Wright Finite Element Method:

An easily extensible research-oriented finite element code

Joseph C. Slater

December 5, 2016

Contents

Wright Finite Element Method (WFEM) is designed to be an easily extensible research-oriented structural finite element code. The design concept is that all "parts" of the model are actions by external codes. The central finite element code knows nothing about any particular element or boundary condition type. It depends on a completely self-capable code for implementing the "method" of the element or boundary condition. The advantage of this approach is that the code is able to handle the introduction of new elements without any modification of existing parts of the package. Further, additional capabilities can be added to elements without disruption to the main body of the code.

1 Anatomy of the body of the code

The main body of FEM is currently under construction, and likely will be indefinitely. In terms of performing analysis, this is not all that bad, since the real work of the code is handled internally to the element codes. The ability to find mode shapes and natural frequencies, plot deflections, and find stresses are separate aspects from the main body, and are performed by the element codes.

1.1 Running WFEM

To run WFEM, set your MATLAB path to the WFEM directory and type `wfem`. WFEM will prompt you for an input file name. Alternatively, type `help wfem` for other means by which to start a run of WFEM.

1.2 Linear Dynamic Analysis

Finding mode shapes and natural frequencies of a linear model can be performed using `soeig.m` which uses Cholesky decomposition of the mass matrix and assumes nothing about the stiffness matrix, K, in terms of symmetry. However, no asymmetric capabilities have been added yet (the FEM code doesn't have the ability to make an asymmetric element).

1.3 Guyan Reduction

Guyan reduction of the model can be easily performed using `guyan.m`. In the automatic mode, it will remove DOFs with a mass/stiffness below a predefined threshold.

1.4 Plotting

The ability to be plotted is one that is defined internally to an element. The plotting routine knows nothing about elements, only about nodes, their locations, and how they are connected (via lines, surfaces...). Lines and patches are defined by what nodes they connect, not by geometry.

To draw lines within elements, see the beam3example.m element. Look for the lines variable. Appending the pair of node numbers to be connected by line to this array ensures that it will be drawn by WFEM. Explanation for how to draw a surface is also given in the comments.

To draw surfaces, see the generate section of the panel1.m element. The color of a quadrilateral patch is defined by panelcolor. Adding a patch to be drawn by listing its node numbers (clockwise or counterclockwise) followed by its color is all that's needed to assure that it is drawn in static and dynamic situations.

Plotting of modes is easily accomplished internally by WFEM by adding mode shapes to the undeformed nodal location, and plotting all predefined nodes, lines, and patches. Element authors need do nothing to enable plotting of mode shapes/deformed static shapes.

1.5 Plotting Stresses/Strains (Not completed)

Plotting stresses and strains is accomplishes by using the stress and strain matrices. These are actually stresses and strain levels corresponding to the appropriate lines and patches and are placed in those matrices at the correct locations by the appropriate element code.

2 Entering data

For the sake of ease of use, it is best to define nodes and element property types before elements are defined. An example input code is shown in listing 1 on page 7. The first thing that one notices about the file is that the '%' can be used to add comments throughout the file, at any location, even in the middle of data. The '%' must reside in the *first column* for the comment to be recognized as a comment character. That means *you cannot put a comment at the end of a line of data.*

Properties are defined after the 'element properties' line. Some flexibility is allowed for typographical errors, but this flexibility shouldn't be considered a poetic license. Each type of element can have its own method for defining element properties. Elements can also have multiple methods of data input. For example, it would be quite reasonable to allow for a beam element that has constant properties along its length. As a result, only 7 entries, ρ, A, I_{yy}, I_{zz}, J, E, and G would be required. For example, see the beam3 element documentation in section 7.6.3 regarding allowable material

Listing 1: A simple truss example, example.txt.

```
variables
%All of these actions are not the most efficient for this problem.
t=0.02
l=25
Ixx1=1/12*t^4
Ixx2=1/12*(3/4*t)^4
Ixx3=1/12*(1/2*t)^4
Iyy1=Ixx1
Iyy2=Ixx2
Iyy3=Ixx3
J1=0.95*(Ixx1+Iyy1)
J2=0.95*(Ixx2+Iyy2)
J3=0.95*(Ixx3+Iyy3)

element properties
% Beam format
% E G rho A1     A2        A3    J1 J2 J3 Ixx1 Ixx2 Ixx3 Iyy1 Iyy2 Iyy3
steel     t^2 (3/4*t)^2 (t/2)^2 J1 J2 J3 Ixx1 Ixx2 Ixx3 Iyy1 Iyy2 Iyy3
%Note that these were defined above. I can use variables in my input file.
% I also used "steel" as a property. WFEM has some of these predefined.
% Run "units" and type "who" to see variables/values available inside your
% input file

beam3example elements
%node1 node2 node3 pointnum (beam properties number)
1 3 2 1 1
3 5 4 1 1

nodes
% I can include comment lines
% node num, x y z, Node number isn't ever stored in nodes matrix
1 0 0 0
2 0 l/4 0
3 0 l/2 0
4 0 3*l/4 0
5 0 l   0

points
1 1 1 1

fix clamp
1
% The preceeding put a clamp boundary condition on node 1.

load
2 1 10

%fix pin
%1 0 0 1
%2 0 0 1
%
%fix surfaceball
%3 0 0 1
%
% The preceeding from fix pin on would create a simply supported
% beam in the x-y plane. Note that we needed to fix the translation
% in the z direction of the center node. We will still have torsion
% of the rod in it's middle about the x-axis, and extension of node
% 3 in the x-direction. Don't forget that the blank lines must be
% uncommented out to signify the end of data entry.

actions
modalanalysis
who
fs %dump sorted natural frequencies to the screen
% The stuff inside actions is simply executed at the wfem prompt. Any
% Matlab command can also be executed here. For example, you could double
% the mass matrix then do another modal analysis.
% This will display natural frequencies and prompt for mode shape display
%if uncommented
%modalreview
fsold=fs %Let's store them for later comparison
M=M/4; %Dividing M by 4 should double the natural frequencies
fs=[]; % WFEM won't run another modal analysis unless I force it to
```

input formats and how they are interpreted. The WFEM code makes no attempt to understand these values at all. It simply loads them and stores them where the element codes can easily find them. Element property indices are identified by their row number. In the example shown (See Listing 1), there are two element properties given. All elements defined later must refer to one of these two definitions unless the element has no properties (we're staying very flexible here.).

Nodes are defined subsequent to the `nodes` command (see section 2.5). Unlike elements, the first value on each line is the node number. This example shows some of the flexibility of the reader in that it allows nodes to be defined before or after the elements. In fact, you can alternately define nodes and elements. This is convenient if you have an existing model and would like to simply tack on an additional substructure to the end.

Elements are defined using '*elementname* `elements`' (in this case `beam3`). There must exist a function *elementname*`.m`. Rows subsequent to such a command are defined to be of that type. The definition of elements is concluded with a blank line. Keep in mind that the main body of the finite element code knows *nothing* about any element, so adding an element to the code means writing a function file to do all of the work of using the element.

2.1 Units

Like any code, as long as consistent units are used, the code doesn't care what units you are using. The presumption of the code, however, is that values are entered in mks units (kilogram, meter, seconds). To assist with this, some built in constants are available for doing conversions for the user. For example, to enter 1 inch using presuming mks units, one would instead type *1*in*. Conversions from mks to th inch-pound system are not available directly, but can always be achieved by inverting the conversions. Currently defined units are *in, ft, yard, lbf, lbm, slug, psi, deg, gm, pm, angstrom, nm, μm (mum), mm, cm* and *km*.

2.2 Defined Constants

The gravitational constant g is also defined to be 9.80665 (standard gravity is defined to be at sea level and latitude 45°.

Material propeties, E, G, ρ, are available instead of typing them out as the variables *aluminum6061* (also *aluminum*), *steelsheet,* and *steelhot* (also *steel*). Others can easily be added to `units.m`.

2.3 Variables

Variables can also be used to define geometries. Variables are generated by definitions (e.g. a=2) following the variable command (variable on a line by itself). Variable can be defined using any lowercase letter with the exceptions of i and j which are reserved for $i = j = \sqrt{-1}$. Variables can also be lowercase letters followed by a number. Other variable names may be used with care. The guidelines given here are provided to simplify selection of names. As long as the variable name doesn't conflict with an internally used variable name, it is acceptable to use. A warning will be generated if an illegal choice is used.
Example:

```
command
s=1
```

2.4 Parameters

The third, optional, argument to wfem is a vector of parameters that are used in the problem definition of the input file. If this vector is available, the input file may use variables *parameter(i)* where *i* has a numerical value between 1 and the total length of the input parameter vector. The advantage of parameters is that multiple similar cases can be run without having to repeatedly edit the input file. For instance,

```
for i=1:10
wfem('inputfilename.txt',[],i)
end
```

would run the case defined in inputfilename.txt to be run 10 times, with the quantity referred to as *parameter(1)* in the input file varying between 1 and 10.

2.5 Defining Nodes

Nodes are defined subsequent to the nodes command. the format is

```
nodes
nodenum xlocation ylocation zlocation
%This is a comment
```

Definition of nodes ends with a blank line. Note that the node numbers are listed in the format for convenience only. The actually assigned node number is incremented by one for each new line of data. That is, node numbers monotonically increase by one.

3 Meshing

Meshing in this code is different than meshing in a traditional finite element code. In traditional finite elements, a geometry of an object is defined, the internal region of the body is divided by algorithm into elements, then nodes are automatically generated. In WFEM, meshing is the repeating of a substructure multiple times in a one dimensional arrangement. To do this, the command bay element is entered. Everything between this command and the command repeat... is considered to be the definition of what one bay looks like. This should include *only* elements. Elements must be *bay capable*(See section 7), as the bay meshing requires some knowledge to be obtained regarding the elements. Including non-bay capable elements will lead to erroneous results. No error checking for this exist. The repeat command is formated as:
repeat bay l times. Attach M to N.
Here the nodes listed in M are the innermost nodes of the unit bay, and the nodes listed in N are the outermost nodes, or free nodes. Another way to think of this is that the nodes N are currently the furthest extending nodes in what will become a series of bays. The nodes M are the starting nodes of this unit. For the next bay, the starting nodes will be N.For example, the code segment:

```
bay element
beam3 element
1 2 1

repeat bay 3 times. Attach 1 to 2.
```

will create will create a four element model of a beam by creating the first element (connecting nodes 1 and 2) then creating 3 more elements.

Here l defines how many times the bay is to be repeated. M is a list of nodes defining the attachment nodes of the bay that are attached to the existing defined structure. These can be thought of as the starting nodes of the bay. These node numbers will correspond to nodes N of the next bay, where N is the list of the end, or exposed, nodes of the bay that subsequent bays will be attached to.

4 Boundary Conditions and Constraints

Boundary conditions and constraints are applied using the fix and constrain commands. They are coupled with the appropriate entity as constrain *clamp* or fix *pin*. The former command rigidly connects two nodes in all six degrees of freedom, and the second results in a pin connection at subsequently listed nodes. The commands are followed by lines describing individual boundary conditions/constraints, the format being specific to the type of constraint. The following table describes all available attachment types.

Listing 2: A meshing example., compactexample.txt.

```
variables
%Al of these actions are not the most efficient for this problem.
t=0.02
l=0.5
Ixx1=1/12*t^4
Ixx2=1/12*(3/4*t)^4
Ixx3=1/12*(1/2*t)^4
Iyy1=Ixx1
Iyy2=Ixx2
Iyy3=Ixx3
J1=0.95*(Ixx1+Iyy1)
J2=0.95*(Ixx2+Iyy2)
J3=0.95*(Ixx3+Iyy3)

element properties
% Beam format
% E G rho A1      A2        A3    J1 J2 J3 Ixx1 Ixx2 Ixx3 Iyy1 Iyy2 Iyy3
steel     t^2 (3/4*t)^2 (t/2)^2 J1 J2 J3 Ixx1 Ixx2 Ixx3 Iyy1 Iyy2 Iyy3
%Note that these were defined above.

nodes
% node num, x y z, Node number isn't ever stored in nodes matrix
1 0 0 0
2 1 0 0
3 1 1 0
4 0 1 0

beam3 elements
%node1 node2 node3 pointnum (beam properties number)
1 4 1

bay element
beam3 element
1 2 1
1 3 1
2 3 1
3 4 1
repeat bay 1 times. Attach 1 4 to 2 3.

%points
%1 1 1 0

fix clamp
1


%fix pin
%1 0 0 1
%2 0 0 1
%
%fix surfaceball
%3 0 0 1
%
% The preceeding from fix pin on would create a simply supported
% beam in the x-y plane. Note that we needed to fix the translation
% in the z direction of the center node. We will still have torsion
% of the rod in it's middle about the x-axis, and extension of node
% 3 in the x-direction. Don't forget that the blank lines must be
% uncommented out to signify the end of data entry.
```

Attachment name	Description	Input Format	NASTRAN Support
clamp	all degrees of freedom	node1 (node2), nodes must be coincident	Y
pin	Thrust bearing hinge	node1 (node2) udv	N
roller	Roller in track	node1 (node2) udv hinge, udv motion	N
ball	Ball joint	node1 (node2)	N
rod	rod with pins at each end	node1 node2	N
rbeam	rigid (clamped) beam	node1 node2	Y
surfaceball	1 translation restricted	node1 (node2) udv translation	N
surface	1 trans + 3 rots restricted	node1 (node2) udv translation	N
rigidbody	calculate inertia tensor relative to node	node1 0	N

1. Unit direction vectors must be orthogonal.

In cases of constraints, the appropriate DOFs of the second node are reduced (slaved) to those of node 1.

The final attachment name is a constraint condition where rigid massless beams are presumed to connect all nodes, reducing the model to the single node *node1*. All other constraints will be ignored under these circumstances. The dummy value 0 is inserted for compatibility with other constrain condition formats. The action [totalmass,INERTIATENSOR,CG]=findinertia is necessary to obtain the rigid body parameters. See the file examplerigidbeam.txt for example usage.

See example.txt (Listing 1) for an example application of a boundary condition.

5 Applied Static Loads

Application of a static load is performed using the load command. Subsequent lines list the node number, the degree of freedom number, and the amount of the load applied to that node at that degree of freedom. Loads must be in consistent units. See Listing 1 for an example of load application.

6 Actions

Just as element routines act on the modal, more global actions are performed by action routines. Many actions are performed by default when a model is built. These include assemblemk, applybcs, applyconstraint, and istrainforces, their function being clear by their names. Additional detail regarding these action can be gleaned from the help provided in the code (for example help assemblemk). Additional actions currently available follow. Actions are intelligent enough to recognize when they cannot be performed without another preceding analysis. In the case that another action must take place first, an error is produced stating explicitly what action was needed.

6.1 plotundeformed

The action plotundeformed plots the as-defined structure and geometry, a useful tool in finding date entry errors.

6.2 findinitialstrain

The action findinitialstrain will apply the initial strain inducing loads and return the initial global deflections in the variable X. The initially deformed structure is automatically drawn if graphical output is available.

6.3 staticanalysis

The action staticanalysis will apply the initial strain inducing loads and the prescribed loads and return the global deflections in the variable X. The deformed structure is automatically drawn if graphical output is available.

6.4 plotdeformed

The action plotdeformed will plot the deformed structure under the previously solved for deflections X.

6.5 modalanalysis

The action modalanalysis obtains the natural frequencies (in Hz) and mode shapes and stored them in the variables *fs* and *fms*. These results are also automatically saved to the restart file.

6.6 modalreview

The action modalreview is a poor-man script that runs through the results of modal analysis. Not very sophisticated, but it gets the job done.

6.7 reducedofs

The action reducedofs will reduce the degrees of freedom in the system to remove DOFS associated with boundary conditions and constraints. Matrices returned are *Mr*, *Kr* and *Tr* where

$$M_r = T_r^T M T_r \tag{1}$$

and

$$K_r = T_r^T K T_r \tag{2}$$

6.8 Find Inertia

[totalmass,INERTIATENSOR,CG]=FINDINERTIA solves for the inertia tensor relative to the center of gravity, as well as the center of gravity.

6.9 planefit

The action planefit fit the best possible plane to the nodes containing surface elements. For best results, this plane should b closer to perpendicular to the z axis than either of the others.

6.10 end

The action end will cause wfem to be exited, leaving the user back are the MATLAB prompt.

6.11 MATLAB commands

A great deal of flexibility is obtained by recognizing that any MATLAB command can be executed as an action. This allows manipulation of graphics, intermediate computations, and other miscellaneous actions not listed here to be performed. A warning will be given that the command are not explicitly defined, and that they are being passed directly to the MATLAB interpreter.

7 Elements

7.1 Element modes

Elements are implemented through modal function calls to m-files of the same name as the element. The mode is essentially "what action should be taken". Additional information is passed to the element as needed, appropriate to the mode. Some modes must exist for WFEM to be able to do anything with the element. Some are recommended, which is admittedly subjective. Recommended means that the elements can be read into the data structure, and I think those modes are not necessary for elementary types of analysis (mode shapes, static deflection, linear simulation). Optional modes are those such as dynamic loads due to rotating coordinate systems, nonlinear stiffness terms, and stress and strain calculation capability.

7.2 Required element modes

7.2.1 generate

generate takes the element data and puts all available element information into the next available element entry in the element data structure. If element information is missing, generate may optionally generate it (i.e. internal nodes in beam3 are handled this way).

7.2.2 make

make takes element properties and nodal locations, generates the local coordinate finite element matrices, rotates them to global coordinates, and assembles them into the stiffness and mass matrices. It also generated the drawing properties for the element (points, lines, or surfaces.). These are used for easy and faster drawing of the deformed and undeformed structure.

make is also used to generate entries in the control matrix, as well as nonlinear force flags in the nonlinear force vector. These allow generation of linear matrix models for control law design (in the first case) as well as nonlinear time simulations.

7.3 Highly recommended modes

7.3.1 stressandstrain

7.3.2 buckle

7.4 Optional modes

7.4.1 numofnodes

The numofnodes mode is executed as
elementname('numofnodes',*eldata*)
where *eldata* is the row vector defining the element. It returns as an output the number of the defining parameters that are node numbers, starting from the left. This is required for meshfem.m, and thus is optional if the element isn't to be used for meshing.

7.5 Element tools available

7.5.1 Gauss Legendre Integration

The file gauss.m makes Gauss Legendre integration in 1-3 dimensions and up to 10th order simple with a single function call and a loop. Weights and gauss points are obtained using ([*pg*,*wg*]=gauss(*n*)) where *n* is a vector of length 1-3, each entry determining how many integration points to use in that direction. *pg* and *wg* are the resulting Gauss points and weights. *pg* is in the form

$$pg = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \end{bmatrix}$$

and *wg* is simply a list of weights corresponding to the points of *pg*.

7.6 Element types available

7.6.1 rod3

The rod element is a simple constant-cross section two-noded rod (spring) element with a consistent mass matrix. It has linear shape functions. Cross section properties are assumed to be constant.

inconsistent mass matrix should be added later

The element may be defined by any of the following:

```
rod3 elements
node1 node2 node3 pointnumber materialnumber
node1 node2 pointnumber materialnumber
node1 node2 materialnumber
```

Only the first two nodes and the material number matter. The remainder of the input options are available only for simplicity of switching from beam3 elements to rod3 elements. See section 7.6.3 on beam3 elements for details of the meaning of other values.

Material properties for the rod3 element are better referred to as rod properties. They are entered using the material properties command and can be entered as any of the following forms:

```
element properties
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3
E G rho A1 A2 J1 J2 Izz1 Izz2 Iyy1 Iyy2
E G rho A J Izz Iyy
E G rho A J Izz Iyy sx2 sy2 sz2 srx2 sry2 srz2 distype
 E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    sx2 sy2 sz2 srx2 sry2 srz2 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz3 mx3 my3 mz3 mrx3 mry3 mrz3
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 ...
    mx3 my3 mz3 mrx3 mry3 mrz3 sx3 sy3 sz3 srx3 sry3 srz3 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 ...
    lx2 ly2 lz2 lrx2 lry2 lrz2
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 ...
    lx2 ly2 lz2 lrx2 lry2 lrz2 ...
    mx3 my3 mz3 mrx3 mry3 mrz3 sx3 sy3 sz3 srx3 sry3 srz3 ...
    lx3 ly3 lz3 lrx3 lry3 lrz3
E rho A
E rho A m
E rho A m s
E rho A m s distype
E rho A m l distype
E rho A m s l
```

Property formats other than the last 6 add no capabilities but are available only for substituting rod3 elements for beam3 elements. m, s, and l stand for mean, standard deviation, or limit. A distype of 0 means Gaussian distribution. A distype of -1 means uniform distribution.

7.6.2 rod3t

The rod3t element is a simple constant-cross section two-noded rod (spring) element with a consistent mass matrix. It has linear shape functions. Cross section properties are assumed to be constant. In can handle initial strain definitions as well as thermal strains that change over time.

The element may be defined by any of the following:

```
rod3 elements
node1 node2 materialnumber
```

Material properties for the rod3 element are better referred to as rod properties. They are entered using the material properties command and can be entered as any of the following forms:

```
element properties
E rho A m s distype alpha alim thermalinput
E rho A m l distype alpha alim thermalinput
E rho A m s l       alpha alim thermalinput
```

The second 3 variables are for initial strains. m, s, and l stand for mean, standard deviation, or limit. A distype of 0 means Gaussian distribution. A distype of -1 means uniform distribution.

alpha is the thermal coefficient of expansion. alim defines the maximum positive deviation of alpha, $(\delta\alpha)_{\max}$. thermalinput allows temperature inputs to be applied to multiple elements simultaneously. α is allowed only to vary linearly at this time. The thermal input matrix is Bthermal, with columns corresponding to the thermalinput numbers. The matrix Bthermal must be multiplied by the *deviation* from the nominal temperature to obtain the psuedo-applied thermal load.

7.6.3 beam3

The beam3 element is a three-noded Euler-Bernoulli beam/rod/torsion element. It has sixth order shape functions for the beam deformations, and quadratic shape functions for torsion and extension deformations. Properties vary quadratically (or lower order) along the length of the element. The beam is asymmetric, allowing definition of the local coordinates for each element (see below). Second moment of area properties (I_{yy} and I_{zz}) must be defined in the principle area coordinate frame. J is the effective torsional polar moment of area. It should be equal to $I_{yy} + I_{zz}$ for circular cross sections, and less than the true polar moment of area for non-circular cross sections. For torsional inertia of the element, I_0 is calculated appropriately as $I_0 = I_{yy} + I_{zz}$. Nodes are numbered 1-3-2, so that the extra node (3) is in the middle of the beam. Node three must be placed precisely in the middle of the beam if it is defined explicitly. Deviation will cause errors.

	SS mode 1	SS mode 2	FF mode 2	FF mode 3
Continuous theory	9.87	39.48	22.40	61.62
Single 6th order Element	9.87	39.65	22.56	63.54
2 4th order Elements	9.91	43.82	22.42	70.17

Table 1: Quality of 6th order beam element for simply supported and free-free frequency determination. Coefficient to $\sqrt{EI/\rho A l^2}$.

The element may be defined by any of the following:

```
beam3 elements
node1 node2 node3 pointnumber materialnumber
node1 node2 pointnumber materialnumber
node1 node2 materialnumber
```

The point number is defined using the points command (see listing 1, page 7) in the same fashion as the nodes command (Section 2.5). The location of the point, along with nodes 1 and 2, defines the $x-y$ plane. The local x axis is from node 1 to node 2. The local y axis is from node 1 to the point, but only the component perpendicular to the x axis. The z axis is then known via the right hand rule.

Kind	$\omega =$	Boundary Cond.	Mode #	% Error
Rod (Extension)	$\frac{\pi}{l}\sqrt{\frac{E}{\rho}}$	FF	2	10.27
Rod (Extension)	$\frac{\pi}{2l}\sqrt{\frac{E}{\rho}}$	CF	1	0.375
Rod (Extension)	$\frac{3\pi}{2l}\sqrt{\frac{E}{\rho}}$	CF	2	21.4
Rod (Torsion)	$\frac{\pi}{l}\sqrt{\frac{G}{\rho}}$	FF	2	10.27
Rod (Torsion)	$\frac{\pi}{2l}\sqrt{\frac{G}{\rho}}$	CF	1	0.375
Rod (Torsion)	$\frac{3\pi}{2l}\sqrt{\frac{G}{\rho}}$	CF	2	21.4
Beam (Local y plane)	22.3733α	FF	2	0.7
Beam (Local y plane)	61.6728α	FF	3	3.1
Beam (Local y plane)	$\pi^2\alpha$	SS	1	<0.05
Beam (Local y plane)	$4\pi^2\alpha$	SS	2	0.4
Beam (Local z plane)	22.3733α	FF	2	0.7
Beam (Local z plane)	61.6728α	FF	3	3.1
Beam (Local z plane)	$\pi^2\alpha$	SS	1	<0.05
Beam (Local z plane)	$4\pi^2\alpha$	SS	2	0.4

Table 2: Single finite element natural frequency estimate error relative to continuous theory. Circular cross sections assumed, with $G = 1.5911 \times 10^{10}$, $E = 4.1369 \times 10^{10}$, $l = 3.048$, $\rho = 1.6608 \times 10^3$, $I_{yy} = I_{zz} = 7.0612 \times 10^{-7}$, and $A = 2.4322 \times 10^{-4}$. For beam frequencies, $\alpha = \sqrt{\frac{EI}{\rho A l^4}}$.

Material properties for the beam3 element are better referred to as beam properties. They are entered using the material properties command and can be entered as any of the following forms:

```
element properties
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3
E G rho A1 A2 J1 J2 Izz1 Izz2 Iyy1 Iyy2
E G rho A J Izz Iyy
E G rho A J Izz Iyy sx2 sy2 sz2 srx2 sry2 srz2 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    sx2 sy2 sz2 srx2 sry2 srz2 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz3 mx3 my3 mz3 mrx3 mry3 mrz3
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 ...
    mx3 my3 mz3 mrx3 mry3 mrz3 sx3 sy3 sz3 srx3 sry3 srz3 distype
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 ...
    lx2 ly2 lz2 lrx2 lry2 lrz2
E G rho A1 A2 A3 J1 J2 J3 Izz1 Izz2 Izz3 Iyy1 Iyy2 Iyy3 ...
    mx2 my2 mz2 mrx2 mry2 mrz2 sx2 sy2 sz2 srx2 sry2 srz2 ...
    lx2 ly2 lz2 lrx2 lry2 lrz2 ...
    mx3 my3 mz3 mrx3 mry3 mrz3 sx3 sy3 sz3 srx3 sry3 srz3 ...
    lx3 ly3 lz3 lrx3 lry3 lrz3
```

If the second format is used, a linear interpolation of the properties is presumed. If the third format is used, constant properties are assumed. In subsequent lines, initial deflections can be prescribed, deterministically, or stochastically (random). The character m stands for *mean* value, and r stands for *rotation* value. If a stochastic form is used, a distribution type must be prescribes. A distype of 0 means normal distribution with standard deviation of s values as prescribe. A distype of -1 means uniform distribution with bounds relative to the mean set by the s values. A truncated Gaussian distribution is demonstrated in the last line. There the l values are limits relative to the mean (just as the s values for a uniform distribution). Note that the dots illustrate continuation of the same line. Continuations of a line using the MATLAB notation ... can only be used in defining element properties. Torsional rigidity, J, must be less than or equal to $Iyy + Izz$ at any given cross section.

In addition, the variable DoverL can be defined to override the warnings that occur if the beam is defined to have $\frac{D}{l} < 0.1$. This is generally not advisable, as beams with a small value of $\frac{D}{l}$ tend to behave much more like Timoshenko beams.

7.6.4 inertia

The inertia element adds point masses and inertias. The inertia properties can be entered in the following forms and are treated as a "material type". In order of increasing complexity, the acceptable formats are:

```
element properties
m I
m Ixx Ixy Ixz Iyy Iyz Izz
m Ixx Ixy Ixz Iyy Iyz Izz l1x ly1 lz1 l2x l2y l2z
```

where the local x coordinate points in the direction of the vector [l1x l1y l1z], and the local y coordinate points in the direction of the vector [l2x l2y l2z]. Elements are defined using

```
 inertia elements
node materialnumber
```

7.6.5 panel1

A panel1 element is a method used to mark panels for obtaining geometric output later. Element properties are simply the *total* mass of the panel:

```
 element properties
m
```

Elements are defined using

```
 inertia elements
node1 node2 node3 node4 materialnumber
```

7.6.6 sensor

A sensor element provides outputs of displacement, velocity, or acceleration to y. Property values 1, 2, and 3 represent displacement, velocity, or acceleration respectively. All 6 DOF are measured. If you want fewer, ignore the extras, or dispose of the appropriate rows in the output matrices.

7.6.7 surfacenode

A surfacenode element is a method used to mark nodes on a surface of importance for any given reason. The set of surface nodes provides the points included in any accuracy calculations to determine surface quality. surfacenodes have no material property, and are defined by only the node numbers.