WinFElt Manual

Problem description

The problem description section is used to define the problem title and the number of nodes and elements in the problem. The problem description section is the only section which you cannot repeat within a given input file. The format for a problem-description is given below.

problem description
[**title** = *string*]
[**nodes** = *integer*]
[**elements** = *integer*]
[**analysis** = static | transient | modal | static-thermal | transient-thermal | spectral]

nodes = integer
elements = integer

These numbers will be used for error checking so the specifications given here must match the actual number of nodes and elements given in the definition sections. Note that the definitions for nodes and elements do not have to be given in numerical order, as long as nodes 1 ... m and elements 1 ... n (where m is the number of nodes and n is the number of elements) all get defined in one of the element and node definition sections in the file.

analysis = problem type

Defines the type of problem that you wish to solve. Currently it can either be static, transient, static-substitution, modal, static-thermal, transient-thermal, or spectral. If you do not specify anything, static analysis will be assumed.

Analysis parameters

The analysis parameters section is required only if you are doing some type of **transient**, **modal**, or **spectral analysis** (e.g., analysis=transient, analysis=spectral in the problem description section). For modal analysis it is simply used to set the type of element mass matrices that will be formed, but for transient and spectral analyses it contains information that further defines the problem and the parameters for the numerical integration in time.

analysis parameters
[**alpha** = *expression*]
[**beta** = *expression*]
[**gamma** = *expression*]
[**step** = *expression*]
[**stop** = *expression*]
[**Rm** = *expression*]
[**Rk** = *expression*]
[**nodes** = [*node-list*]]
[**dofs** = [*dof-list*]]
[**mass-mode** = **limped** | **consistent**]

The **alpha**, **beta**, and **gamma** parameters are used in numerical integration schemes (transient and transient-thermal analysis). **start**, **stop**, and **step** define the range of time or frequency interest for transient or spectral analyses. In transient analyses, **start** is meaningless and **duration** and **dt** can be used as aliases for **stop** and **step**, respectively. **Rk** and **Rm** are global Rayleigh (stiffness and mass) damping proportionality constants. The *node-list* is a comma or white space separated list of node numbers that are of interest in the analysis. Similarly, the *dof-list* is a list of the degrees of freedom (**Tx**, **Ty** , **Tz**, **Rx**, **Ry**, and **Rz**) that are of interest.

An object-definition section defines objects of a specified type. Objects include nodes, elements, materials, constraints, forces, and distributed loads. Each of these types of objects is discussed below. Multiple object-definition sections are allowed and the sections may occur in any order.

Nodes

Nodes are points in Cartesian space to which elements are attached. A node must have a constraint and may have an optional force. A node is identified by a natural number. The syntax is as follows:

nodes
node-definitions

where a *node-definition* takes the following form:

node-number
[**x** = *expression*]
[**y** = *expression*]
[**z** = *expression*]
[**constraint** = *constraint-name*]
[**force** = *force-name*]
[**mass** = *expression*]

The node-number starts the definition. Each node must have a unique number. If a Cartesian coordinate is not given then the coordinate of the previous node is used. Similarly, if no constraint is given then the constraint applied to the previous node is used. As above, the assignments can appear in any order and any number of times. As indicated above, some objects are identified by their name and some by their number. Elements and nodes have numbers while materials, forces, loads, and constraints have names.

Elements

Elements are linear, planar, or solid objects which are attached to nodes. Each element must have a material and may have optional loads. Furthermore, each element has a type, or definition. Like nodes, elements are identified by a unique natural number. Elements of specific type are defined with the following syntax:

element-type **elements**
element-definition

where an element-type is one of the following:

spring	**truss**
beam	**beam3d**
CSTPlaneStrain	**CSTPlaneStress**
iso2d-PlaneStrain	**iso2d-PlaneStress**
quad-PlaneStrain	**quad-PlaneStress**
timoshenko	**htk**
brick	**ctg**
rod	

and an *element-definition* has the following form:

element-number
nodes = [*node-list*]
[**material** = *material-name*]
[**load** = *load-name-list*]

The *element-number* starts the definition. Each element must have a unique number. If no material is given then the material applied to the previous element is used. The *load-name-list* is a list of up to three loads to apply to the element. The *node-list* is a comma or white space separated list of node numbers.

Each type of element requires a certain number of nodes and in some cases a special "null node" which is numbered zero may be used to indicate a gap or filler in the list.

Materials

Elements are made of a type of material. Each material has a name and certain physical properties not all of which may be used by anyone element. The syntax for defining materials is as follows:

material properties
material-definitions

where *material-definition* has the following form:

material-name
[**E** = *expression*] # Young's modulus
[**Ix** = *expression*] # moment of inertia about x-x axis
[**Iy** = *expression*] # moment of inertia about y-y axis
[**Iz** = *expression*] # moment of inertia about z-z axis
[**A** = *expression*] # cross-sectional area
[**J** = *expression*] # polar moment of inertia
[**G** = *expression*] # bulk (shear) modulus
[**t** = *expression*] # thickness
[**rho** = *expression*] # density
[**nu** = *expression*] # Poisson's ratio
[**kappa** = *expression*] # shear force correction
[**Rk** = *expression*] # Rayleigh damping coefficient (K)
[**Rm** = *expression*] # Rayleigh damping coefficient (M)
[**Kx** = *expression*] # thermal conductivity in the x-direction
[**Ky** = *expression*] # thermal conductivity in the y-direction
[**Ky** = *expression*] # thermal conductivity in the z-direction
[**c** = *expression*] # heat capacitance

The *material-name* starts the definition. If an attribute of a material is not specified then that attribute is zero. The assignments may occur in any order.

Constraints

Constraints are applied to nodes to indicate about which axes a node can move. The syntax for defining a constraint is as follows:

constraints
constraint -definitions

where *constraint-definition* has the following form:

constraint-name
[**tx** = **c** | **u** | *expression*] # boundary translation along x axis
[**ty** = **c** | **u** | *expression*] # boundary translation along y axis
[**tz** = **c** | **u** | *expression*] # boundary translation along z axis
[**rx** = **c** | **u** | *expression* | **h**] # boundary rotation about x axis
[**ry** = **c** | **u** | *expression* | **h**] # boundary rotation about y axis
[**rz** = **c** | **u** | *expression* | **h**] # boundary rotation about z axis
[**itx** = *expression*] # initial displacement along x axis
[**ity** = *expression*] # initial displacement along y axis
[**itz** = *expression*] # initial displacement along z axis
[**irx** = *expression*] # initial rotation about x axis
[**iry** = *expression*] # initial rotation about y axis
[**irz** = *expression*] # initial rotation about z axis

[**vx** = *expression*] # initial velocity along x axis
[**vy** = *expression*] # initial velocity along y axis
[**vz** = *expression*] # initial velocity along z axis
[**ax** = *expression*] # initial acceleration along x axis
[**ay** = *expression*] # initial acceleration along y axis
[**az** = *expression*] # initial acceleration along z axis

The *constraint-name* starts the definition. A value of **c** for a boundary condition indicates that the axis is constrained; a value of **u** indicates that the axis is unconstrained. An *expression* indicates a displacement (non-zero) boundary condition and may contain the **t** variable for time varying boundary conditions in transient analysis problems. The initial displacement, velocity and acceleration specifications are only used in transient problems. A value of **h** for a rotational boundary condition indicates a hinge. By default, all axes are unconstrained.

Forces

Forces, or point loads, may be applied to nodes. The syntax for a force definition is as follows:

forces
force-definitions

where a *force-definition* has the following form:

force-name
[**Fx** = *expression*] # force along x axis
[**Fy** = *expression*] # force along y axis
[**Fz** = *expression*] # force along z axis
[**Mx** = *expression*] # moment about x axis
[**My** = *expression*] # moment about y axis
[**Mz** = *expression*] # moment about z axis
[**Sfx** = *expression*] # frequency-domain spectra of force along x axis
[**Sfy** = *expression*] # frequency-domain spectra of force along y axis
[**Sfz** = *expression*] # frequency-domain spectra of force along z axis
[**Smx** = *expression*] # frequency-domain spectra of moment about x axis
[**Smy** = *expression*] # frequency-domain spectra of moment about y axis
[**Smz** = *expression*] # frequency-domain spectra of moment about z axis

The *force-name* starts the definition. If the force or moment is not specified then it is assumed to be zero. The *expressions* for forces may be time-varying. Time-varying expressions include the single variable **t** to represent the current time in the solution of a dynamic problem or consist of a list of discrete (time, value) pairs. Frequency varying expressions for spectra can also use **w** to represent the independent variable (radial frequency).

Loads

Distributed loads, or loads for short, are applied to elements. The syntax for a defining a distributed load is as follows:

distributed loads
load-definitions

where a *load-definition* has the following form:

load-name
[**direction** = *dir*] # direction
[**values** = *pair-list*] # local nodes and magnitudes

The *load-name* starts the definition. The *dir* is one of **LocalX**, **LocalY**, **LocalZ** (local coordinate system), **GlobalX**, **GlobalY**, **GlobalZ** (global coordinate system), **parallel**, or **perpendicular**. The *pair-list* is a

sequence of pairs. A pair is a node number and an expression enclosed in parentheses. The node number refers to the position within the element rather than referring to an actual node.

Expressions

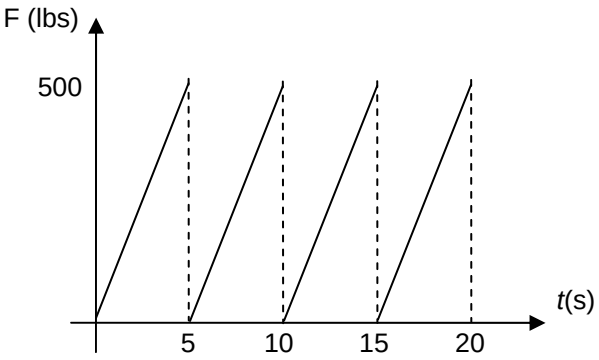
An *expression* can be either constant or time-varying. As discussed above, time-varying expressions contain the variable **t** or consist of a list of discrete (time, value) pairs. If a time-varying expression is given where a constant expression is expected, the expression is evaluated at time zero. An *expression* has one of the following forms, where all operators have the precedences and associativities given to them in the C programming language.

expression **?** *expression*: *expression*	# in-line conditional
expression **||** *expression*	# logical or
expression **&&** *expression*	# logical and
expression **|** *expression*	# integer inclusive or
expression **^***expression*	# integer exclusive or
expression **&** *expression*	# integer and
expression **==** *expression*	# equality
expression **!=** *expression*	# inequality
expression **<** *expression*	# less than
expression **>** *expression*	# greater than
expression **<=** *expression*	# less than or equal
expression **>=** *expression*	# greater than or equal
expression **<<** *expression*	# integer shift left
expression **>>** *expression*	# integer shift right
expression **+** *expression*	# addition
expression **-***expression*	# subtraction
expression ***** *expression*	# multiplication
expression / *expression*	# division
expression **%** *expression*	# integer remainder
-*expression*	# arithmetic negation
! *expression*	# logical negation
~ *expression*	# integer bitwise negation
(*expression*)	# enforce precedence
sin (*expression*)	# sine
cos (*expression*)	# cosine
tan (*expression*)	# tangent
pow (*expression*, *expression*)	# power (exponentiation)
exp (*expression*)	# exponential
log (*expression*)	# natural logarithm
log10 (*expression*)	# base-IO logarithm
sqrt (*expression*)	# square root
hypot (*expression*, *expression*)	# Euclidean distance
floor (*expression*)	# floor
ceil (*expression*)	# ceiling
Cmod (*expression*, *expression*)	# floating point remainder
Cabs (*expression*)	# absolute value
number	# literal value
t	# current time

Finally, a discretely valued expression has the following syntax, where the optional **+** indicates that the list represents one cycle of an infinite waveform.

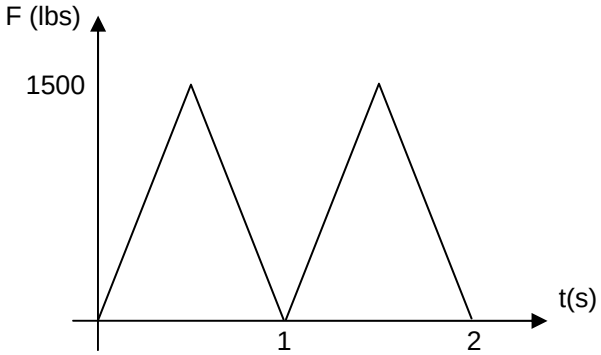
(*expression* ',' *expression*) ...[+]

Below are a few examples of loading expressions:



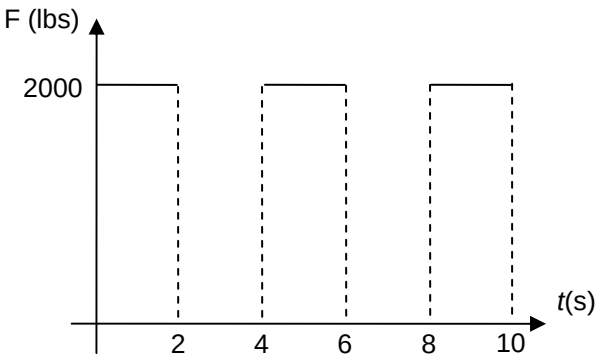
F = 100 * fmod(*t*,5.0)

F = (0,0.0) (4.99,500.0) +



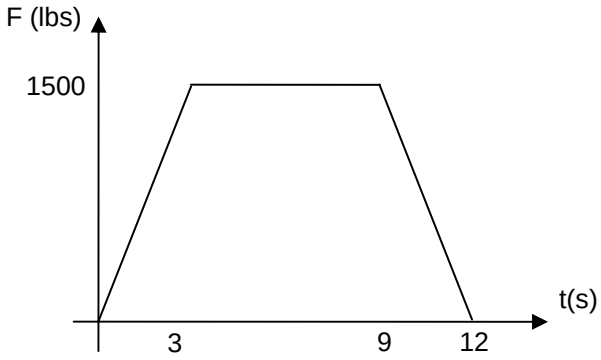
F = (fmod(*t*,1.0) < 0.5 ? 3000 * fmod(*t*,0.5) :
3000 * (0.5 - fmod(*t*,0.5)))

F = (0,0.0) (0.5,1500.0) (1,0) +



F = (0,2000) (1.99,2000) (2,0) (3.99,0) +

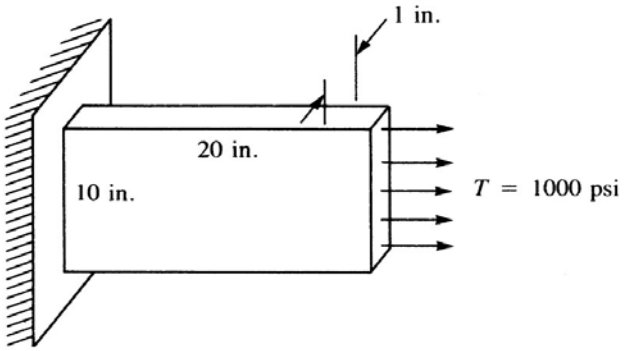
F = (fmod(*t*,4.0) < 2.0 ? 2000 : 0)



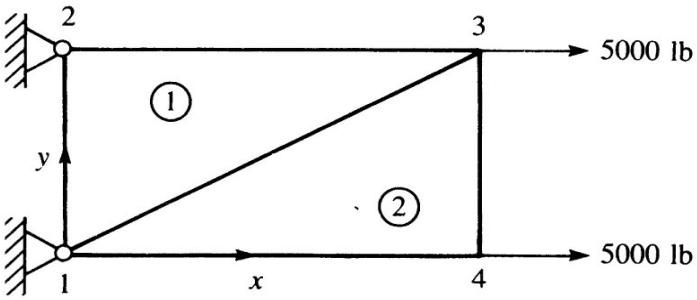
F = *t* < 3 ? 650/3 * *t* :
(*t* < 9 ? 650 : (*t* <= 12 ? 650/3 * (12 – *t*) : 0))

F = (0,0.0) (3,650) (9,650) (12,0) (13,0)

Use WinFElt to determine the nodal displacements and the element stresses.



Assume plane stress conditions. Let E = 30 x 10^6 psi, ν = 0.30, and t = 1 in. Consider the following discretization of two plane stress CST.



WinFElt input file:

```
problem description
title="CST Sample Problem (Logan 6.2, p.356)" nodes=4 elements=2

nodes
1 x=0      y=0.0    constraint=pin
2 x=0      y=10.0   constraint=pin
3 x=20.0   y=10.0   constraint=free force=point
4 x=20.0   y=0.0    constraint=free force=point

CSTPlaneStress elements
1 nodes=[1,3,2] material=steel
2 nodes=[1,4,3] material=steel

material properties
steel e=30e06 nu=0.30 t=1.0

forces
point fx=5000

constraints
pin  tx=c ty=c
free tx=u ty=u
end
```

WinFElt output file:

```
** CST Sample Problem (Logan 6.2, p.356) **

Nodal Displacements
-------------------------------------------------------------------------------
Node #      DOF 1       DOF 2       DOF 3       DOF 4       DOF 5       DOF 6
-------------------------------------------------------------------------------
  1             0           0           0           0           0           0
  2             0           0           0           0           0           0
  3    0.00060958  4.1633e-06           0           0           0           0
  4     0.0006637  0.00010408           0           0           0           0

Element Stresses
-------------------------------------------------------------------------------
  1:       1004.8      301.44      2.4019      1004.8      301.43     0.19566
  2:        995.2      -1.201     -2.4019       995.2     -1.2068    -0.13812


Reaction Forces
-----------------------------------
Node #     DOF     Reaction Force
-----------------------------------
  1        Tx              -5000
  1        Ty            -3002.4
  2        Tx              -5000
  2        Ty             3002.4
```