VibroSim COMSOL

Generated by Doxygen 1.8.5

Wed Jul 22 2020 13:51:59

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 BuildLater Class Reference

Public Member Functions

- function BuildLater (in M, in tag, in buildlaterclasses, in buildfcn)

Public Attributes

- Property buildlaterclasses
- Property is_built
- Property buildfcn

4.1.1 Detailed Description

This class is intended to represent wrapped models that are not instantiated immediately on creation, but need to be built at some later time. This is a common need, because often you want to define things together, but one of the things must be built much later in the model instantiation process than the other. For example, it often makes sense to integrate the specification of some boundary conditions with the geometry construction. Unfortunately, boundary conditions cannot be set until much later, after the physics nodes have been selected and created. Similar situations occur with meshing, material selection, and with autodetermining orientations through surface-normal extraction.

The way all this is accomplished is by instantiating BuildLater objects, which provide immediate lasting references to objects that don't exist yet. The BuildLater objects also store the callable object to be used to create the object when the time comes.

The BuildLater object is given one or more class names, that are placed in its 'buildlaterclasses' cell array of strings. When it is time to build a certain class of objects, use FindBuildLater() to extract all of the buildlater objects matching a particular class. Then you can call the buildfcn() method with appropriate arguments.

ALWAYS BE SURE TO SET THE is_built PROPERTY AFTER CALLING BUILDFCN!!!

NOTE: The buildfcn should almost always set the .parent property to what would otherwise be the 3rd argument to ModelWrapper(), so that the created COMSOL Objects can be properly destroyed.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 function BuildLater (in *M,* in *tag,* in *buildlaterclasses,* in *buildfcn*)

function object=BuildLater(M,tag,buildlaterclasses,buildfcn)

Create a BuildLater object that represents something that still needs to be built.

Parameters

M: ModelWrapper for top level model tag: Desired tag buildlaterclasses: String or cell array of strings representing classification(s) of this BuildLater object. buildfcn: Function to call to build the object. Parameters vary by buildlaterclass

4.1.3 Member Data Documentation

4.1.3.1 Property buildfcn

the function to call to build this object. The required arguments depend on context The buildfcn usually needs to manually set the .parent element so in debug mode we have a way to destroy the wrapped COMSOL object

4.1.3.2 Property buildlaterclasses

cell array of strings representing the 'class'es of BuildLater object that this object satisifes. These classes represent the different opportunities and calling conventions for BuildLater objects to be built.

4.1.3.3 Property is_built

true, or false. The routine that searches out and builds the objects should set this flag once it has built the object.

The documentation for this class was generated from the following file:

- BuildLater.m

4.2 ModelWrapper Class Reference

Public Member Functions

- function ModelWrapper (in M, in tagname, in parent)
- function setprop (in obj, in propname, in propvalue)
- function or (in a, in b)

Public Attributes

- Property tag

  The tag of this ModelWrapper – usually also the tag of node.
- Property node

  The wrapped COMSOL object (if present)
- Property index
- Property parent
- Property name

name of this model data

- Property getdomainselection
- Property mesh
- Property applymaterial

  this is a BuildLater for applying a material to a domain
- Property boundaryconditions

  a struct where each property name is the physics class and
- Property children

4.2.1 Detailed Description

ModelWrapper is MATLAB object that (usually) wraps COMSOL objects. The COMSOL object is placed in the 'node' property. You can dynamically add properties to instances with addprop(H,'PropertyName') see http://www.mathworks.com/help/matlab/matlab_oop/dynamic-properties–adding-properties-to-an-insta html#brffvja

Because this class is derived from dynamicprops, which is in turn derived from handle, you can pass it around and it will be passed by reference, not by value

4.2.2 Constructor & Destructor Documentation

4.2.2.1 function ModelWrapper (in *M,* in *tagname,* in *parent*)

[object] = ModelWrapper(M,tagname,parent): M – ModelWrapper object for top-level model (not currently used) tagname – name of object to create and enter into the database parent (optional) – if this is provided, it is stored in the object so that if you need to destroy the object later you can call parent.remove(tagname) object is already created and default superclass constructor already called

4.2.3 Member Function Documentation

4.2.3.1 function or (in *a,* in *b*)

4.2.3.2 function setprop (in *obj,* in *propname,* in *propvalue*)

4.2.4 Member Data Documentation

4.2.4.1 Property applymaterial

this is a BuildLater for applying a material to a domain

4.2.4.2 Property boundaryconditions

a struct where each property name is the physics class and

4.2.4.3 Property children

Cell array of children. Often used by BuildLater functions the value is a cell array of struct with .classnames (cell array of applicable boundary condition class names), .tag (geometry portion of tag) and buildfunc (anonymous build function) buildfunc is called as buildfunc(M,physics,bcobj)

4.2.4.4 Property getdomainselection

– getdomainselection is a function that when called as getdomainselection(model,geom,this_object) returns the entities of the object's domain selection

4.2.4.5 Property index

This is the numerical index of this object compared to all other created objects. Used so objects can be sorted by creation time

4.2.4.6 Property mesh

this is a BuildLater for the mesh, except for the main model object, for which this is the main mesh object

4.2.4.7 Property name

name of this model data

4.2.4.8 Property node

The wrapped COMSOL object (if present)

4.2.4.9 Property parent

This is node's parent COMSOL object. Present so we can call parent.remove() to delete the object in debug mode so it can be safely reconstructed.

4.2.4.10 Property tag

The tag of this ModelWrapper – usually also the tag of node.

The documentation for this class was generated from the following file:

- ModelWrapper.m

Chapter 5

File Documentation

5.1 add_cellstr_array.m File Reference

Functions

- function add_cellstr_array (in vec1, in vec2)

5.1.1 Function Documentation

5.1.1.1 function add_cellstr_array (in *vec1,* in *vec2*)

5.2 AddBoundaryCondition.m File Reference

Functions

- function AddBoundaryCondition (in M, in specimen, in object, in tag, in physicsclassorcellarray, in classnameorcellarray, in buildfcn)

5.2.1 Function Documentation

5.2.1.1 function AddBoundaryCondition (in *M,* in *specimen,* in *object,* in *tag,* in *physicsclassorcellarray,* in *classnameorcellarray,* in *buildfcn*)

AddBoundaryCondition creates, for each listed physics, a BuildLater object that runs buildfcn as a boundary condition creation function for that physics class (the physics name of the instantiated physics object is appended to the tag when boundary conditions are build for that physics). It is registered for a single physics class if physicsclassorcellarray is a string or under multiple physics classses if physicsclassorcellarray is a cell array. It is registered under a single BC class name if classnameorcellarray is a string, or under multiple BC classes if classnameorcellarray is a cell array

A BC class ("boundary condition class") represents the combination of physics and type of study for which the boundary condition applies.

See util/ModelWrapper.m for definition of boundaryconditions structure

buildfcn is called as buildfcn(M,physics,bcobj) which should fill out the bcobj structure. All created COMSOL boundary condition objects must be accessible either as bcobj.node, or through the node of an element of bcobj.children (or the children's children, etc.).

5.3 AddMeasurementToExpLog.m File Reference

Functions

- function AddMeasurementToExpLog (in explog, in using_datacollect)

5.3.1 Function Documentation

5.3.1.1 function AddMeasurementToExpLog (in *explog,* in *using_datacollect*)

ADDMEASUREMENTTOEXPLOG Add Measurement to Experiment Log ADDMEASUREMENTTOEXPLOG(explog) Add measurement to specified experiment log This function appends a measurement element at the end of experiment log and increments the next measurement number in dc:summary field

5.4 AddParamToExpLogSummary.m File Reference

Functions

- function AddParamToExpLogSummary (in explog, in paramname)

5.4.1 Function Documentation

5.4.1.1 function AddParamToExpLogSummary (in *explog,* in *paramname*)

[explog] = AddParamToExpLogSummary(explog,paramname) Add a parameter to summary node of the experiment log parameters explog - experiment log structure paramname - name of the parameter to add to the summary

5.5 AddParamToParamdb.m File Reference

Functions

- function AddParamToParamdb (in M, in varargin)

5.5.1 Function Documentation

5.5.1.1 function AddParamToParamdb (in *M,* in *varargin*)

ADDPARAMTOPARAMDB Adds A Parameter to A Local Paramdb Object ADDPARAMTOPARAMDB(M,field, string) Adds a String Value to A Local Paramdb ADDPARAMTOPARAMDB(M,field, number) Adds a Unitless Numeric Value to A Local Paramdb ADDPARAMTOPARAMDB(M,field, number, units) Adds a Numeric Value to A Local Paramdb NOTE: The parameter M is requested for consistency and for future compatibility should the local parameter database be shifted from a global variable to existing within the wrapped model object M. As of the current version it is not used and you can just pass []

5.6 AddView.m File Reference

Functions

- function AddView (in M, in specimen, in zoomanglefull, in position, in target, in up, in rotationpoint)

5.6.1 Function Documentation

5.6.1.1 function AddView (in *M,* in *specimen,* in *zoomanglefull,* in *position,* in *target,* in *up,* in *rotationpoint*)

AddView creates a named view for a specimen with the specified parameters This view is automatically used when viewing mode shapes, heating etc. Returns the specimen with view attribute added function specimen=AddView(M,specimen, zoomanglefull, position, target, up, rotationpoint)

specimen: Specimen object the view should be added to position: Position vector or cellstr array representing camera position target: Position vector or cellstr array representing camera target up: Unit vector or cellstr array representing up direction rotationpoint: Position vector or cellstr array representing rotation point

5.7 AddXducerContactProbe.m File Reference

Functions

- function AddXducerContactProbe (in M, in geom, in specimen, in xducercoord)

5.7.1 Function Documentation

5.7.1.1 function AddXducerContactProbe (in *M,* in *geom,* in *specimen,* in *xducercoord*)

function M = AddXDucerContactProbe(M,geom,specimen,xducercoord)

Parameters

xducercoord,:	coordinates of transducer contact (i.e. couplant_coord)

This is intended to be sequenced with a pipe after VibroPhysics to add a transducer contact probe to the relevant physics nodes.

5.8 ApplyMaterials.m File Reference

Functions

- function ApplyMaterials (in M, in geom)

5.8.1 Function Documentation

5.8.1.1 function ApplyMaterials (in *M,* in *geom*)

Apply materials, in creation order – by searching the TaggedObjectDB for objects with a 'applymaterial' property, and applying the given selections.

applymaterial property is called as applymaterial(M,geom,object)

5.9 AttachThinCouplantIsolators.m File Reference

Functions

- function AttachThinCouplantIsolators (in M, in geom, in specimen, in couplant_coord, in isolator_coords)

5.9.1 Function Documentation

5.9.1.1 function AttachThinCouplantIsolators (in *M,* in *geom,* in *specimen,* in *couplant_coord,* in *isolator_coords*)

5.10 BoundaryEvaluateAtFirstVertexCoord.m File Reference

Functions

- function BoundaryEvaluateAtFirstVertexCoord (in M, in geom, in boundaryentity, in fcn)

5.10.1 Function Documentation

5.10.1.1 function BoundaryEvaluateAtFirstVertexCoord (in *M,* in *geom,* in *boundaryentity,* in *fcn*)

5.11 BoundaryEvaluateFirstVertexCoord.m File Reference

Functions

- function BoundaryEvaluateFirstVertexCoord (in M, in geom, in boundaryentity)

5.11.1 Function Documentation

5.11.1.1 function BoundaryEvaluateFirstVertexCoord (in *M,* in *geom,* in *boundaryentity*)

5.12 BoundaryEvaluateVertexCoords.m File Reference

Functions

- function BoundaryEvaluateVertexCoords (in M, in geom, in boundaryentity)
  returns matrix of column vectors, each representing vertex coords

5.12.1 Function Documentation

5.12.1.1 function BoundaryEvaluateVertexCoords (in *M,* in *geom,* in *boundaryentity*)

returns matrix of column vectors, each representing vertex coords

5.13 BoundaryMeasureArea.m File Reference

Functions

- function BoundaryMeasureArea (in M, in geom, in boundaryentities)

5.13.1 Function Documentation

5.13.1.1 function BoundaryMeasureArea (in *M,* in *geom,* in *boundaryentities*)

5.14 BoundaryMeasureEdgeLength.m File Reference

Functions

- function BoundaryMeasureEdgeLength (in M, in geom, in boundaryentity)

5.14.1 Function Documentation

5.14.1.1 function BoundaryMeasureEdgeLength (in *M,* in *geom,* in *boundaryentity*)

5.15 buildabspath.m File Reference

Functions

- function buildabspath (in relpath)

5.15.1 Function Documentation

5.15.1.1 function buildabspath (in *relpath*)

5.16 BuildAllContinuityPairs.m File Reference

Functions

- function BuildAllContinuityPairs (in M, in geom, in physics, in object, in bcobj)

5.16.1 Function Documentation

5.16.1.1 function BuildAllContinuityPairs (in *M,* in *geom,* in *physics,* in *object,* in *bcobj*)

5.17 BuildBoundaryConditions.m File Reference

Functions

- function BuildBoundaryConditions (in M, in geom, in physics, in physicsclass)

5.17.1 Function Documentation

5.17.1.1 function BuildBoundaryConditions (in *M,* in *geom,* in *physics,* in *physicsclass*)

5.18 BuildBoundaryHeatConductivePairBC.m File Reference

Functions

- function BuildBoundaryHeatConductivePairBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc)

  Create a boundary condition representing heatflow permitted across a boundary. Please note that this can only be specified across objects in an assembly, not within a union. Because insulating is the default boundary condition, we have to find the automatically-generated pair objects and create continuity boundary conditions referencing them.

5.18.1 Function Documentation

5.18.1.1 function BuildBoundaryHeatConductivePairBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc*)

Create a boundary condition representing heatflow permitted across a boundary. Please note that this can only be specified across objects in an assembly, not within a union. Because insulating is the default boundary condition, we have to find the automatically-generated pair objects and create continuity boundary conditions referencing them.

5.19 BuildBoundaryHeatInsulatingBC.m File Reference

Functions

- function BuildBoundaryHeatInsulatingBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc)

  Create a boundary condition representing no heatflow permitted across a boundary.

5.19.1 Function Documentation

5.19.1.1 function BuildBoundaryHeatInsulatingBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc*)

Create a boundary condition representing no heatflow permitted across a boundary.

5.20 BuildBoundaryHeatSourceBC.m File Reference

Functions

- function BuildBoundaryHeatSourceBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in heatflowQb)

  Create a boundary condition representing a heat source on a face.

5.20.1 Function Documentation

5.20.1.1 function BuildBoundaryHeatSourceBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *heatflowQb*)

Create a boundary condition representing a heat source on a face.

5.21 BuildBoundaryHeatSourceBCs.m File Reference

Functions

- function BuildBoundaryHeatSourceBCs (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in heatflowQb)

5.21.1 Function Documentation

5.21.1.1 function BuildBoundaryHeatSourceBCs (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *heatflowQb*)

builds a set of boundary heat source BCs, one on each boundary according to the output of getfaceselectionfunc. The intensity of heat sources is set to heatflowQbs, which should be a cell array with length matching the number of returns from getfaceselectionfunc()OBSOLETE weakformpdephysicstag, if supplied, is the tag of a weak-form boundary PDE physics node to be used to minimize the calls to the functions in heatflowQbs. This is useful when those functions are calls to MATLAB that are slow. the weak-form boundary PDE physics node is evaluated in a separate step and converts the heatflowQB values into COMSOL results that can be evaluated directly with no more calls to MATLAB. This function will create the needed weak form PDE equations over the relevant selections

5.22 BuildContactor.m File Reference

Functions

- function BuildContactor (in M, in geom, in contactor, in shape, in pos, in normalvec, in angle, in leng, in width, in thickness)

5.22.1 Function Documentation

5.22.1.1 function BuildContactor (in *M,* in *geom,* in *contactor,* in *shape,* in *pos,* in *normalvec,* in *angle,* in *leng,* in *width,* in *thickness*)

5.23 BuildCouplant.m File Reference

Functions

- function BuildCouplant (in M, in geom, in couplant, in shape, in pos, in normalvec, in angle)

5.23.1 Function Documentation

5.23.1.1 function BuildCouplant (in *M,* in *geom,* in *couplant,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.24 BuildCrackElasticLayerBCs.m File Reference

Functions

- function BuildCrackElasticLayerBCs (in M, in geom, in physics, in crack, in bcobj)

5.24.1 Function Documentation

5.24.1.1 function BuildCrackElasticLayerBCs (in *M,* in *geom,* in *physics,* in *crack,* in *bcobj*)

5.25 BuildFaceContinuityBCs.m File Reference

Functions

- function BuildFaceContinuityBCs (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc)

5.25.1 Function Documentation

5.25.1.1 function BuildFaceContinuityBCs (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc*)

Build a set of continuity boundary conditions for a selection that presumably has identity contact pairs with another object.

We error out if no such pair is found.

Returns a cell array of boundary conditinos

5.26 BuildFaceDirectionalDisplacementBC.m File Reference

Functions

- function BuildFaceDirectionalDisplacementBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in direction, in magnitude, in harmonicperdirection, in harmonicpermagnitude)

5.26.1 Function Documentation

5.26.1.1 function BuildFaceDirectionalDisplacementBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *direction,* in *magnitude,* in *harmonicperdirection,* in *harmonicpermagnitude*)

NOTE: Direction MUST be a unit vector harmonicperdirection, harmonicpermagnitude are optional and if given result in the creation of a harmonic perturbation

5.27 BuildFaceDirectionalElasticDisplacementBC.m File Reference

Functions

- function BuildFaceDirectionalElasticDisplacementBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in direction, in magnitude, in harmonicperdirection, in harmonicpermagnitude, in stiffnessperunitarea, in dashpotcoeffperunitarea)

5.27.1 Function Documentation

5.27.1.1 function BuildFaceDirectionalElasticDisplacementBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *direction,* in *magnitude,* in *harmonicperdirection,* in *harmonicpermagnitude,* in *stiffnessperunitarea,* in *dashpotcoeffperunitarea*)

5.28 BuildFaceDisplacementBC.m File Reference

Functions

- function BuildFaceDisplacementBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in H, in R, in HarmonicPerH, in HarmonicPerR)

5.28.1 Function Documentation

5.28.1.1 function BuildFaceDisplacementBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *H,* in *R,* in *HarmonicPerH,* in *HarmonicPerR*)

H and R are the displacement constraints, defined below and in the COMSOL manual Note that if you provide H as a 1 dimensional cell array you must provide the elements in Fortran order (column by column) HarmonicPerR and HarmonicPerR are optional. If given, a hamonic perturbation node is created in addition to a static node with the given values Create a fixed boundary condition for a face identified via a selection function

5.29 BuildFaceFixedBC.m File Reference

Functions

- function BuildFaceFixedBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc)

  Create a fixed boundary condition for a face identified via a selection function.

5.29.1 Function Documentation

5.29.1.1 function BuildFaceFixedBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc*)

Create a fixed boundary condition for a face identified via a selection function.

5.30 BuildFaceSpringFoundationBC.m File Reference

Functions

- function BuildFaceSpringFoundationBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in k_A, in DampPerArea)

5.30.1 Function Documentation

5.30.1.1 function BuildFaceSpringFoundationBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *k_A,* in *DampPerArea*)

5.31 BuildFaceTotalForceBC.m File Reference

Functions

- function BuildFaceTotalForceBC (in M, in geom, in physics, in object, in bcobj, in getfaceselectionfunc, in forcevec, in isharmonicperturbation)

5.31.1 Function Documentation

5.31.1.1 function BuildFaceTotalForceBC (in *M,* in *geom,* in *physics,* in *object,* in *bcobj,* in *getfaceselectionfunc,* in *forcevec,* in *isharmonicperturbation*)

Create a boundary condition representing the total force vector on a face. if isharmonicperturbation (optional) is true, the harmonicPerturbation flag will be set to enable this for frequency studies

5.32 BuildFixedBC.m File Reference

Functions

- function BuildFixedBC (in M, in geom, in ShapeObj, in specimen, in shape, in pos, in normalvec, in angle, in length, in width)

5.32.1 Function Documentation

5.32.1.1 function BuildFixedBC (in *M,* in *geom,* in *ShapeObj,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle,* in *length,* in *width*)

5.33 BuildIsolator.m File Reference

Functions

- function BuildIsolator (in M, in geom, in isolator, in shape, in pos, in normalvec, in angle)

5.33.1 Function Documentation

5.33.1.1 function BuildIsolator (in *M,* in *geom,* in *isolator,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.34 BuildLater.m File Reference

Classes

- class BuildLater

5.35 BuildLaterWithNormal.m File Reference

Functions

- function BuildLaterWithNormal (in M, in geom, in tag, in pos, in buildfcn)

5.35.1 Function Documentation

5.35.1.1 function BuildLaterWithNormal (in *M,* in *geom,* in *tag,* in *pos,* in *buildfcn*)

5.36 BuildMeshDCObject.m File Reference

Functions

- function BuildMeshDCObject (in M, in geom, in mesh, in object, in meshobj, in dcprefix, in domainselectionfunc, in namededgesselectiontag, in sourcefaceselectionfunc, in targetfaceselectionfunc, in sweep_use_distributions)

5.36.1 Function Documentation

5.36.1.1 function BuildMeshDCObject (in *M,* in *geom,* in *mesh,* in *object,* in *meshobj,* in *dcprefix,* in *domainselectionfunc,* in *namededgesselectiontag,* in *sourcefaceselectionfunc,* in *targetfaceselectionfunc,* in *sweep_use_distributions*)

Mesh a block or similar object using datacollect parameters prefixed by dcprefix. Parameters:

Parameters

M,:	the class ModelWrapper containing our representation of the model
geom,:	the ModelWrapper containing our representation of the geometry
mesh,:	the ModelWrapper containing our representation of the top level mesh object
object,:	The ModelWrapper containing our block
meshobj,:	The ModelWrapper or BuildLater object to build
dcprefix,:	prefix on datacollect parameters, e.g. 'spc' for specimen
domainselectionfunct,-:	function returning domain entities to be meshed
namededgesselectiontag,-:	name of selection corresponding to edges of domain to be meshed

sourcefaceselectionfunc,-:	function returning source face boundary entities
targetfaceselectionfunc,-:	function returning target face boundary entities
sweep_use_-distributions	(true/false, optional default false): Should we Create the edge distributions to bound the element size along edges when using the sweep?

(note: object and namededgeselectiontag are not currently used, except object is passed to another function that doesn't use it)

If the given mesh type (dcprefix 'meshtype') is 'TETRAHEDRAL' then a simple FreeTet object is created. Otherwise the mesh type should be HEXAHEDRAL. In this case the meshing is done through a sweep from the specified source face to the specified target face. datacollect parameters: dcprefix 'meshtype': TETRAHEDRAL or HEXAHEDRAL dcprefix 'facemethod': For HEXAHEDRAL, 'FreeTri', 'FreeQuad', or 'Map dcprefix 'meshsize': For HEXAHEDRAL, Meshing size, in the sweep source/target face planes dcprefix 'sweepelements': For HEXAHEDRAL, Number of elements in the sweep direction

5.37 BuildMeshFreeTet.m File Reference

Functions

- function BuildMeshFreeTet (in M, in geom, in mesh, in object, in meshobj, in meshsizemin, in meshsize, in domainselectionfunc)

5.37.1 Function Documentation

5.37.1.1 function BuildMeshFreeTet (in *M,* in *geom,* in *mesh,* in *object,* in *meshobj,* in *meshsizemin,* in *meshsize,* in *domainselectionfunc*)

Mesh an object using Free Tetrahedral meshing Parameters:

Parameters

M,:	the ModelWrapper containing our representation of the model
geom,:	the ModelWrapper containing our representation of the geometry
mesh,:	the ModelWrapper containing our representation of the top level mesh object
object,:	The ModelWrapper containing our block
meshobj,:	The ModelWrapper or BuildLater to store the mesh in
meshsizemin,:	minimum mesh element size, or [] to disable
meshsize,:	maximum mesh element size, or [] to disable
domainselectionfunc,-:	function returning domain entities to be meshed

5.38 BuildMeshSweep.m File Reference

Functions

- function BuildMeshSweep (in M, in geom, in mesh, in object, in meshobj, in facemethod, in meshsize, in sweepelements, in domainselectionfunc, in namededgesselectiontag, in sourcefaceselectionfunc, in targetfaceselectionfunc, in use_distributions)

5.38.1 Function Documentation

5.38.1.1 function BuildMeshSweep (in *M,* in *geom,* in *mesh,* in *object,* in *meshobj,* in *facemethod,* in *meshsize,* in *sweepelements,* in *domainselectionfunc,* in *namededgesselectiontag,* in *sourcefaceselectionfunc,* in *targetfaceselectionfunc,* in *use_distributions*)

Mesh a block or similar object using a sweep operation Parameters:

Parameters

M,:	the class ModelWrapper containing our representation of the model
geom,:	the ModelWrapper containing our representation of the geometry
mesh,:	the ModelWrapper containing our representation of the top level mesh object
object,:	The ModelWrapper containing our block
meshobj,:	The object in which to store the mesh we are creating
facemethod,:	'FreeTri', 'FreeQuad', or 'Map
meshsize,:	Meshing size, in the sweep source/target face planes
sweepelements',:	Number of elements in the sweep direction
domainselectionfunc,-:	function returning domain entities to be meshed
namededgesselectiontag,-:	name of selection corresponding to edges of domain to be meshed
sourcefaceselectionfunc,-:	function returning boundary entities corresponding to source face
targetfaceselectionfunc,-:	function returning boundary entities corresponding to target face
use_distributions	(true/false, optional default false): Should we Create the edge distributions to bound the element size along edges? (note: object and namededgeselectiontag are not currently used)

This uses distributions, with element spacing determined by meshsize, on all edges adjacent to sourceface to feed the selected facemethod, operated on sourceface. A size node is also created to limit the size away from edges. Then that sourceface is swept to targetface. In addition a size object is created as well to bound the maximum size

5.39 BuildThinContactor.m File Reference

Functions

- function BuildThinContactor (in M, in geom, in contactor, in specimen, in shape, in pos, in normalvec, in angle, in leng, in width)

5.39.1 Function Documentation

5.39.1.1 function BuildThinContactor (in *M,* in *geom,* in *contactor,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle,* in *leng,* in *width*)

5.40 BuildThinCouplant.m File Reference

Functions

- function BuildThinCouplant (in M, in geom, in couplant, in specimen, in shape, in pos, in normalvec, in angle)

5.40.1 Function Documentation

5.40.1.1 function BuildThinCouplant (in *M,* in *geom,* in *couplant,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.41 BuildThinIsolator.m File Reference

Functions

- function BuildThinIsolator (in M, in geom, in isolator, in specimen, in shape, in pos, in normalvec, in angle)

5.41.1 Function Documentation

5.41.1.1 function BuildThinIsolator (in *M,* in *geom,* in *isolator,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.42 BuildVibroModel.m File Reference

Functions

- function BuildVibroModel (in M, in geometryfunc, in flawfunc, in physicsstudyfunc, in resultsfunc, in savefilename)

5.42.1 Function Documentation

5.42.1.1 function BuildVibroModel (in *M,* in *geometryfunc,* in *flawfunc,* in *physicsstudyfunc,* in *resultsfunc,* in *savefilename*)

BUILDVIBROMODEL Creates and Runs a COMSOL model of the vibrothermography process based on the given functions which instantiate the different portions of the simulation.

Parameters:

Parameters

geometryfunc,:	Function, called as specimen=GeometryFunc(M,geom,'specimen') Which should build the geometry and define materials. It also needs to specify appropriate classes of boundary conditions and to be used by the physics, below, as well as meshing instructions.

flawfunc,:	Function (or omit by passing []) that is to be used to add flaws to the geometry. Called as flaw=flawfunc(M,geom,specimen)
physicsstudyfunc,:	Function, called as physicsstudyfunc(M,geom,specimen,M.flaw) which defines a set of physics nodes and then a set of study nodes to perform the simulation.
resultsfunc,:	Function to call to set up Results tree. Called as resultsfunc(model). This code is intended to be pasted directly from Comsol Save As... m-file. You can use structured code if appropriate, but we don't recommend accessing the structured objects, as then you won't be able to run this code (for example) on results from running studies in a loaded .mph file.
savefilename,:	Optional filename to save the model, if given .

5.43 BuildWithNormals.m File Reference

Functions

- function BuildWithNormals (in M, in geom)

5.43.1 Function Documentation

5.43.1.1 function BuildWithNormals (in *M,* in *geom*)

NOTE: Since we use GetNormal() rather than GetOutwardNormal() if you call this multiple times, such as in debug mode you might get flipped normals – from getting the normal to the thing built in the previour round !!!

5.44 BuildWrappedModel.m File Reference

Functions

- function BuildWrappedModel (in M, in object, in parent, in varargin)

  Build a wrapped model inside a pre-existing wrapper.

5.44.1 Function Documentation

5.44.1.1 function BuildWrappedModel (in *M,* in *object,* in *parent,* in *varargin*)

Build a wrapped model inside a pre-existing wrapper.

5.45 calc_straincoefficient.m File Reference

Functions

- function calc_straincoefficient (in basename, in id)

5.45.1 Function Documentation

5.45.1.1 function calc_straincoefficient (in *basename,* in *id*)

5.46 CalcQfactor.m File Reference

Functions

- function CalcQfactor (in freqbase, in displ)
- function argmin (in array)

5.46.1 Function Documentation

5.46.1.1 function argmin (in *array*)

5.46.1.2 function CalcQfactor (in *freqbase,* in *displ*)

CalcQFactor calculates the Q factor from the displacement curve. [Qfactor] = CalcQfactor(freqs,displ) calculates the Q factor of the vibration from the frequency response. Q factor is defined as: (resonance frequency)/(bandwidth) bandwidth is the frequency range between which peak amplitude reduces by 3db, or to 0.707 of its value

5.47 CalculateDynamicStrainCoeff.m File Reference

Functions

- function CalculateDynamicStrainCoeff (in model, in geomtag, in cracktag, in physicstag, in objecttag, in skipassignment)

5.47.1 Function Documentation

5.47.1.1 function CalculateDynamicStrainCoeff (in *model,* in *geomtag,* in *cracktag,* in *physicstag,* in *objecttag,* in *skipassignment*)

Determine the magnitude of the strain across the specified crack in the center of its face (NOTE: Does not work if the crack center is outside the material)

*** VERY IMPORTANT!!! *** This will only give a meaningful strain if the specified physics has continuity boundary conditions across the crack face. That means, when creating the physics, you must NOT have set the crackdiscontinuity option

Parameters:

Parameters

model,:	COMSOL model node
geomtag,:	COMSOL tag for the main geometry node (usually 'Geom'
cracktag,:	Tag for the crack to investigate (usually 'crack'

physicstag,:	Tag for the physics solved for. Usually solidmech_harmonic or solidmech_harmonicper. assumes the solution to run is [physicstag '_solution'].
objecttag,:	Tag for the physics of the object we are finding the vibration of

5.48 ClearWrappedObjects.m File Reference

Functions

- function ClearWrappedObjects ()

5.48.1 Function Documentation

5.48.1.1 function ClearWrappedObjects ( )

5.49 ClearWrappedObjectStruct.m File Reference

Functions

- function ClearWrappedObjectStruct (in structure_to_clear)

5.49.1 Function Documentation

5.49.1.1 function ClearWrappedObjectStruct (in *structure_to_clear*)

5.50 CrackPointNormal.m File Reference

Functions

- function CrackPointNormal (in model, in geomtag, in cracktag)

5.50.1 Function Documentation

5.50.1.1 function CrackPointNormal (in *model,* in *geomtag,* in *cracktag*)

function [crackpos,cracknormal]=CrackPointNormal(model,geomtag,cracktag)

Determine the location of the crack and the vector normal to its face.

Prototype: CrackPointNormal(model,'Geom','crack');

Parameters:

Parameters

model,:	COMSOL model node

geomtag,:	COMSOL tag for the main geometry node (usually 'Geom'
cracktag,:	Tag for the crack to investigate (usually 'crack'

5.51 CrackStrain.m File Reference

Functions

- function CrackStrain (in model, in geomtag, in cracktag, in physicstag, in freqidx, in solutiontag)

5.51.1 Function Documentation

5.51.1.1 function CrackStrain (in *model,* in *geomtag,* in *cracktag,* in *physicstag,* in *freqidx,* in *solutiontag*)

function [normalstrain,shearstrain]=CrackStrain(model,cracktag,freqidx,solutiontag)

Determine the magnitude of the strain across the specified crack in the center of its face (NOTE: Does not work if the crack center is outside the material)

*** VERY IMPORTANT!!! *** This will only give a meaningful strain if the specified physics has continuity boundary conditions across the crack face. That means, when creating the physics, you must NOT have set the crackdiscontinuity option

Parameters:

Parameters

model,:	COMSOL model node
geomtag,:	COMSOL tag for the main geometry node (usually 'Geom'
cracktag,:	Tag for the crack to investigate (usually 'crack'
physicstag,:	Tag for the physics solved for. Usually solidmech_harmonic or solidmech_harmonicper
solutiontag,:	Optional. Tag for the solution to use (usually solidmech_harmonic_solution or solidmech_harmonicper_solution). Defaults to [physicstag '_solution']

5.52 CrackStress.m File Reference

Functions

- function CrackStress (in model, in geomtag, in cracktag, in physicstag, in freqidx, in solutiontag)

5.52.1 Function Documentation

5.52.1.1 function CrackStress (in *model,* in *geomtag,* in *cracktag,* in *physicstag,* in *freqidx,* in *solutiontag*)

function [normalstress,shearstress]=CrackStress(model,cracktag,freqidx,solutiontag)

Determine the magnitude of the engineering stress across the specified crack in the center of its face (NOTE: Does not work if the crack center is outside the material)

*** VERY IMPORTANT!!! *** This will only give a meaningful stress if the specified physics has continuity boundary conditions across the crack face. That means, when creating the physics, you must NOT have set the crackdiscontinuity option

Parameters:

Parameters

model,:	COMSOL model node
geomtag,:	COMSOL tag for the main geometry node (usually 'Geom'
cracktag,:	Tag for the crack to investigate (usually 'crack'
physicstag,:	Tag for the physics solved for. Usually solidmech_harmonic or solidmech_harmonicper
solutiontag,:	Optional. Tag for the solution to use (usually solidmech_harmonic_solution or solidmech_-harmonicper_solution). Defaults to [physicstag '_solution']

5.53 CrackStress_point.m File Reference

Functions

- function CrackStress_point (in model, in geomtag, in cracktag, in physicstag, in freqidx, in stress_point, in solutiontag)

5.53.1 Function Documentation

5.53.1.1 function CrackStress_point (in *model,* in *geomtag,* in *cracktag,* in *physicstag,* in *freqidx,* in *stress_point,* in *solutiontag*)

function [normalstress,shearstress]=CrackStress(model,cracktag,freqidx,solutiontag)

Determine the magnitude of the engineering stress across the specified crack in the center of its face (NOTE: Does not work if the crack center is outside the material)

*** VERY IMPORTANT!!! *** This will only give a meaningful stress if the specified physics has continuity boundary conditions across the crack face. That means, when creating the physics, you must NOT have set the crackdiscontinuity option

Parameters:

Parameters

model,:	COMSOL model node
geomtag,:	COMSOL tag for the main geometry node (usually 'Geom'
cracktag,:	Tag for the crack to investigate (usually 'crack'
physicstag,:	Tag for the physics solved for. Usually solidmech_harmonic or solidmech_harmonicper
solutiontag,:	Optional. Tag for the solution to use (usually solidmech_harmonic_solution or solidmech_-harmonicper_solution). Defaults to [physicstag '_solution']

5.54 Create1DPlot.m File Reference

Functions

- function Create1DPlot (in varargin)

5.54.1 Function Documentation

5.54.1.1 function Create1DPlot (in *varargin*)

CREATE1DPLOT Adds a Plot to a Plot Group [plotgrouptag,plottag,plotgroupnode,plotnode] = CREATE1DPLOT(model, grouptag, plottype, varargin) Creates Plot of Given Type

Required parameters:

'model' - (obj) comsol model object 'plotgrouptag' - (str) name tag of plot group to be created 'plotproperties' - (struct) plot group properties;

plotproperties are the property field/value pairs to be set for the 1D plot being generated Following are the allowed fields in this struct: data - name tag of the 1D dataset expr - expression to plot differential - whether or not to set differential, allowable options are:'on', 'off' evalmethod - method to evalue the expression with: allowed options: 'linpoint' to plot Static Solution 'harmonic' to plot Harmonic Perturbation 'lintotal' to plot Total Instantaneous Solution 'lintotalavg' to plot Average for Total Solution 'lintotalrms' to plot RMS for Total Solution 'lintotalpeak' to plot Peak Value for Total Solution

5.55 Create3DPlot.m File Reference

Functions

- function Create3DPlot (in varargin)

5.55.1 Function Documentation

5.55.1.1 function Create3DPlot (in *varargin*)

CREATE3DPLOT Adds a Plot to a Plot Group [tag] = CREATE3DPLOT(model, grouptag, plottype, varargin) Creates Plot of Given Type

5.56 Create3DPlotGroup.m File Reference

Functions

- function Create3DPlotGroup (in varargin)

5.56.1 Function Documentation

5.56.1.1 function Create3DPlotGroup (in *varargin*)

CREATE3DPLOTGROUP Creates a 3D Plot Group [grouptag] = CREATE3DPLOTGROUP(model, grouptag) Creates Node of Given Type

5.57 CreateBlankSolutions.m File Reference

Functions

- function CreateBlankSolutions (in M)

WARNING: Reruns SelectBoundaryConditionsForStudy() on each study!

5.57.1 Function Documentation

5.57.1.1 function CreateBlankSolutions (in *M*)

WARNING: Reruns SelectBoundaryConditionsForStudy() on each study!

To run studies with custom solution solvers, you must manually activate each solution, rather than just run the study.

(If you just run the study, it will auto-create a new solver, which isn't usually what you want)

This runs each study by seeking out its solver This ignores the 'getnormals' study

5.58 CreateCameraNoise.m File Reference

Functions

- function CreateCameraNoise (in M, in tag, in camera_netd)

5.58.1 Function Documentation

5.58.1.1 function CreateCameraNoise (in *M,* in *tag,* in *camera_netd*)

5.59 CreateContactor.m File Reference

Functions

- function CreateContactor (in M, in geom, in tag, in shape, in pos, in normalvec, in angle, in leng, in width, in thickness)

  NOTE: Does not specify material or meshing.

5.59.1 Function Documentation

5.59.1.1 function CreateContactor (in *M,* in *geom,* in *tag,* in *shape,* in *pos,* in *normalvec,* in *angle,* in *leng,* in *width,* in *thickness*)

NOTE: Does not specify material or meshing.

NOTE: For now normalvec must be numeric because the face identification algorithm operates numerically.

5.60 CreateCouplant.m File Reference

Functions

- function CreateCouplant (in M, in geom, in tag, in shape, in pos, in normalvec, in angle)

5.60.1 Function Documentation

5.60.1.1 function CreateCouplant (in *M,* in *geom,* in *tag,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.61 CreateCrack.m File Reference

Functions

- function CreateCrack (in M, in geom, in tag, in specimen, in centerpoint, in semimajoraxislen, in semiminoraxislen, in axismajordirection, in axisminordirection, in subradii, in vibration_physicstags, in heatingfile, in cracktype)

5.61.1 Function Documentation

5.61.1.1 function CreateCrack (in *M,* in *geom,* in *tag,* in *specimen,* in *centerpoint,* in *semimajoraxislen,* in *semiminoraxislen,* in *axismajordirection,* in *axisminordirection,* in *subradii,* in *vibration_physicstags,* in *heatingfile,* in *cracktype*)

CREATECRACK Creates a a crack at a given position [crack] = CREATECRACK(M, geom, tag, specimen, centerpoint, semimajoraxislen, semiminoraxislen, axismajordirection, axisminordirection, subradii, vibration_physicstags, heatingfile, cracktype)

Parameters:

M: Top level ModelWrapper geom: Top level geometry tag: Tag for crack's WorkPlane specimen: Object in which to place the crack centerpoint: Center point for the crack. Should be on or in the specimen semimajoraxislen: Half-crack length, along semi-major (surface) axis semiminoraxislen: Half-crack length, along semi-minor (depth) axis axismajordirection-: Direction of semi-major (surface) axis axisminordirection: Direction of semi-minor (depth) axis subradii: A vector of axismajor_endboundary values. The last element should match the value of semimajoraxislen. The crack boundary will be split into sub boundaries, with the real purpose of enforcing a sufficiently fine mesh to be able to resolve the spatial distribution of predicted heating. The shape depend on the crack type, annuli for penny shaped and rectangles for through cracks. vibration_physicstags: A cell array of tag names representing physics models used for vibration calculation. For each of these a variable [cracktag '*centerstress*' physicstag] will be created representing the local stress field at the centerpoint. Please note that for this to be meaningful that physics should be configured for continuity mechanical boundary conditions across the crack (otherwise stress isn't well defined for a discontinuity). heatingfile: Optional file with crack heating data. It should have four columns: Time, surface radius, side1 heating, side2 heating. Side 1 corresponds to the negative major axis direction; side2 corresponds to the positive major axis direction. If not provided then the heat generation boundary condition will not be created. cracktype: A string that is either 'through' or 'penny' closure: A 2 column matrix of (axismajor_endboundary, closurestate_MPa). Last entry of first column of closure should match the value of semimajoraxislen. weakformpdephysicstag (obsolete, removed) A parameter, may be blank [] that is passed to BuildBoundaryHeatSourceBCs to accelerate MATLAB heat source calculations through an intermediate weak form PDE physics calculation

Closure is measured in MPa. Positions should be in order, representing the outer radius of each semi-annular ring, with the corresponding closure stress representing the average closure stress over that semi-annular ring. QbExpressions: An optional parameter, used for or to generate the expressions used for the heat source intensity Qb. It can be:

- An sprintf-style string with up to one 'd'-type substitution, which gets the corresponding row number of the closure parameter, or
- A cell array of strings, with length matching closure – passed directly as the heating parameter
- Numeric values – passed directly as heating param.

- If QbExpressions is not provided, the vibrothermography crack heating model is used instead. Example parameter values: centerpoint=[.07,.0254/2,.012]; semimajoraxislen=.003; semiminoraxislen=.0028; axismajordirection=[0,1,0]; axisminordirection=[0,0,-1]; closure=[.001, -30 ; .002, 0 ; .003, 60]; QbExpressions='heatintensity%.2d'

5.62 CreateCrackHeatingModel.m File Reference

Functions

- function CreateCrackHeatingModel (in M, in tag)

5.62.1 Function Documentation

5.62.1.1 function CreateCrackHeatingModel (in *M,* in *tag*)

5.63 CreateCrackStrainVariable.m File Reference

Functions

- function CreateCrackStrainVariable (in M, in tag, in cracktag, in physicstag)

5.63.1 Function Documentation

5.63.1.1 function CreateCrackStrainVariable (in *M,* in *tag,* in *cracktag,* in *physicstag*)

5.64 CreateCutPlane.m File Reference

Functions

- function CreateCutPlane (in varargin)

5.64.1 Function Documentation

5.64.1.1 function CreateCutPlane (in *varargin*)

CREATECUTPLANE Creates a Cut Plane Using the Provided Parameters [cutplane] = CREATECUTPLANE(model, cutplane, tagname) Creates a Cut Plane

5.65 CreateCutPoint3D.m File Reference

Functions

- function CreateCutPoint3D (in varargin)

5.65.1 Function Documentation

5.65.1.1 function CreateCutPoint3D (in *varargin*)

CREATECUTPOINT3D Creates a Cut Plane Using the Provided Parameters [cutpointnode] = CREATECUTPOINT3-D(model, cutpoint) Creates a Cut Plane

5.66 CreateDCMaterialIfNeeded.m File Reference

Functions

- function CreateDCMaterialIfNeeded (in M, in geom, in materialname, in materialprefix)

5.66.1 Function Documentation

5.66.1.1 function CreateDCMaterialIfNeeded (in *M,* in *geom,* in *materialname,* in *materialprefix*)

function material=CreateDCMaterialIfNeeded(M,geom,materialname,materialprefix) Auto-names tag according to value of materialname Only creates material if it doesn't already exist. Otherwise silently returns existing material materialprefix is used to identify the material DC parameters from the DC database (AddParamToParamDB, etc.): e.g. [materialprefix 'YoungsModulus'] if the materialprefix is not given, the material name is used in its place (with spaces and dashes converted to underscores)

5.67 CreateDistributionsOnEdges.m File Reference

Functions

- function CreateDistributionsOnEdges (in M, in geom, in tagbase, in meshedobj, in boundaryselectionentities, in numelemfunc)

5.67.1 Function Documentation

5.67.1.1 function CreateDistributionsOnEdges (in *M,* in *geom,* in *tagbase,* in *meshedobj,* in *boundaryselectionentities,* in *numelemfunc*)

Create a Distribution on each edge surrounding and inside boundaryselection – which is a set of boundary entities. These Distributions are for meshedobj (caller is responsible for adding to the struct). numelemfunc is the function to call to determine the number of elements in a particular edge. It is called as numelem=numelemfunc(-M,geom,meshedobj,edgeid,edgelength)

5.68 CreateExcitationWindow.m File Reference

Functions

- function CreateExcitationWindow (in M, in excitationwindowtag, in t0, in t1, in t2, in t3)

5.68.1 Function Documentation

5.68.1.1 function CreateExcitationWindow (in *M,* in *excitationwindowtag,* in *t0,* in *t1,* in *t2,* in *t3*)

CreateExcitationWindow Creates The Excitation Window

This function defines a rectangular window with smooth edges to mark the beginning and end of excitation Smoothing is performed as raised cosine window defined according to the excitation string. GEN:BURST ARB <excfreq> Hz <t0> s <t1> s <t2> s <t3> s where excfreq is the excitation frequency, t0 is the start of excitation, t1 is the point at which maximum ampitude is reached, t2 is the point at which excitation starts to ramp down and t2 is when excitation stops.

5.69 CreateExperimentLog.m File Reference

Functions

- function CreateExperimentLog (in filename, in summarystruct)

5.69.1 Function Documentation

5.69.1.1 function CreateExperimentLog (in *filename,* in *summarystruct*)

CREATEEXPERIMENTLOG Create An Experiment Log CREATEEXPERIMENTLOG(filename,reldest,summarystruct) Creates An Experiment Log with Specified Filename Parameters: summarystruct - (struct) structure with all the (field-name,value) pairs to add to the summary node of experiment log

5.70 CreateExplicitSelection.m File Reference

Functions

- function CreateExplicitSelection (in varargin)

5.70.1 Function Documentation

5.70.1.1 function CreateExplicitSelection (in *varargin*)

5.71 CreateFunction.m File Reference

Functions

- function CreateFunction (in M, in tag, in type)

  CREATEFUNCTION Creates a Function.

5.71.1 Function Documentation

5.71.1.1 function CreateFunction (in *M,* in *tag,* in *type*)

CREATEFUNCTION Creates a Function.

type can be 'Interpolation', 'Analytic', 'Rectangle', etc

5.72 CreateGeometryNode.m File Reference

Functions

- function CreateGeometryNode (in M, in tag, in meshtag, in ndim)

5.72.1 Function Documentation

5.72.1.1 function CreateGeometryNode (in *M,* in *tag,* in *meshtag,* in *ndim*)

CREATEGEOMETRYNODE Creates a New Geometry and Mesh Node in a Given Model [geom,mesh] = CREATEGEOMETRYNODE(M,tag,meshtag,ndim) Creates a New mD Geometry with Provided Tag names

5.73 CreateImpulseExcitation.m File Reference

Functions

- function CreateImpulseExcitation (in M, in impulseexcitationtag, in impulseexcitation_t0, in impulseexcitation_width)

  CreateImpulseExcitation defines the Gaussian pulse for time domain simulations of impulse excitation.

5.73.1 Function Documentation

5.73.1.1 function CreateImpulseExcitation (in *M,* in *impulseexcitationtag,* in *impulseexcitation_t0,* in *impulseexcitation_width*)

CreateImpulseExcitation defines the Gaussian pulse for time domain simulations of impulse excitation.

This function defines a temporal Gaussian pulse with a specified center time and a specified width. The pulse has units of 1/s and integrates to 1.0

5.74 CreateIsolator.m File Reference

Functions

- function CreateIsolator (in M, in geom, in tag, in shape, in pos, in normalvec, in angle)

5.74.1 Function Documentation

5.74.1.1 function CreateIsolator (in *M,* in *geom,* in *tag,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.75 CreateLaser.m File Reference

Functions

- function CreateLaser (in M, in tag, in physicstag, in laserx, in lasery, in laserz, in laserdx, in laserdy, in laserdz)

5.75.1 Function Documentation

5.75.1.1 function CreateLaser (in *M,* in *tag,* in *physicstag,* in *laserx,* in *lasery,* in *laserz,* in *laserdx,* in *laserdy,* in *laserdz*)

5.76 CreateMeshSizeProperty.m File Reference

Functions

- function CreateMeshSizeProperty (in M, in geom, in mesh, in meshobj, in propname, in tag, in meshsizemin, in meshsize, in dimensionality, in entities)

5.76.1 Function Documentation

5.76.1.1 function CreateMeshSizeProperty (in *M,* in *geom,* in *mesh,* in *meshobj,* in *propname,* in *tag,* in *meshsizemin,* in *meshsize,* in *dimensionality,* in *entities*)

Add a mesh size property to meshobj. Parameters:

Parameters

M,:	the ModelWrapper containing our representation of the model
geom,:	the ModelWrapper containing our representation of the geometry
mesh,:	the ModelWrapper containing our representation of the top level mesh object
meshobj,:	The meshing object to get the new property. This MUST be the the object you want the size applied to (can't be an ancestor) as it is used to extract the parent.
propname,:	Name of new property (typically 'size')
tag,:	Tag – typically [tag '_size']
meshsizemin,:	minimum mesh element size, or [] to disable
meshsize,:	maximum mesh element size, or [] to disable
dimensionality,:	2 if parent is meshing a boundary, 3 if a domain
entities,:	boundary or domain entities.

5.77 CreateModel.m File Reference

Functions

- function CreateModel (in modeltag, in componenttag)

5.77.1 Function Documentation

5.77.1.1 function CreateModel (in *modeltag,* in *componenttag*)

CREATEMODEL Creates a New Model on Comsol Server with Given Model Tag Will overwrite existing model if the name is the same [model] = CREATEMODEL() Creates a New Model with Default Tag [model] = CREATEMODEL(modeltag,componenttag) Creates a New Model with Given Tag

5.78 CreateOrReplace.m File Reference

Functions

- function CreateOrReplace (in parent, in tag, in varargin)

5.78.1 Function Documentation

5.78.1.1 function CreateOrReplace (in *parent,* in *tag,* in *varargin*)

function node=CreateOrReplace(parent,tag,type,...) Remove any existing COMSOL child of parent with the specified tag. Then create a new COMSOL node with the specified tag and type

5.79 CreateParameter.m File Reference

Functions

- function CreateParameter (in varargin)

5.79.1 Function Documentation

5.79.1.1 function CreateParameter (in *varargin*)

5.80 CreatePhysics.m File Reference

Functions

- function CreatePhysics (in M, in geom, in tag, in type, in varargin)

5.80.1 Function Documentation

5.80.1.1 function CreatePhysics (in *M,* in *geom,* in *tag,* in *type,* in *varargin*)

Parameters

type	can be 'SolidMechanics' or other types of COMSOL-supported physics

5.81 CreateProbe.m File Reference

Functions

- function CreateProbe (in M, in tag, in physicstag, in coordx, in coordy, in coordz, in directionx, in directiony, in directionz, in varargin)

5.81.1 Function Documentation

5.81.1.1 function CreateProbe (in *M,* in *tag,* in *physicstag,* in *coordx,* in *coordy,* in *coordz,* in *directionx,* in *directiony,* in *directionz,* in *varargin*)

CREATEPROBE Create a probe point where displacement and velocity can be measured obj = CreateProbe(M, tag, physicstag, coordx,coordy,coordz,directionx,directiony,directionz,...) Creates displacement probe at a specified point that is sensitive to motion in a specified direction

Required parameters:

M - (obj) comsol model object tag - (str) name tag of domain probe to be created physics - (str) name tag of physics domain for probe to be created coordx, coordy,coordz - point coordinates (may be numbers or strings) directionx,directiony,directionz - sensitivity direction vector (will be normalized to a unit vector) (additional parameters) - Additional parameters are (set parameter name, set parameter value) pairs, e.g 'bndsnap3','on' will enable snap to boundary

Returns:

obj – wrappedobject

5.82 CreateRectangularBarSpecimen.m File Reference

Functions

- function CreateRectangularBarSpecimen (in M, in geom, in tag, in spclength, in spcwidth, in spcthickness, in spcmaterial)

5.82.1 Function Documentation

5.82.1.1 function CreateRectangularBarSpecimen (in *M,* in *geom,* in *tag,* in *spclength,* in *spcwidth,* in *spcthickness,* in *spcmaterial*)

CREATERECTANGULARBARSPECIMEN Creates a Rectangular Bar Specimen [specimen] = CREATERECTANGULARBARSPECIMEN(M, geom, tag, spclength, spcwidth, spcthickness) Creates Bar With given geometry

5.83 CreateSolution.m File Reference

Functions

- function CreateSolution (in M, in study, in tag, in varargin)

5.83.1 Function Documentation

5.83.1.1 function CreateSolution (in *M,* in *study,* in *tag,* in *varargin*)

Parameters

Type	can be 'Stationary', 'Eigenvalue', or 'Time' varargin is stepobj1,type1, stepobj2,type2, ...

5.84 CreateStudy.m File Reference

Functions

- function CreateStudy (in M, in geom, in tag)

5.84.1 Function Documentation

5.84.1.1 function CreateStudy (in *M,* in *geom,* in *tag*)

5.85 CreateThinContactor.m File Reference

Functions

- function CreateThinContactor (in M, in geom, in tag, in specimen, in shape, in pos, in normalvec, in angle, in leng, in width)

5.85.1 Function Documentation

5.85.1.1 function CreateThinContactor (in *M,* in *geom,* in *tag,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle,* in *leng,* in *width*)

5.86 CreateThinCouplant.m File Reference

Functions

- function CreateThinCouplant (in M, in geom, in tag, in specimen, in shape, in pos, in normalvec, in angle)

5.86.1 Function Documentation

5.86.1.1 function CreateThinCouplant (in *M,* in *geom,* in *tag,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.87 CreateThinIsolator.m File Reference

Functions

- function CreateThinIsolator (in M, in geom, in tag, in specimen, in shape, in pos, in normalvec, in angle)

5.87.1 Function Documentation

5.87.1.1 function CreateThinIsolator (in *M,* in *geom,* in *tag,* in *specimen,* in *shape,* in *pos,* in *normalvec,* in *angle*)

5.88 CreateTransducerDisplacementVariable.m File Reference

Functions

- function CreateTransducerDisplacementVariable (in M, in xducercalib, in amplitude)

  NOTE: tag hardwired to 'xducerdisplacement'.

5.88.1 Function Documentation

5.88.1.1 function CreateTransducerDisplacementVariable (in *M,* in *xducercalib,* in *amplitude*)

NOTE: tag hardwired to 'xducerdisplacement'.

5.89 CreateVariable.m File Reference

Functions

- function CreateVariable (in M, in tag, in value)

5.89.1 Function Documentation

5.89.1.1 function CreateVariable (in *M,* in *tag,* in *value*)

5.90 CreateVibroHarmonic.m File Reference

Functions

- function CreateVibroHarmonic (in M, in geom, in tag, in freqrange, in crackdiscontinuity, in use_impulse_force_-excitation)

5.90.1 Function Documentation

5.90.1.1 function CreateVibroHarmonic (in *M,* in *geom,* in *tag,* in *freqrange,* in *crackdiscontinuity,* in *use_impulse_force_excitation*)

Create physics, study, step, and solution for a harmonic (NON-perturbation) vibration analysis for Vibrothermography. No static analysis is required.

Parameters

Parameters

M,:	ModelWrapper around top level model

geom,:	Wrapped top level geometry
tag	: Tag for physics. Should usually be 'solidmech_harmonic'
freqrange,:	Single frequency or range of frequencies to use. can be 'range(freqstart,freqstep,freqend)'
crackdiscontinuity,-:	Optional boolean, default false. If false the crack face boundary conditions are continuity conditions. If true, the thin elastic layer BC is used.
use_impulse_-force_excitation,:	Optional boolean, default false. If false use the excitation boundary condition class, which enables the couplant model. If true, use the impulseforceexcitation class which is just a simple impulse force.

Return values

solidmech_harmonic	

5.91 CreateVibroHarmonicPer.m File Reference

Functions

- function CreateVibroHarmonicPer (in M, in geom, in solidmech_static, in tag, in freqrange, in crackdiscontinuity, in nonlinear)

5.91.1 Function Documentation

5.91.1.1 function CreateVibroHarmonicPer (in *M,* in *geom,* in *solidmech_static,* in *tag,* in *freqrange,* in *crackdiscontinuity,* in *nonlinear*)

Create physics, study, step, and solutin for a harmonic perturbation vibration analysis for Vibrothermography given a static analysis (solidmech_static)

Parameters

Parameters

M,:	ModelWrapper around top level model
geom,:	Wrapped top level geometry
solidmech_-static,:	Wrapped static physics analysis to perturb
tag,:	Tag for physics to create. Should usually be 'solidmech_harmonicper'
freqrange,:	Single frequency or range of frequencies to use. can be 'range(freqstart,freqstep,freqend)'
crackdiscontinuity,-:	Optional boolean, default false. If false the crack face boundary conditions are continuity conditions. If true, the thin elastic layer BC is used.
nonlinear,:	Optional boolean, default false. If false the problem is presumed to be linear. If true, geometric nonlineary is considered.

If optional nonlinear parameter is true, then enable geometric nonlinearity and nonlinear solver.

5.92 CreateVibroHeatConvert.m File Reference

Functions

- function CreateVibroHeatConvert (in M, in geom, in tag, in vibration_physics, in vibrationsolutionindex)

5.92.1 Function Documentation

5.92.1.1 function CreateVibroHeatConvert (in *M,* in *geom,* in *tag,* in *vibration_physics,* in *vibrationsolutionindex*)

This function creates a Weak Form Boundary PDE that converts the output Of the statistical crack heating model into a result value that doesn't Horribly slow down COMSOL. If we just refer to the crack heating model result Directly, the heatflow model is VERY slow to run.

Return values

heatconvert	

5.93 CreateVibroHeatFlow.m File Reference

Functions

- function CreateVibroHeatFlow (in M, in geom, in tag)

5.93.1 Function Documentation

5.93.1.1 function CreateVibroHeatFlow (in *M,* in *geom,* in *tag*)

Return values

heatflow	

5.94 CreateVibroModal.m File Reference

Functions

- function CreateVibroModal (in M, in geom, in tag, in crackdiscontinuity)

5.94.1 Function Documentation

5.94.1.1 function CreateVibroModal (in *M,* in *geom,* in *tag,* in *crackdiscontinuity*)

Create physics, study, step, and solution for a modal (NON-perturbation) vibration analysis for Vibrothermography. No static analysis is required.

Parameters

Parameters

M,:	ModelWrapper around top level model
geom,:	Wrapped top level geometry
tag	: Tag for physics. Should usually be 'solidmech_modal'
crackdiscontinuity,-:	Optional boolean, default false. If false the crack face boundary conditions are continuity conditions. If true, the thin elastic layer BC is used.

Return values

solidmech_modal	

5.95 CreateVibroMultiSweep.m File Reference

Functions

- function CreateVibroMultiSweep (in M, in geom, in tag, in seg1_freqrange, in seg2_freqrange, in seg3_freqrange, in seg4_freqrange)

5.95.1 Function Documentation

5.95.1.1 function CreateVibroMultiSweep (in *M,* in *geom,* in *tag,* in *seg1_freqrange,* in *seg2_freqrange,* in *seg3_freqrange,* in *seg4_freqrange*)

Create physics, study, step, and solution for a harmonic (NON-perturbation) vibration analysis Consisting of three subsweeps No static analysis is required.

Parameters

Parameters

M,:	ModelWrapper around top level model
geom,:	Wrapped top level geometry
tag	: Tag for physics. Should usually be 'solidmech_harmonic'
seg1_freqrange,:	Single frequency or range of frequencies to use. can be 'range(freqstart,freqstep,freqend)'
seg2_freqrange,:	Single frequency or range of frequencies to use. can be 'range(freqstart,freqstep,freqend)'
seg3_freqrange,:	Single frequency or range of frequencies to use. can be 'range(freqstart,freqstep,freqend)'
seg4_freqrange,:	Single frequency or range of frequencies to use. can be 'range(freqstart,freqstep,freqend)'

Uses solidmech_harmonic physicsclass boundary conditions The crack face boundary conditions are continuity conditions. This function uses the impulseforceexcitation class which is just a simple impulse force.

Return values

solidmech_multisweep	

5.96 CreateVibroStatic.m File Reference

Functions

- function CreateVibroStatic (in M, in geom, in tag, in crackdiscontinuity, in nonlinear)

5.96.1 Function Documentation

5.96.1.1 function CreateVibroStatic (in *M,* in *geom,* in *tag,* in *crackdiscontinuity,* in *nonlinear*)

Create a static distortion analysis for vibrothermography.

Parameters

Parameters

M,:	ModelWrapper around top level model
geom,:	Wrapped top level geometry
tag,:	Tag for physics. Should usually be 'solidmech_static'
crackdiscontinuity,:	Optional boolean, default false. If false the crack face boundary conditions are continuity conditions. If true, the thin elastic layer BC is used.
nonlinear,:	Optional boolean, default false. If false the problem is presumed to be linear. If true, geometric nonlineary is considered and a nonlinear solver is used

If optional nonlinear parameter is true, then enable geometric nonlinearity and nonlinear solver.

Return values

solidmech_static	

5.97 CreateVibroTimeDomain.m File Reference

Functions

- function CreateVibroTimeDomain (in M, in geom, in tag, in crackdiscontinuity)

5.97.1 Function Documentation

5.97.1.1 function CreateVibroTimeDomain (in *M,* in *geom,* in *tag,* in *crackdiscontinuity*)

Create physics, study, step, and solution for a time-domain vibration analysis for Vibrothermography. No static analysis is required.

Parameters

Parameters

M,:	ModelWrapper around top level model
geom,:	Wrapped top level geometry
tag	: Tag for physics. Should usually be 'solidmech_timedomain'
couplant_coord,:	Coordinates of transducer – used to correctly locate transducer contact point motion probe
crackdiscontinuity,-:	Optional boolean, default false. If false the crack face boundary conditions are continuity conditions. If true, the thin elastic layer BC is used.

Return values

solidmech_timedomain	

5.98 CreateWorkPlane.m File Reference

Functions

- function CreateWorkPlane (in varargin)

5.98.1 Function Documentation

5.98.1.1 function CreateWorkPlane (in *varargin*)

CREATEWORKPLANE Creates a Work Plane With Given Parameters [node] = CREATEWORKPLANE(model, geomtag, properties) Creates Work Plane With Given Properties

5.99 CreateWrappedModel.m File Reference

Functions

- function CreateWrappedModel (in M, in tag, in parent, in varargin)

5.99.1 Function Documentation

5.99.1.1 function CreateWrappedModel (in *M,* in *tag,* in *parent,* in *varargin*)

5.100 CreateWrappedProperty.m File Reference

Functions

- function CreateWrappedProperty (in M, in object, in propname, in tag, in parent, in varargin)

5.100.1 Function Documentation

5.100.1.1 function CreateWrappedProperty (in *M,* in *object,* in *propname,* in *tag,* in *parent,* in *varargin*)

Add a newly created wrapped model as a property to object The new ModelWrapper will be object.propname parent is the COMSOL object which will be used for creation (parent.create(tag,varargin{:})

5.101 crossproduct_cellstr_array.m File Reference

Functions

- function crossproduct_cellstr_array (in vec1, in vec2)

5.101.1 Function Documentation

5.101.1.1 function crossproduct_cellstr_array (in *vec1,* in *vec2*)

5.102 CurveFitLoadDeformation.m File Reference

Functions

- function CurveFitLoadDeformation (in filename, in degree, in thresholdload)

  CURVEFITLOADDEFORMATION fits a polynomial to the load deformation data found in the file filename.

5.102.1 Function Documentation

5.102.1.1 function CurveFitLoadDeformation (in *filename,* in *degree,* in *thresholdload*)

CURVEFITLOADDEFORMATION fits a polynomial to the load deformation data found in the file filename.

5.103 DataSetExistsForProbe.m File Reference

Functions

- function DataSetExistsForProbe (in model, in probe_name)

5.103.1 Function Documentation

5.103.1.1 function DataSetExistsForProbe (in *model,* in *probe_name*)

5.104 DataSetExistsForSolution.m File Reference

Functions

- function DataSetExistsForSolution (in model, in solution_name)

5.104.1 Function Documentation

5.104.1.1 function DataSetExistsForSolution (in *model,* in *solution_name*)

5.105 DebugBuildRemainingGeometry.m File Reference

Functions

- function DebugBuildRemainingGeometry (in M)

5.105.1 Function Documentation

5.105.1.1 function DebugBuildRemainingGeometry (in *M*)

5.106 DebugFlawFunc.m File Reference

Functions

- function DebugFlawFunc (in M, in GeometryFunc, in tag, in flawfunc)

5.106.1 Function Documentation

5.106.1.1 function DebugFlawFunc (in *M,* in *GeometryFunc,* in *tag,* in *flawfunc*)

To debug a flawfunc, set any extrinsic variables, Execute your script up to where the FlawFunc (bldcrack) is defined, then call: [M,geom,specimen,tag]=DebugFlawFunc(M,geometryfunc,tag,flawfunc) (flawfunc, as usual, is optional) Then you can copy and paste your code.

5.107 DebugGeometryFunc.m File Reference

Functions

- function DebugGeometryFunc (in M, in tag, in GeometryFunc)

5.107.1 Function Documentation

5.107.1.1 function DebugGeometryFunc (in *M,* in *tag,* in *GeometryFunc*)

To debug or interactively develop a geometryfunc, set any extrinsic variables, Execute your script up to where the GeometryFunc (bldgeom) is defined, then call: [M,geom,tag]=DebugGeometryFunc(M,'specimen',GeometryFunc); (GeometryFunc is optional) Then you can copy and paste your code. When done, you can try DebugTryFlawFunc() and/or DebugBuildRemainingGeometry() and/or DebugTryPhysics()

5.108 DebugPhysicsFunc.m File Reference

Functions

- function DebugPhysicsFunc (in M, in geometryfunc, in flawfunc)

5.108.1 Function Documentation

5.108.1.1 function DebugPhysicsFunc (in *M,* in *geometryfunc,* in *flawfunc*)

To debug a physicsfunc, set any extrinsic variables, then call: Execute your script up to where the PhysicsFunc (bldphysics) is defined, then call: [M,geom,specimen,flaw]=DebugPhysicsFunc(M,geometryfunc,flawfunc) (flawfunc, as usual, is optional) Then you can copy and paste your code.

5.109 DebugTryFlawFunc.m File Reference

Functions

- function DebugTryFlawFunc (in M, in flawfunc)

5.109.1 Function Documentation

5.109.1.1 function DebugTryFlawFunc (in *M,* in *flawfunc*)

function DebugTryFlawFunc(M,flawfunc) You can call this after manually creating geometry to see if your FlawFunc runs OK.

5.110 DebugTryPhysics.m File Reference

Functions

- function DebugTryPhysics (in M, in physicsstudyfunc)

5.110.1 Function Documentation

5.110.1.1 function DebugTryPhysics (in *M,* in *physicsstudyfunc*)

function DebugTryPhysicsFunc(M,physicsfunc) You can call this after manually creating geometry and running DebugBuildRemainingGeometry() to see if your boundary conditions get built OK

5.111 DerivedValueExistsForProbeExpr.m File Reference

Functions

- function DerivedValueExistsForProbeExpr (in model, in probe_name, in probeexpr_name)

5.111.1 Function Documentation

5.111.1.1 function DerivedValueExistsForProbeExpr (in *model,* in *probe_name,* in *probeexpr_name*)

5.112 DisableBoundaryConditionInStudyStep.m File Reference

Functions

- function DisableBoundaryConditionInStudyStep (in M, in study, in step, in physicstag, in bcobj)

  PRIVATE: Intended to be called only by SelectBoundaryConditionsForStudyStep()

5.112.1 Function Documentation

5.112.1.1 function DisableBoundaryConditionInStudyStep (in *M,* in *study,* in *step,* in *physicstag,* in *bcobj*)

PRIVATE: Intended to be called only by SelectBoundaryConditionsForStudyStep()

5.113 DomainMeasureVolume.m File Reference

Functions

- function DomainMeasureVolume (in M, in geom, in domainentities)

5.113.1 Function Documentation

5.113.1.1 function DomainMeasureVolume (in *M,* in *geom,* in *domainentities*)

5.114 DynamicStrainResults.m File Reference

Functions

- function DynamicStrainResults (in M_or_Model, in physicstag, in modeshapefilename)

5.114.1 Function Documentation

5.114.1.1 function DynamicStrainResults (in *M_or_Model,* in *physicstag,* in *modeshapefilename*)

5.115 example_physics.m File Reference

Functions

- function example_physics (in M, in geom, in specimen, in flaw)

5.115.1 Function Documentation

5.115.1.1 function example_physics (in *M,* in *geom,* in *specimen,* in *flaw*)

5.116 example_physics2.m File Reference

Functions

- function example_physics2 (in M, in geom, in specimen, in flaw)

5.116.1 Function Documentation

5.116.1.1 function example_physics2 (in *M,* in *geom,* in *specimen,* in *flaw*)

5.117 ExecuteRunLater.m File Reference

Functions

- function ExecuteRunLater (in M, in runlaterclass)

  Execute all the runlater objects of the specified class.

5.117.1 Function Documentation

5.117.1.1 function ExecuteRunLater (in *M,* in *runlaterclass*)

Execute all the runlater objects of the specified class.

5.118 ExportData.m File Reference

Functions

- function ExportData (in varargin)

5.118.1 Function Documentation

5.118.1.1 function ExportData (in *varargin*)

EXPORTDATA Exports Data [export] = EXPORTDATA(model, exporttag,exporttype,exportproperties) Exports data with the given properties

Required parameters:

model - (obj) comsol model object exporttag - (str) name tag of export object; exportype - (str) type of export, allowed values are: 'Data', 'Plot','Table','Mesh','Image1D' exportproperties - (struct) export properties

Export Properties

header - (string) whether or not to incude header, allowed values: 'on','off'

data - (str) name tag of data set to export plotgroup - (str) name tag of plotgroup to export table - (str) name tag of table to export

filename - (str) full path of file name to export data to expr - (cell array), list of expressions to export struct - (str) 'sectionwise' or 'spreadsheet', only for plot or data export fullprec - (string) whether or not to incude full precision, allowed values: 'on','off' transpose - (str) transpose data (available only if struct is spreadsheet), allowed: 'on', 'off'

differential - whether or not to set differential, allowable options are:'on', 'off'

evalmethod - method to evalue the expression with: allowed options: 'linpoint' to plot Static Solution 'harmonic' to plot Harmonic Perturbation 'lintotal' to plot Total Instantaneous Solution 'lintotalavg' to plot Average for Total Solution 'lintotalrms' to plot RMS for Total Solution 'lintotalpeak' to plot Peak Value for Total Solution

5.119 ExportPlot.m File Reference

Functions

- function ExportPlot (in varargin)

5.119.1 Function Documentation

5.119.1.1 function ExportPlot (in *varargin*)

EXPORTPLOT Exports Plot [export] = EXPORTPLOT(model, export) Exports data with the given properties export-properties is a structure with the following fields

5.120 ExportRectangularBarFrontFaceData.m File Reference

Functions

- function ExportRectangularBarFrontFaceData (in varargin)

5.120.1 Function Documentation

5.120.1.1 function ExportRectangularBarFrontFaceData (in *varargin*)

EXPORTRECTANGULARBARFRONTFACEDATA Exports Mode Shape Data for the Front Face of the Rectangular Bar Specimen [] = EXPORTRECTANGULARBARFRONTFACEDATA(model, exportfilename) Exports Data to File of Given Name

5.121 ExportRectangularBarVolumeData.m File Reference

Functions

- function ExportRectangularBarVolumeData (in varargin)

5.121.1 Function Documentation

5.121.1.1 function ExportRectangularBarVolumeData (in *varargin*)

EXPORTRECTANGULARBARFRONTFACEDATA Exports Mode Shape Data for the Front Face of the Rectangular Bar Specimen [] = EXPORTRECTANGULARBARFRONTFACEDATA(model, exportfilename) Exports Data to File of Given Name

5.122 FilterMatching.m File Reference

Functions

- function FilterMatching (in inputvec, in filterfcn)

5.122.1 Function Documentation

5.122.1.1 function FilterMatching (in *inputvec,* in *filterfcn*)

5.123 FindBuildLater.m File Reference

Functions

- function FindBuildLater (in M, in buildlaterclass, in includebuilt)

5.123.1 Function Documentation

5.123.1.1 function FindBuildLater (in *M,* in *buildlaterclass,* in *includebuilt*)

function [sortedobjects]=FindBuildLater(M,buildlaterclass) function [sortedobjects]=FindBuildLater(M,buildlaterclass,includebuilt)

Find and return a cell array of all of the BuildLater objects matching the specified class. Only those that have not been built are returned unless 'includebuilt' is passed as true.

Also sorts by index, so results are in a consistent order

AFTER YOU BUILD EACH OBJECT DON'T FORGET TO SET object.is_built!!!

5.124 FindClosestFreq.m File Reference

Functions

- function FindClosestFreq (in model, in geomtag, in physicstag, in targetfreq, in solutiontag)

5.124.1 Function Documentation

5.124.1.1 function FindClosestFreq (in *model,* in *geomtag,* in *physicstag,* in *targetfreq,* in *solutiontag*)

5.125 FindContactBoundaries.m File Reference

Functions

- function FindContactBoundaries (in M, in geom, in domain1entities, in domain2entities)

5.125.1 Function Documentation

5.125.1.1 function FindContactBoundaries (in *M,* in *geom,* in *domain1entities,* in *domain2entities*)

5.126 FindPairs.m File Reference

Functions

- function FindPairs (in M, in geom, in object, in faceselectionfunc)

5.126.1 Function Documentation

5.126.1.1 function FindPairs (in *M,* in *geom,* in *object,* in *faceselectionfunc*)

Find and return identify contact or identity pairs referring to the face (boundary) entities returned by faceselectionfunc(-M,geom,object) returned pairs are identified by their tag names

5.127 FindWrappedObject.m File Reference

Functions

- function FindWrappedObject (in M, in tagname)

5.127.1 Function Documentation

5.127.1.1 function FindWrappedObject (in *M,* in *tagname*)

5.128 FindWrappedObjectsWithNonNumericParam.m File Reference

Functions

- function FindWrappedObjectsWithNonNumericParam (in M, in paramname)

5.128.1 Function Documentation

5.128.1.1 function FindWrappedObjectsWithNonNumericParam (in *M,* in *paramname*)

5.129 GenerateFieldExpressions.m File Reference

Functions

- function GenerateFieldExpressions (in physicstag)

5.129.1 Function Documentation

5.129.1.1 function GenerateFieldExpressions (in *physicstag*)

Extract stress and strain fields from physics tag

Parameters:

phsyicstag - (str) tag name of physics interface to extract fields

Returns:

strainfieldexpr - (cell array) expressions for strain field stressfieldexpr - (cell array) expressions for stress field freqexpr - (str) expressions for excitation frequency

5.130 GetAutomaticSelectionEntities.m File Reference

Functions

- function GetAutomaticSelectionEntities (in M, in geom, in object, in scope)

  scope can be 'pnt', 'edg', 'bnd', or 'dom'

5.130.1 Function Documentation

5.130.1.1 function GetAutomaticSelectionEntities (in *M,* in *geom,* in *object,* in *scope*)

scope can be 'pnt', 'edg', 'bnd', or 'dom'

5.131 GetBlockFace.m File Reference

Functions

- function GetBlockFace (in M, in geom, in block, in outwardnormal)

5.131.1 Function Documentation

5.131.1.1 function GetBlockFace (in *M,* in *geom,* in *block,* in *outwardnormal*)

block must have been created with 'createselection' 'on'!!! returns a vector of boundary id numbers

5.132 GetBoundary.m File Reference

Functions

- function GetBoundary (in M, in geom, in object)

5.132.1 Function Documentation

5.132.1.1 function GetBoundary (in *M,* in *geom,* in *object*)

5.133 GetBoundaryDisplacement.m File Reference

Functions

- function GetBoundaryDisplacement (in model, in geomtag, in physicstag, in objecttag, in closestfreqidx)

5.133.1 Function Documentation

5.133.1.1 function GetBoundaryDisplacement (in *model,* in *geomtag,* in *physicstag,* in *objecttag,* in *closestfreqidx*)

Measure the boundary displacement magnitude at each boundary point of a geometric object, at solnum=closestfreqidx

5.134 GetCrackBoundaries.m File Reference

Functions

- function GetCrackBoundaries (in M, in geom, in crack)

5.134.1 Function Documentation

5.134.1.1 function GetCrackBoundaries (in *M,* in *geom,* in *crack*)

function sorted_boundaries=GetCrackBoundaries(M,geom,crack)

Return a list of the boundary entity numbers that make up the specified crack. This uses an algorithm that is not quite perfect but most likely problems are diagnosed by the assert() below.

The list of entity numbers is sorted from the center of the crack to the outside. Our algorithm for identifying the boundaries: For each boundary in 'Geom_crack_bnd' (automatically created selection of the crack object) measure the total length of adjacent edges. Sort the boundaries by total length and use them from shortest to longest, checking adjacency each time

(Might fail if only a corner of the crack intersects the specimen, but this would be a badly placed crack)

Will fail if crack intersects an internal boundary or similar

5.135 GetCylinderFace.m File Reference

Functions

- function GetCylinderFace (in M, in geom, in cyl, in outwardnormal)

5.135.1 Function Documentation

5.135.1.1 function GetCylinderFace (in *M,* in *geom,* in *cyl,* in *outwardnormal*)

cylinder must have been created with 'createselection' 'on'!!! outwardnormal must be numeric!!! returns a vector of boundary id numbers

5.136 GetDataSetForProbe.m File Reference

Functions

- function GetDataSetForProbe (in model, in probe_name)

5.136.1 Function Documentation

5.136.1.1 function GetDataSetForProbe (in *model,* in *probe_name*)

5.137 GetDataSetForSolution.m File Reference

Functions

- function GetDataSetForSolution (in model, in solution_name)

5.137.1 Function Documentation

5.137.1.1 function GetDataSetForSolution (in *model,* in *solution_name*)

5.138 GetDCParamExcitationValue.m File Reference

Functions

- function GetDCParamExcitationValue (in M, in varargin)

5.138.1 Function Documentation

5.138.1.1 function GetDCParamExcitationValue (in *M,* in *varargin*)

GETDCPARAMSTRINGVALUE Fetches a Datacollect Exciation Params Value [valuestruct] = GETDCPARAMEXCITATIONVALUE(field) Fetches DC Param

5.139 GetDCParamNumericValue.m File Reference

Functions

- function GetDCParamNumericValue (in M, in varargin)

5.139.1 Function Documentation

5.139.1.1 function GetDCParamNumericValue (in *M,* in *varargin*)

GETDCPARAMNUMERICVALUE Fetches a Datacollect Numeric Value [valuestruct] = GETDCPARAMNUMERICVALUE(M,field, [units]) Fetches DC Param in Specified Units

5.140 GetDCParamStringValue.m File Reference

Functions

- function GetDCParamStringValue (in M, in varargin)

5.140.1 Function Documentation

5.140.1.1 function GetDCParamStringValue (in *M,* in *varargin*)

GETDCPARAMSTRINGVALUE Fetches a Datacollect Numeric Value [valuestruct] = GETDCPARAMSTRINGVALUE(field) Fetches DC Param

5.141 GetDerivedValueForProbeExpr.m File Reference

Functions

- function GetDerivedValueForProbeExpr (in model, in probe_name, in probeexpr_name)

5.141.1 Function Documentation

5.141.1.1 function GetDerivedValueForProbeExpr (in *model,* in *probe_name,* in *probeexpr_name*)

5.142 GetDomain.m File Reference

Functions

- function GetDomain (in M, in geom, in object)

  This function extracts the domain entities of an object.

5.142.1 Function Documentation

5.142.1.1 function GetDomain (in *M,* in *geom,* in *object*)

This function extracts the domain entities of an object.

5.143 GetEdgesInDirec.m File Reference

Functions

- function GetEdgesInDirec (in M, in geom, in geomobj, in direc)

5.143.1 Function Documentation

5.143.1.1 function GetEdgesInDirec (in *M,* in *geom,* in *geomobj,* in *direc*)

block must have been created with 'createselection' 'on'!!! returns a vector of boundary id numbers

5.144 GetMeasurement.m File Reference

Functions

- function GetMeasurement (in varargin)

5.144.1 Function Documentation

5.144.1.1 function GetMeasurement (in *varargin*)

GETMEASUREMENT get the dimension of the selcted entity [measure] = GETMEASUREMENT(model, geomtag,selectionnumber) measures the selectiondimension

5.145 GetNamedSelectionEntities.m File Reference

Functions

- function GetNamedSelectionEntities (in M, in name)

5.145.1 Function Documentation

5.145.1.1 function GetNamedSelectionEntities (in *M,* in *name*)

5.146 GetNormal.m File Reference

Functions

- function GetNormal (in M, in geom, in pos)

5.146.1 Function Documentation

5.146.1.1 function GetNormal (in *M,* in *geom,* in *pos*)

Get the normal vector at the closest boundary node to pos Note that this doesn't necessarily guarantee which geometric object the normal comes from, so you might get the normal in the wrong direction!

THIS FUNCTION MAY ONLY BE CALLED ONCE GEOM IS MESHED AND getnormals STUDY COMPLETED

If you want a normal that is always flipped to point outward from a convex object, see GetOutwardNormal(M,Geom,pos)

5.147 GetNormalBoundaries.m File Reference

Functions

- function GetNormalBoundaries (in M, in geom, in boundaryentities, in centerpos, in outwardnormal)

5.147.1 Function Documentation

5.147.1.1 function GetNormalBoundaries (in *M,* in *geom,* in *boundaryentities,* in *centerpos,* in *outwardnormal*)

Identify and return those of the provided boundary entities which have an outward normal matching (99%) outwardnormal

5.148 GetOutwardNormal.m File Reference

Functions

- function GetOutwardNormal (in M, in geom, in pos, in refpos)

5.148.1 Function Documentation

5.148.1.1 function GetOutwardNormal (in *M,* in *geom,* in *pos,* in *refpos*)

Get the normal vector at the closest boundary node to pos, flipped to point away from refpos. To get the outward normal from a convex object, refpos should be a point inside (not on the surface of) the object. Note that this doesn't necessarily guarantee which geometric object the normal comes from, so you might still get the wrong normal if you aren't careful

THIS FUNCTION MAY ONLY BE CALLED ONCE GEOM IS MESHED AND getnormals STUDY COMPLETED

So far, this works only on numeric vectors pos and refpos

5.149 GetParallelEdges.m File Reference

Functions

- function GetParallelEdges (in M, in geom, in edgeentities, in direc)

5.149.1 Function Documentation

5.149.1.1 function GetParallelEdges (in *M,* in *geom,* in *edgeentities,* in *direc*)

Identify and return those of the provided edge entities which have an vector matching direc (99%)

5.150 GetSolutionParamVals.m File Reference

Functions

- function GetSolutionParamVals (in model, in solution_name)

5.150.1 Function Documentation

5.150.1.1 function GetSolutionParamVals (in *model,* in *solution_name*)

[solutionparamvals] = GetSolutionParamVals(model,solution_name) get the values of time or frequeny parameters at which simulation is run

parameters

model - model object solution_name - tag name of the desired solution

5.151 IfElse.m File Reference

Functions

- function IfElse (in condition, in if_fcn, in else_fcn)

5.151.1 Function Documentation

5.151.1.1 function IfElse (in *condition,* in *if_fcn,* in *else_fcn*)

5.152 ImportCadGeometry.m File Reference

Functions

- function ImportCadGeometry (in varargin)

5.152.1 Function Documentation

5.152.1.1 function ImportCadGeometry (in *varargin*)

ImportCadGeometry Imports a CAD file into comsol workspace. [importcadtag,importcadnode] = ImportCadGeometry(model, geomtag, importcadtag, cadfilename) imports a cad model from disk. As of now, parasolid file format '.x_t' seems to be the most compatible file format for CAD import. Importing a STL file format may result in bad elements and the geometry not being rendered correctly.

Parameters:

model - (obj) comsol model object geomtag - (str) tag name for geometry node importcadtag - (str) tag name for the import cad node cadfilename - (str) filename of cad file to import

5.153 InitializeVibroSimScript.m File Reference

Functions

- function InitializeVibroSimScript (in mphfile)

  InitializeVibroSimScript Initializes the Environment to Run VibroSim Models.

5.153.1 Function Documentation

5.153.1.1 function InitializeVibroSimScript (in *mphfile*)

InitializeVibroSimScript Initializes the Environment to Run VibroSim Models.

InitializeVibroSimScript() Initializes Everything

Parameters

mphfile	(optional) allows you to load an existing mph file, which should just have specimen geometry and possibly materials and meshing – no physics or studies. This way you can create geometry manually in advance rather than having to construct everything from Matlab.

Return values

[M,model]	

5.154 innerprod_cellstr_array.m File Reference

Functions

- function innerprod_cellstr_array (in vec1, in vec2)

5.154.1 Function Documentation

5.154.1.1 function innerprod_cellstr_array (in *vec1,* in *vec2*)

5.155 LaserDisplacement.m File Reference

Functions

- function LaserDisplacement (in model, in geomtag, in physicstag, in freqidx, in vibnum, in solutiontag)

5.155.1 Function Documentation

5.155.1.1 function LaserDisplacement (in *model,* in *geomtag,* in *physicstag,* in *freqidx,* in *vibnum,* in *solutiontag*)

function disp=LaserDisplacement(model,geomtag,physicstag,freqidx,vibnum,solutiontag)

Determine the displacement at a particular point, as would be measured using a laser vibrometer.

Assumes the following parameters have been set up – probably by ObtainDCParameter() Laser position: laserx lasery laserz Laser direction: laserdx laserdy laserdz

Parameters: model: COMSOL model node geomtag: COMSOL tag for the main geometry node (usually 'Geom' physicstag: Tag for the physics solved for. Usually solidmech_harmonic or solidmech_harmonicper solutiontag: Optional. Tag for the solution to use (usually solidmech_harmonic_solution or solidmech_harmonicper_solution). Defaults to [physicstag '_solution'] vibnum: Number of vibrometer in use

5.156 LoadPairDatabase.m File Reference

Functions

- function LoadPairDatabase (in M, in geom)

5.156.1 Function Documentation

5.156.1.1 function LoadPairDatabase (in *M,* in *geom*)

5.157 LogMsg.m File Reference

Functions

- function LogMsg (in varargin)

5.157.1 Function Documentation

5.157.1.1 function LogMsg (in *varargin*)

LOGMSG Writes a log message to the console given provided log level Log Levels: 1 - Errors/Always Displayed Messages 2 - Warnings 3 - Information 4 - Debugging [] = LOGMSG(message) Displays Log Message with Log Level 3 [] = LOGMSG(message, loglevel) Displays Log Message with Given Log Level

5.158 magnitude_cellstr_array.m File Reference

Functions

- function magnitude_cellstr_array (in array)

5.158.1 Function Documentation

5.158.1.1 function magnitude_cellstr_array (in *array*)

5.159 meeker_statmodel_040815_eval.m File Reference

Functions

- function meeker_statmodel_040815_eval (in freqvec, in centerstrainmagvec, in semimajoraxislenvec, in closurevec, in semimajoraxispos1vec, in semimajoraxispos2vec, in whvec, in percentilevec)

5.159.1 Function Documentation

5.159.1.1 function meeker_statmodel_040815_eval (in *freqvec,* in *centerstrainmagvec,* in *semimajoraxislenvec,* in *closurevec,* in *semimajoraxispos1vec,* in *semimajoraxispos2vec,* in *whvec,* in *percentilevec*)

5.160 meeker_statmodel_040815_test.m File Reference

5.161 MergeSelections.m File Reference

Functions

- function MergeSelections (in M, in geom, in object, in selection_getter_ca)

5.161.1 Function Documentation

5.161.1.1 function MergeSelections (in *M,* in *geom,* in *object,* in *selection_getter_ca*)

5.162 MeshRemainingObjects.m File Reference

Functions

- function MeshRemainingObjects (in M, in geom, in mesh)

5.162.1 Function Documentation

5.162.1.1 function MeshRemainingObjects (in *M,* in *geom,* in *mesh*)

Instantiate all unbuilt buildable mesh objects, in creation order given the global mesh object. Buildable mesh objects are found by searching the WrappedObjectDB for objects with a 'meshbuilder' property, and add them to the COMSOL tree.

meshbuilder called as meshbuilder(M,geom,mesh,object);

5.163 MeshSetBounds.m File Reference

Functions

- function MeshSetBounds (in M, in mesh, in sizemin, in sizemax)

5.163.1 Function Documentation

5.163.1.1 function MeshSetBounds (in *M,* in *mesh,* in *sizemin,* in *sizemax*)

Also set the min and max parameters of the automatic global mesh size object

5.164 ModelWrapper.m File Reference

Classes

- class ModelWrapper

5.165 mul_cellstr_array_scalar.m File Reference

Functions

- function mul_cellstr_array_scalar (in vec, in scalar)

5.165.1 Function Documentation

5.165.1.1 function mul_cellstr_array_scalar (in *vec,* in *scalar*)

5.166 mul_cellstrs.m File Reference

Functions

- function mul_cellstrs (in str1, in str2)

5.166.1 Function Documentation

5.166.1.1 function mul_cellstrs (in *str1,* in *str2*)

5.167 MultiplyScalarStrByNumericVec.m File Reference

Functions

- function MultiplyScalarStrByNumericVec (in str, in vec)

5.167.1 Function Documentation

5.167.1.1 function MultiplyScalarStrByNumericVec (in *str,* in *vec*)

function product = MultiplyScalarStrByNumericVec(str,vec) This function gives the cell string array result of multiplying a scalar value represented as a string str by each component of numeric vector vec.

5.168 negate_cellstr_array.m File Reference

Functions

- function negate_cellstr_array (in vec1)

5.168.1 Function Documentation

5.168.1.1 function negate_cellstr_array (in *vec1*)

5.169 normalize_cellstr_array.m File Reference

Functions

- function normalize_cellstr_array (in array)

5.169.1 Function Documentation

5.169.1.1 function normalize_cellstr_array (in *array*)

5.170 ObtainDCParameter.m File Reference

Functions

- function ObtainDCParameter (in M, in name, in units)

5.170.1 Function Documentation

5.170.1.1 function ObtainDCParameter (in *M,* in *name,* in *units*)

Obtains the named COMSOL parameter corresponding to the named DC parameter. (The names are always the same, but this function makes sure that the parameter has a COMSOL definitition and is initialized to the current dc_param value)

5.171 PlotResonanceCurve.m File Reference

Functions

- function PlotResonanceCurve (in varargin)

5.171.1 Function Documentation

5.171.1.1 function PlotResonanceCurve (in *varargin*)

PlotResonanceCurve plots the frequency vs displacement [freqs,totaldispl,w] = PlotResonanceCurve(model,physicstag,varargin) extracts data from a single point and plots the resonance curve.

Required parameters:

model - (obj) comsol model object physicstag - (str) name tag of physics node whose data is extracted saveplot - (boolean) flag set whether or not to save image saveplotfilename - (str) name of the png file to save

Required keyword Arguments:

dataset - (str) name tag of dataset to be extracted

Optional keyword arguments:

coord - (string) coordinates evalmethod - (str) evaluation method differential - (str) whether or not to turn on differential

Returns:

freqs - list of frequencies totaldispl - total displacement values w - displacement, z component

5.172 ReferenceNamedMaterial.m File Reference

Functions

- function ReferenceNamedMaterial (in M, in geom, in obj, in materialtag, in domainselectionfunc)

5.172.1 Function Documentation

5.172.1.1 function ReferenceNamedMaterial (in *M,* in *geom,* in *obj,* in *materialtag,* in *domainselectionfunc*)

5.173 RunAllStudies.m File Reference

Functions

- function RunAllStudies (in model, in studyfilter)

5.173.1 Function Documentation

5.173.1.1 function RunAllStudies (in *model,* in *studyfilter*)

To run studies with custom solution solvers, you must manually activate each solution, rather than just run the study.

(If you just run the study, it will auto-create a new solver, which isn't usually what you want)

This runs each study by seeking out its solver This ignores the 'getnormals' study

studyfilter is a cell array of study names to ignore.

5.174 RunLater.m File Reference

Functions

- function RunLater (in M, in tag, in runlaterclasses, in runfcn)

5.174.1 Function Documentation

5.174.1.1 function RunLater (in *M,* in *tag,* in *runlaterclasses,* in *runfcn*)

RunLater is essentially a renamed wrapper around BuildLater, intended for actions that do not necessarily involve 'build'ing.

Note that runlaterclass and buildlaterclass names must not conflict

runfcn will be called as runfcn(M,rlobj)

5.175 SearchVariable.m File Reference

Functions

- function SearchVariable (in model, in varname)

5.175.1 Function Documentation

5.175.1.1 function SearchVariable (in *model,* in *varname*)

5.176 SelectBoundaryConditionsForStudy.m File Reference

Functions

- function SelectBoundaryConditionsForStudy (in M, in study, in classnameorcellarray)

5.176.1 Function Documentation

5.176.1.1 function SelectBoundaryConditionsForStudy (in *M,* in *study,* in *classnameorcellarray*)

function SelectBoundaryConditionsForStudy(M,study,classnameorcellarray) Select the boundary conditions for a particular study WARNING: May not be executed until ALL physicses have been created. Usually passed to RunLater() as RunLater(M,'rl_thisstudy_selectbcs','select_boundaryconditions',@(M,rlobj) SelectBoundaryConditionsForStudy(M,study,classnameorcellarray)) where 'rl_thisstudy_selectbcs' is a unique tag name Note that this also resets which physicses are turned on in the solver according to the settings in each study step.

5.177 SelectBoundaryConditionsForStudyStep.m File Reference

Functions

- function SelectBoundaryConditionsForStudyStep (in M, in study, in step, in classnameorcellarray)

5.177.1 Function Documentation

5.177.1.1 function SelectBoundaryConditionsForStudyStep (in *M,* in *study,* in *step,* in *classnameorcellarray*)

Select the boundary conditions for a particular study step WARNING: May not be executed until ALL physicses have been created. Usually passed to RunLater() as RunLater(M,'rl_thisstudystep_selectbcs','select_boundaryconditions',@(M,rlobj) SelectBoundaryConditionsForStudyStep(M,study,step,classnameorcellarray)) where 'rl_thisstudystep_selectbcs' is a unique tag name Note that this also resets which physicses are turned on in the solver according to the settings in each study step.

5.178 SetGeometryFinalization.m File Reference

Functions

- function SetGeometryFinalization (in M, in geom, in action, in createpairs, in pairtype)

5.178.1 Function Documentation

5.178.1.1 function SetGeometryFinalization (in *M,* in *geom,* in *action,* in *createpairs,* in *pairtype*)

Set_Geometry_Finalization selects the geometry finalization action: 'union' vs 'assembly'. You can also optionally select whether, for an assembly, to create pairs and whether the pairs should be 'identity' pairs or 'contact' pairs Also runs the geometry

5.179 SetMaterialProperty.m File Reference

Functions

- function SetMaterialProperty (in M, in material, in matprop, in value_or_expression)

5.179.1 Function Documentation

5.179.1.1 function SetMaterialProperty (in *M,* in *material,* in *matprop,* in *value_or_expression*)

5.180 SolidMechanics_SetInitialDisplacement.m File Reference

Functions

- function SolidMechanics_SetInitialDisplacement (in M, in physics, in initdisplvals)

5.180.1 Function Documentation

5.180.1.1 function SolidMechanics_SetInitialDisplacement (in *M,* in *physics,* in *initdisplvals*)

5.181 SolidMechanics_SetInitialStrain.m File Reference

Functions

- function SolidMechanics_SetInitialStrain (in M, in physics, in initstrainvals)

5.181.1 Function Documentation

5.181.1.1 function SolidMechanics_SetInitialStrain (in *M,* in *physics,* in *initstrainvals*)

5.182 straincoefficient_params.m File Reference

Functions

- function straincoefficient_params (in M, in isolators_and_couplant, in using_datacollect)

5.182.1 Function Documentation

5.182.1.1 function straincoefficient_params (in *M,* in *isolators_and_couplant,* in *using_datacollect*)

You can define parameters using AddParamToParamdb. As an alternative you can define COMSOL parameters with CreateParameter() As another alternative you can store parameters in the M structure (fields can be added to M, with addprop()).

5.183 string_in_cellstr_array.m File Reference

Functions

- function string_in_cellstr_array (in strng, in cellstrarray)

5.183.1 Function Documentation

5.183.1.1 function string_in_cellstr_array (in *strng,* in *cellstrarray*)

5.184 StringMatrix2Numeric.m File Reference

Functions

- function StringMatrix2Numeric (in strmat)

  Convert a matrix of java strings to a regular matlab matrix.

5.184.1 Function Documentation

5.184.1.1 function StringMatrix2Numeric (in *strmat*)

Convert a matrix of java strings to a regular matlab matrix.

5.185 StudyAddFrequencyStep.m File Reference

Functions

- function StudyAddFrequencyStep (in M, in geom, in study, in tag, in freqrange, in varargin)

  Add a frequency domain step to the specified study, with the specified physicses enabled and active.

5.185.1 Function Documentation

5.185.1.1 function StudyAddFrequencyStep (in *M,* in *geom,* in *study,* in *tag,* in *freqrange,* in *varargin*)

Add a frequency domain step to the specified study, with the specified physicses enabled and active.

5.186 StudyAddStep.m File Reference

Functions

- function StudyAddStep (in M, in geom, in study, in tag, in type, in varargin)

5.186.1 Function Documentation

5.186.1.1 function StudyAddStep (in *M,* in *geom,* in *study,* in *tag,* in *type,* in *varargin*)

function step = StudyAddStep(M,geom,study,tag,type,physics1, physics2, ...) Add a step of the specified type to the given study. Also enable the specified physics objects within that study. type could be 'Frequency' or other COMSOL-supported study type

5.187 StudyStepEnablePhysicsInSolvers.m File Reference

Functions

- function StudyStepEnablePhysicsInSolvers (in M, in step, in physics, in enabled)

5.187.1 Function Documentation

5.187.1.1 function StudyStepEnablePhysicsInSolvers (in *M,* in *step,* in *physics,* in *enabled*)

5.188 sub_cellstr_array.m File Reference

Functions

- function sub_cellstr_array (in vec1, in vec2)

5.188.1 Function Documentation

5.188.1.1 function sub_cellstr_array (in *vec1,* in *vec2*)

5.189 sub_cellstrs.m File Reference

Functions

- function sub_cellstrs (in str1, in str2)

5.189.1 Function Documentation

5.189.1.1 function sub_cellstrs (in *str1,* in *str2*)

5.190 to_cellstr_array.m File Reference

Functions

- function to_cellstr_array (in vec, in units)

5.190.1 Function Documentation

5.190.1.1 function to_cellstr_array (in *vec,* in *units*)

function ca_vec=to_cellstr_array(vec,units) Convert provided vector, which may be a cell string array already, a numeric vector, etc. into a cell array of strings. Units is an optional parameter that gets added inside [] to each component if vec is numeric

5.191 to_numeric_vector.m File Reference

Functions

- function to_numeric_vector (in M, in vec, in unit)

5.191.1 Function Documentation

5.191.1.1 function to_numeric_vector (in *M,* in *vec,* in *unit*)

function numvec = to_numeric_vector(M,vec,unit) Convert provided vector vec, which may already be numeric, or may be a cell string array, into a numeric vector. unit is an optional parameter (should NOT have brackets) that is passed to mphevaluate if given.

This calls mphevaluate to determine values. Please note that this fixes parameters, etc. and depending on how complete the model is and what has been instantiated, some values may not be available.

returns a row-vector

5.192 to_string.m File Reference

Functions

- function to_string (in inp, in default_units)

5.192.1 Function Documentation

5.192.1.1 function to_string (in *inp,* in *default_units*)

function str=to_string(inp, default_units) Convert provided input, which could be a number or a string representation of a number into a string representation of a number. If the input is a number attach the default_units (optional parameter), if given

5.193 UTCFinalReport_script.m File Reference

5.194 VibroPhysics.m File Reference

Functions

- function VibroPhysics (in M, in geom, in specimen, in flaw_notused, in mode, in use_impulse_force_excitation)

5.194.1 Function Documentation

5.194.1.1 function VibroPhysics (in *M,* in *geom,* in *specimen,* in *flaw_notused,* in *mode,* in *use_impulse_force_excitation*)

Parameters

mode,:	'modal', 'static', 'harmonicsweep', 'harmonicburst', 'heatflow', 'broadbandprocess', 'welderprocess', 'timedomain', or 'all'

5.195 VibroResults.m File Reference

Functions

- function VibroResults (in M_or_Model, in output_base_filename)

5.195.1 Function Documentation

5.195.1.1 function VibroResults (in *M_or_Model,* in *output_base_filename*)

5.196 VibroSim_default_params.m File Reference

Functions

- function VibroSim_default_params (in M)

5.196.1 Function Documentation

5.196.1.1 function VibroSim_default_params (in *M*)

You can define parameters using AddParamToParamdb. As an alternative you can define COMSOL parameters with CreateParameter() As another alternative you can store parameters in the M structure (fields can be added to M, with addprop()).

5.197 vibrosim_demo.m File Reference

5.198 vibrosim_demo2.m File Reference

5.199 vibrosim_demo3.m File Reference

5.200 vibrosim_test.m File Reference

5.201 WrapComsolNode.m File Reference

Functions

- function WrapComsolNode (in M, in node, in parent)

5.201.1 Function Documentation

5.201.1.1 function WrapComsolNode (in *M,* in *node,* in *parent*)

5.202 WrappedObjectDebug.m File Reference

Functions

- function WrappedObjectDebug (in debug)

5.202.1 Function Documentation

5.202.1.1 function WrappedObjectDebug (in *debug*)

5.203 WrappedObjectExists.m File Reference

Functions

- function WrappedObjectExists (in M, in tag)

  Determine whether a tagged object of the specified tag exists.

5.203.1 Function Documentation

5.203.1.1 function WrappedObjectExists (in *M,* in *tag*)

Determine whether a tagged object of the specified tag exists.

Index