used for many software projects including the Linux Kernel, the gcc compiler, and GNU Emacs. The openness of the GPL has fostered tremendous collaboration in these projects which has resulted in very powerful, robust, and efficient software. It is hoped that the same will occur for NLFET.

This chapter assumes some familiarity with MATLAB, including basic scripting and programming techniques.

1.2 Problem Setup

An analysis is run with NLFET by setting up a general purpose data structure. Figure 1.1 shows a graphical layout of the GNUFEM data structure, including the rela-

Table 1.1: Required material properties for elements that have been developed for this dissertation.

element	E	Fy	G	alpha	c	nu
beam3d	x		x			x
sbeam3d	x					
truss3d	x					
unlsbeam3d	x	x		x		
unlsbeam3d.old	x	x		x		
vdamper3d					x	

Note: For dependent material properties (e.g. E, G, nu) not all values are required. Only the independent properties must be given.

tionships between the individual fields. A detailed description of the structure elements follows.

problem_name (*optional*) A string which gives the problem a name. This is typically used for logging or for graphics displays (e.g. the title of a plot).

num_nodes The total number of nodes defined. This is equivalent to $size(nodelist, 1)$.

num_elements The number of elements in the model. This is usually the same as the matlab expression `length(elements)`. However, the value of `num_elements` can be less than `length(elements)` if a subset of the elements (comprising elements 1-`num_elements`) is desired for an analysis.

num_dofs The number of degrees of freedom per node.

nodelist An nxc matrix of coordinates where n is the number of nodes and c is the number of coordinate axes (i.e. `c` = 2 for two-dimensional problems, `c` = 3 for three-dimensional problems. Thus, for a three-dimensional problem, the matrix `nodelist` would be

$$\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{1.1}$$

where x_i, y_i, and z_i are the x, y, and z coordinates, respectively, of node i. The vector of coordinates for element i are given by the MATLAB expression `nodelist(i,:)`.

materials An array of structures defining the materials (*e.g.* steel, aluminum, etc.) used in the finite element model. The fields are dependent on the material, with common fields being *E* for Young's Modulus of Elasticity, *nu* for the Poisson's Ratio, etc. Table 1.1 lists the required material fields for elements developed for this dissertation. Section 1.3 has further information on material properties. If

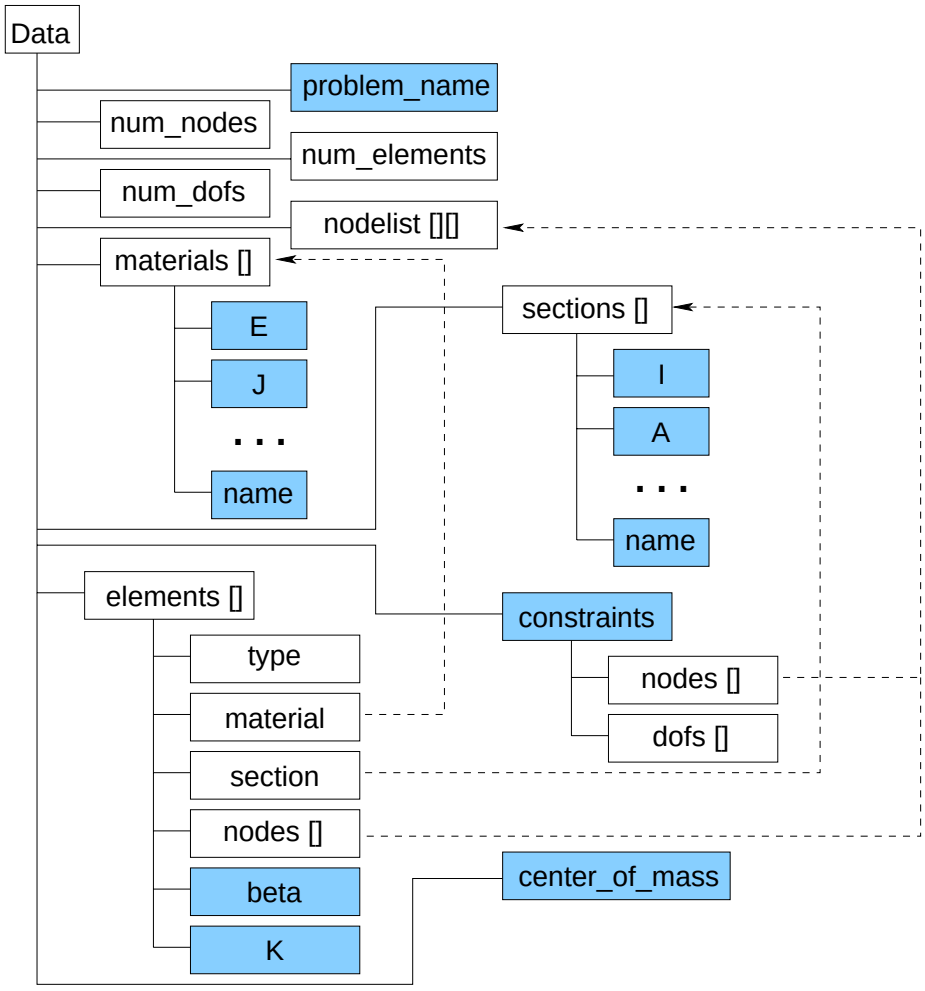

Figure 1.1: Schematic diagram of the data structure used in a GNUFEM model. Shaded boxes represent fields that are optional or that vary based on the element used. Dashed lines show relations between the fields of the data structure. Single bracket pairs ([]) denote one-dimensional arrays (vectors), while double bracket pairs are for two-dimensional arrays (matrices).

Table 1.2: Required section properties for elements in the base distribution.

element	A	Iy	Iz	J	Z
beam3d	x	x	x	x	
sbeam3d		x	x		
truss3d	x				
unlsbeam3d		x	x		x
unlsbeam3d.old		x	x		x
vdamper3d					

Note: For dependent section properties (e.g. Iy, Iz, J) not all values are required. Only the independent properties must be given.

the material defines an optional *name* field, this can be displayed when using the graphical interface.

sections An array of structures defining geometric properties. Common fields are *I* for the moment of inertia, *A* for the cross-sectional area, etc. The required fields depend on the type of element. Table 1.2 lists the required section fields for elements included in the base distribution. Section 1.4 has further information on section properties. If the section defines a *name* field, this will be displayed when using the `femplot` and other graphical interface commands.

elements An array of structures with the following fields:

type The type of the element stored as a string. ("beam", "truss", "nlspring", etc.) Element setup and assembly is then accomplished by calling a .m function file of the same name (e.g. beam.m, truss.m, nlspring.m, etc.). See Section 1.5 (page 8) of this chapter for more details.

material The index of the material array as defined above indicating the element material.

section The index of the section array as defined above indicating the element section.

nodes A vector defining the nodes which make up the element. Each member of the *nodes* vector is the index of a node stored in *nodelist*.

beta (default=0) Counterclockwise rotation (in degrees) of the element necessary to make the principal axes line up with the element axes. If this field is not defined, the default value of 0 will be used.

K (*optional*) The element stiffness matrix. Typically, this will be computed from the material, section, and geometric properties but it can be specified directly.

Specific elements may require other fields to be present. For example, a beam element may require additional information in order to formulate the stiffness matrix for semi-rigid connections (Chan and Chui 2000).

The above fields are required for any analysis using NLFET. Other fields of the *data* structure can be set to further define the problem. These additional fields are described below.

constraints An array of structures used to "lock" degrees of freedom together. The elements of each structure are:

nodes A vector of node numbers for the nodes that are locked together.

dofs A 1x6 boolean vector indicating which degrees of freedom are locked together. Locked degrees of freedom are specified with a non-zero entry in the corresponding vector entry.

A typical implementation of constraints is to specify rigid floors in a building system. The `constrain` function is used to enforce the constraints on the system matrices.

center_of_mass A matrix with row i containing the coordinates of the center of mass for constraint i.

1.3 Material Properties

Below are descriptions of material fields. The material properties are used to specify the fundamental constants which govern behavior of the element independent of its shape or geometric orientation.

E Young's modulus for an elastic material.

G Shear modulus.

nu Poisson's ratio. Typically, either G or nu will be required for a material. The relationship between the two is:

$$G = \frac{E}{2(1 + nu)}$$

alpha The post-yield elastic stiffness coefficient in the inelastic range. When this field is defined for an elastic-plastic material, the post-yield stiffness, E_p is given by:

$$E_p = (alpha)E$$

where E is the pre-yield elastic stiffness modulus. A perfectly plastic material would have $alpha = 0$.

Fy Yield stress of an elastic-plastic element.

name (*optional*) A string value identifying the material. When using graphical displays, this name may be displayed next to elements having the given material properties.

1.4 Section Properties

Below are descriptions of section fields. Section properties define the attributes of an element which depend on the shape of the element. Typically, finite element codes combine the material properties (above) and the section properties into a generic material property. NLFET splits the two so that redefinition of material properties is not necessary for different sized members composed of the same material.

Iy The moment of inertia about the element y axis.

Iz The moment of inertia about the element z axis. Normally, a W-section will have a larger value for Iy than for Iz.

J The polar moment of inertia (about the x-axis):

$$J = \int_{\Omega} r^2 dA = I_y + I_z$$

where r is the distance from the x-axis to the differential area, dA, integrated over the entire cross-section, Ω.

A Cross-sectional area.

d Nominal outside diameter (used, for example, for pipe members).

S Section modulus. This can be used to determine when an element begins to yield. For example, a beam element will begin to yield when the moment, M, is

$$M = SF_y \tag{1.2}$$

Z The plastic modulus. This is used to determine when a section has become completely plastic. For a beam, the so-called plastic moment, M_p, is the maximum moment a beam can sustain and is given by (Gere and Timoshenko 1990):

$$M_p = ZF_y \tag{1.3}$$

name (*optional*) A string value identifying the section. When using graphical displays, this name may be displayed next to elements having the given section properties.

1.5 Creating Elements with NLFET

1.5.1 The Element Callback Function

An element is added to NLFET by creating a MATLAB function which returns information about the element. This function will hereafter be referred to as the "callback" function of the element. The information returned by the callback function depends on a string-variable flag that is passed to the element. The prototype of the callback function is

Table 1.3: Alphabetical listing of required flags for a NLFET element callback function.

`'dof-vector'`	`'num-dof'`	`'num-nodes'`
`'stiffness-matrix'`	`'transformation-matrix'`	

Table 1.4: Numbers and Degrees of Freedom

Index	Degree of Freedom
1	x displacement
2	y displacement
3	z displacement
4	x rotation
5	y rotation
6	z rotation

```
element_name( flag, ...)
```

where `element_name` is the name of the element, `flag` is the string-variable flag indicating the data to be retrieved, and any additional arguments dependent on `flag` as indicated by the ellipsis.

1.5.2 Required Callback Flags

For an element to meet the minimal usability requirements in NLFET it must provide enough information to assemble a system stiffness matrix. Accordingly, all elements *must* process the flags in their callback functions: More details on each of the required callback flags is given below.

'dof-vector' A vector indicating the degrees of freedom which the element alters. The vector has length n, where n is the number returned from the callback function with the 'num-dof' flag. Each member of the vector is a number from one to six, inclusive, specifying a degree of freedom. The numbers with corresponding degrees of freedom are shown in Table 1.4. The calling prototype is

```
element_name( 'dof-vector')
```

'num-dof' Returns the number of degrees of freedom per node for the element. A two-dimensional element would have 3 or fewer degrees of freedom (x- and y-displacement and z-rotation). A three-dimensional element can have up to six degrees of freedom. The calling prototype is

```
element_name( 'num-dof')
```

'num-nodes' The number of nodes used to construct the element. The calling prototype is

```
element_name( 'num-nodes')
```

'stiffness-matrix' For a linear element, returns an element stiffness matrix, $\underline{\hat{k}}$, such that for absolute element nodal displacements, $\vec{\hat{d}}$, and element forces, $\vec{\hat{f}}$,

$$\underline{\hat{k}}\hat{d} = \hat{f} \tag{1.4}$$

The calling prototype is

```
element_name( 'stiffness-matrix', element, material,
    section, nodelist)
```

where `element`, `material`, and `section` are the element, material and section data structures, respectively, and `nodelist` is a matrix, all of which are part of the general problem data structure as defined in Section 1.2 of this chapter (p. 3). Thus, the stiffness matrix for element `n` can be retrieved with the following MATLAB command using the problem data structure:

```
% Retrieve stiffness matrix for element n
k = element_name( 'stiffness-matrix', ...
data.element(n), ...
data.material(data.element(n).material), ...
data.section(data.element(n).section), ...
data.nodelist)
```

Note that for maximum flexibility, NLFET has been designed such that the callback functions have no knowledge of the problem data structure. (The utility function, `elget`, which is included with NLFET, provides a simplified interface to the element callback functions.)

Nonlinear elements must also process the `'stiffness-matrix'` flag. The matrix returned, however, is the post-yielding, incremental stiffness matrix, $\underline{\hat{k}_y}$, such that for an increment of displacement, $\delta\vec{\hat{d}}$, the incremental forces, $\delta\vec{\hat{f}}$ on the element are

$$\underline{\hat{k}_y}\delta\vec{\hat{d}} = \delta\vec{\hat{f}} \tag{1.5}$$

As an example, consider a bilinear, non-hysteretic, element with stiffness given by the left-hand plot in Figure 1.2. The element has an initial stiffness K and a post-yielding, incremental stiffness αK. The left hand plot is simply the linear combination of the two plots on the right-hand side of the equals sign. Thus, the matrix returned from the `'stiffness-matrix'` callback, would be based on a linear element with stiffness αK. It should be noted that though αK is the incremental post-yield stiffness, the NLFET algorithms use it as a constant linear stiffness matrix throughout the analysis. Thus, the bilinear element of this example must use $(1-\alpha)K$ as the elastic incremental stiffness in the pre-yielding state.

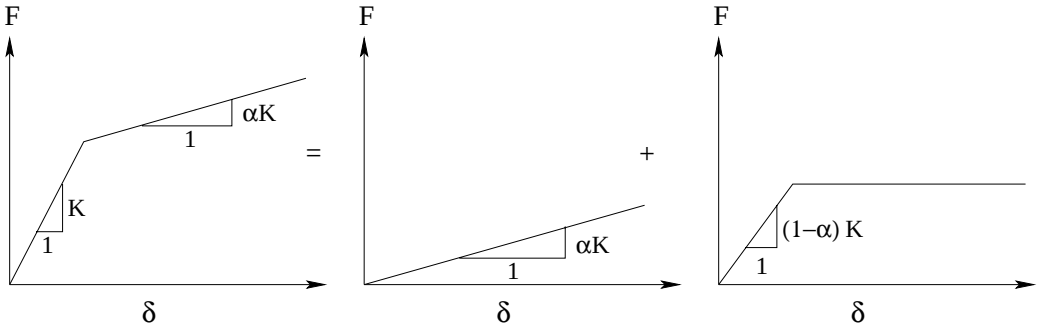

Figure 1.2: Force versus displacement for a bilinear element.

'transformation-matrix' Returns the matrix, $\underline{T}$, which transforms the global displacement vector, $\vec{d}$, into element coordinates, $\vec{\hat{d}}$:

$$\underline{T}\vec{d} = \vec{\hat{d}} \tag{1.6}$$

The $\underline{T}$ matrix also transforms the global forces, $\vec{f}$, to element forces, $\vec{\hat{f}}$:

$$\underline{T}\vec{f} = \vec{\hat{f}} \tag{1.7}$$

The calling prototype is

```
element_name( 'transformation-matrix', element, nodelist)
```

Note that the transformation matrix is based only on the element orientation so that material and section properties are not needed.

Note that if $\hat{k}$ is the matrix returned from the `'stiffness-matrix'` callback, and $\underline{T}$ is returned by the `'transformation-matrix'` callback, then the dimension of $\underline{T}^T\underline{\hat{k}}\underline{T}$ must be n x n, where n is the value returned by the `'num-dof'` callback, and T^T denotes the transpose of the matrix T.

1.5.3 Additional Callback Flags

Additional callback flags may be required for post-processing, for nonlinear problems (whether or not the element itself is nonlinear), or for "non-traditional" elements such as damping elements. These additional flags are listed below.

'damping-matrix' This flag is processed by "damping" elements. When processing this flag, an element damping matrix, $\hat{c}$, is returned such that for element nodal velocities, $\dot{\hat{d}}$, and element forces, $\hat{f}$,

$$\underline{\hat{c}}\dot{\vec{\hat{d}}} = \vec{\hat{f}} \tag{1.8}$$

The calling prototype is

```
element_name( 'damping-matrix', element, material,
    section, nodelist)
```

'element-forces' Returns the forces in the element based on the nodal displacements. The calling prototype is

```
element_name( 'element-forces', element, material,
  section, nodelist, D)
```

where `D` is a vector of nodal displacements for the element in *global* coordinates. The forces are returned, however, in local (element) coordinates. Thus, the element is required to make any transformations to local coordinates in order to compute the element forces, and the caller of the function is required to transform to global coordinates (e.g., using the `'transformation-matrix'` callback) if desired.

'pre-yield-stiffness-matrix' This flag is processed by nonlinear elements. The element returns the incremental stiffness matrix, $\hat{k}$, such that for element nodal displacements, $\hat{d}$, and element forces, $\hat{f}$,

$$\underline{\hat{k}}\vec{\hat{d}} = \vec{\hat{f}} \tag{1.9}$$

prior to any nonlinear behavior or yielding. This matrix is typically used for modal analysis of the pre-yielding system. The stiffness matrix, $\hat{k}$, must represent the complete stiffness of the member in its pre-yield state. Thus, in the example of the bilinear element of Figure 1.2, the matrix returned by the `'pre-yield-stiffness-matrix'` flag must be based on the initial stiffness, K, and not $(1-\alpha)K$.

'element-type' Returns a structure with more details on the element. The calling prototype is

```
element_name( 'element-type')
```

The returned structure can have any of the following fields, or the callback function can return an empty array to indicate that none of the fields are applicable.

keep_K Setting to a nonzero value will keep the element stiffness matrix, $\hat{k}$, stored for future use (post-processing, nonlinear time history analysis, etc.) so that reassembly will not be required.

nonlinear Set to a nonzero value for nonlinear elements.

damping_element If this is set to a nonzero value, then the element is a damping element, and must respond to the `'damping-matrix'` flag above.

1.6 Nonlinear Problems with NLFET

1.6.1 Nonlinear Fields in the Data Structure

Nonlinear problems in NLFET add extra fields to the general problem data structure (the general data structure is illustrated in Figure 1.1 on page 5). These extra fields allow element-specific data to be stored by nonlinear elements and assign dynamic state variables to the elements which are added to the state space system of (**??**) on page **??**. The element-specific data are typically used to speed calculations during a time history analysis. These nonlinear fields are defined as follows.

nl_elements An array of structures with element defined data for all nonlinear elements. The length of this array is equal to the number of nonlinear elements in the model. The fields of the structure are as follows.

num The element number of the nonlinear element.

data Element supplied data, typically used to speed calculations when performing time-history analysis. The data are returned by the `'nonlinear-data'` callback as described in Section 1.7.2 below.

state_begin An array indicating the index for the beginning state variables for each element.

state_end An array indicating the index for the ending state variables for each element.

num_states The total number of states used by all nonlinear elements in the model.

1.6.2 Callbacks Required of Nonlinear Elements for a Nonlinear Analysis

When performing a nonlinear analysis, nonlinear elements must process additional callback flags as explained below.

'nonlinear-data' Returns element-specific data to be used in a nonlinear analysis. The data can be of any type (scalar, matrix, structure, cell, etc.) and is available to the element, unchanged, during the analysis. The prototype is

```
  data = element_name( 'nonlinear-data', element, mate-
rial,
    section, nodelist)
```

It is imperative that only one matlab variable be returned when processing the `'nonlinear-data'` callback (i.e., `nargout = 1`). The first return value is all that is stored (`varargout1`), so if multiple data values are needed, the callback function should return a structure or a cell array. The analysis routines of NLFET have absolutely no knowledge of the format or requirements of the nonlinear data.

'nonlinear-global-force' This callback forms the core of the nonlinear analysis. When processing this callback, the element returns the forces on the member based on provided displacement, velocity, and state variables. The prototype is:

```
[F, zdot] = element_name( 'nonlinear-global-force',
  element, material, section, nodelist,
  u, v, z, nl_data)
```

where `u` and `v` are the element displacements and velocities, respectively, given in global coordinates, `z` is the state data, and `nl_data` is the data returned from the `'nonlinear-data'` callback. The return values are `F`, the forces on the member in global coordinates, and `zdot`, the time derivative of the element state variables. The element itself is fully responsible for the state variables—the NLFET analysis code has no knowledge of the state variables and what they represent or the coordinate system used.

As mentioned above, the `'nonlinear-global-force'` callback forms the core of the analysis procedure and is called for each iteration of each time step for all nonlinear elements. Because of this, this callback can be called hundreds of thousands of times during a typical time history analysis. It is therefore imperative that this routine be coded as efficiently as possible so as to minimize computational time.

1.7 Structural Control in NLFET

1.7.1 Nonlinear Fields in the Data Structure

Nonlinear problems in NLFET add extra fields to the general problem data structure (the general data structure is illustrated in Figure 1.1 on page 5). These extra fields allow element-specific data to be stored by nonlinear elements and assign dynamic state variables to the elements which are added to the state space system of (**??**) on page **??**. The element-specific data are typically used to speed calculations during a time history analysis. These nonlinear fields are defined as follows.

nl_elements An array of structures with element defined data for all nonlinear elements. The length of this array is equal to the number of nonlinear elements in the model. The fields of the structure are as follows.

- **num** The element number of the nonlinear element.
- **data** Element supplied data, typically used to speed calculations when performing time-history analysis. The data are returned by the `'nonlinear-data'` callback as described in Section 1.7.2 below.

state_begin An array indicating the index for the beginning state variables for each element.

state_end An array indicating the index for the ending state variables for each element.

num_states The total number of states used by all nonlinear elements in the model.

1.7.2 Callbacks Required of Nonlinear Elements for a Nonlinear Analysis

When performing a nonlinear analysis, nonlinear elements must process additional callback flags as explained below.

'nonlinear-data' Returns element-specific data to be used in a nonlinear analysis. The data can be of any type (scalar, matrix, structure, cell, etc.) and is available to the element, unchanged, during the analysis. The prototype is

```
  data = element_name( 'nonlinear-data', element, mate-
rial,
    section, nodelist)
```

It is imperative that only one matlab variable be returned when processing the `'nonlinear-data'` callback (i.e., `nargout = 1`). The first return value is all that is stored (`varargout1`), so if multiple data values are needed, the callback function should return a structure or a cell array. The analysis routines of NLFET have absolutely no knowledge of the format or requirements of the nonlinear data.

'nonlinear-global-force' This callback forms the core of the nonlinear analysis. When processing this callback, the element returns the forces on the member based on provided displacement, velocity, and state variables. The prototype is:

```
[F, zdot] = element_name( 'nonlinear-global-force',
  element, material, section, nodelist,
  u, v, z, nl_data)
```

where `u` and `v` are the element displacements and velocities, respectively, given in global coordinates, `z` is the state data, and `nl_data` is the data returned from the `'nonlinear-data'` callback. The return values are `F`, the forces on the member in global coordinates, and `zdot`, the time derivative of the element state variables. The element itself is fully responsible for the state variables—the NLFET analysis code has no knowledge of the state variables and what they represent or the coordinate system used.

As mentioned above, the `'nonlinear-global-force'` callback forms the core of the analysis procedure and is called for each iteration of each time step for all nonlinear elements. Because of this, this callback can be called hundreds of thousands of times during a typical time history analysis. It is therefore imperative that this routine be coded as efficiently as possible so as to minimize computational time.

1.8 Element Documentation

This section documents finite elements included in the base distribution of NLFET .

Element beam3d

Description

This is a two-node, three-dimensional, linear element.

Element axes are defined as outlined in Balfour (1992). The element x axis lies on the line joining the element nodes. The y axis is then oriented to lie in the global x,y plane. The z axis is then defined using the "right hand rule."

Typically, w-sections used for beams will have $I_y > I_z$.

Required Material Properties

E, G or nu

Required Section Properties

Iy, Iz, J, A

Element sbeam3d

Description

This is a two-node, three-dimensional, linear element. It is a "simplified" element because axial torsion and stiffness are neglected when forming the stiffness matrix.

This element is identical to the beam3d element except that axial and torsional stiffness is neglected. Note that the lack of axial stiffness may requires the use of a constraint or a truss element for numerical stability.

Required Material Properties

E

Required Section Properties

Iy, Iz

Element truss3d

Description

This is a two-node, three-dimensional, linear element.

Linear, three-dimensional truss element.

Required Material Properties

E

Required Section Properties

A

Element unlsbeam3d.base

Description

This is a two-node, three-dimensional, bilinear, "simplified" element. Axial and torsional stiffnesses are neglected, thus a constraint or a truss element must be used to provide axial stiffness if desired. If the pre-yielding stiffness is given by EI, then the post-yielding stiffness is given by αEI. A perfectly elastic-plastic element can be defined by setting $\alpha = 0$. The state variables for this element are the angles of the "elastic" portions of the beam.

Element axes are defined as outlined in Balfour (1992). The element x axis lies on the line joining the element nodes. The y axis is then oriented to lie in the global x,y plane. The z axis is then defined using the "right hand rule."

Required Material Properties

E, $alpha$, Fy

Required Section Properties

Iy, Iz, Z

Element unlsbeam3d

Description

This is a two-node, three-dimensional, bilinear, "simplified" element. Axial and torsional stiffnesses are neglected, thus a constraint or a truss element must be used to provide axial stiffness if desired. If the pre-yielding stiffness is given by EI, then the post-yielding stiffness is given by αEI. A perfectly elastic-plastic element can be defined by setting $\alpha = 0$. The state variables for this element are the angles of the "elastic" portions of the beam.

Element axes are defined as outlined in Balfour (1992). The element x axis lies on the line joining the element nodes. The y axis is then oriented to lie in the global x,y plane. The z axis is then defined using the "right hand rule."

Required Material Properties

E, $alpha$, Fy

Required Section Properties

Iy, Iz, Z

Element unlsbeam3d.old

Description

This is a two-node, three-dimensional, bilinear, "simplified" element. Axial and torsional stiffnesses are neglected, thus a constraint or a truss element must be used to provide axial stiffness if desired. If the pre-yielding stiffness is given by EI, then the post-yielding stiffness is given by αEI. A perfectly elastic-plastic element can be defined by setting $\alpha = 0$. The state variables for this element are the angles of the "elastic" portions of the beam.

Element axes are defined as outlined in Balfour (1992). The element x axis lies on the line joining the element nodes. The y axis is then oriented to lie in the global x,y plane. The z axis is then defined using the "right hand rule."

Required Material Properties

E, $alpha$, Fy

Required Section Properties

Iy, Iz, Z

Element vdamper3d

Description

This is a two-node, three-dimensional, linear element.

Linear, three-dimensional viscous damper element.

Required Material Properties

c

Required Section Properties

APPENDIX A

GNU FREE DOCUMENTATION LICENSE

Version 1.1, March 2000

Preamble

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

Applicability and Definitions

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or

with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

Verbatim Copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

Copying in Quantity

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

- List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

- State on the Title page the name of the publisher of the Modified Version, as the publisher.
- Preserve all the copyright notices of the Document.
- Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- Include an unaltered copy of this License.
- Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
- Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties – for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

Combining Documents

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled “History” in the various original documents, forming one section entitled “History”; likewise combine any sections entitled “Acknowledgements”, and any sections entitled “Dedications”. You must delete all sections entitled “Endorsements.”

Collections of Documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted

document, and follow this License in all other respects regarding verbatim copying of that document.

Aggregation With Independent Works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

Future Revisions of This License

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License ”or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have no Invariant Sections, write “with no Invariant Sections” instead of saying which ones are invariant. If you have no Front-Cover Texts, write “no Front-Cover Texts” instead of “Front-Cover Texts being LIST”; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

APPENDIX B

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this

License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form)

with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution

system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

    NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

@smallexample @varone line to give the program's name and a brief idea of what it does. Copyright (C) @varyyyy @varname of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA. @end smallexample

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

@smallexample Gnomovision version 69, Copyright (C) 19@varyy @varname of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details. @end smallexample

The hypothetical commands @sampshow w and @sampshow c should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than @sampshow w and @sampshow c; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

@example Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

@varsignature of Ty Coon, 1 April 1989 Ty Coon, President of Vice @end example

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

REFERENCES

Balmès, E. (1999). “Sensors, degrees of freedom, and generalized modeshape expansion methods.” *International Modal Analysis Conference (IMAC)*, Orlando. 628–634.

Chan, S. and Chui, P. (2000). *Non-linear static and cyclic analysis of steel frames with semi-rigid connections*. Elsevier, Amsterdam.

Control System Toolbox (1998). The MathWorks, Inc., Natick, Massachusetts.

Gere, J. M. and Timoshenko, S. P. (1990). *Mechanics of Materials*. PWS-KENT Publishing Company, Boston, Massachusettes, 3rd edition.

MATLAB (1999). The MathWorks, Inc., Natick, Massachusetts.

μ-Analysis and Synthesis Toolbox (2001). The MathWorks, Inc., Natick, Massachusetts.

Shampine, L. F. and Reichelt, M. W. (1997). “The MATLAB ode suite.” *SIAM Journal on Scientific Computing*, 18(1), 1–22.