

Structural Analysis: Going Beyond Standard Load Cases

Ivar KJELBERG

COMSOL Multiphysics Conference 2010,
Paris, 2010 November 18

Who might benefit from using COMSOL Multiphysics ?

- **High school students, and their teachers**
 - to **present**, to **explain** and to better **understand** Physics
- **University and technical students, as well as their Professors**
 - for the same reasons
- **Ph.D. Students and Researchers**
 - to **illustrate**, to **justify** and to **explain** their work
- **Engineers and Developers**
 - to **verify** and to **design quicker** new products



And where do we mainly find COMSOL Multiphysics today ?

Use of COMSOL Multiphysics

- The different category of users have very different needs and demands to COMSOL Multiphysics:
 - To **present**, to **explain** and to **understand** Physics
 - To **illustrate**, to **justify** and to **explain** a Research activity
 - To **verify** and to **design quicker** new Products
- To best serve the *last category* of users is to improve the speed to build and to verify models; and specific training for experienced traditional FEM users.
 - It is easy to get wrong results, which leads to frustrations, particularly to those used to older tools, with now a new subtle, but different, methodology to use.
 - However this complexity is required to couple all the different multiphysics
- I would state that >99% of the “bugs” (wrong results) people refer to are user errors and not “bugs” due to COMSOL Multiphysics, and still, too many errors passes undetected.


COMSOL Multiphysics versus “older” FEM Tools

The (simplified) three step of a classical FEM tool approach:

- **Import a well behaved geometry**
 - **CAD** model
- **Define the mesh its type, and ...**
 - its related **material properties**, and
 - **boundary conditions** on the mesh nodes
- **Solve and post-process**
 - **Document** the results

The COMSOL Multiphysics way

COMSOL Multiphysics has some subtle differences in its approach

- Define all the Physics
- Define your geometry => **OBJECTS**
 - Analyze your geometry via *geomanalyze(...)*, or Finish  and by this create the FEM geometrical => **ENTITIES**
 - Apply material properties onto the **Entities**
 - Apply Physics and Boundary Conditions onto the **Entities**
 - Define shape functions (meshing polynomial) onto the **Entities**
- Define the mesh => onto the **Entities**
- Solve and post-process

COMSOL Multiphysics' *Objects* and *Entities* types

- **CAD Geometric *OBJECTS***

- Points **0D**
- Lines **1D**
- Surfaces / Faces **2D**
- Volumes / Bodies / Parts **3D**

- Rectangle, circle ... **2D**
- Block, Sphere ... **3D**

- ***Composite Objects*** ***CAD assembly of parts***

- **FEM geometrical *ENTITIES***

- Points
- Edges
- **Boundaries** **2nd highest level**
- **Domains** **Highest level**

COMSOL Multiphysics' *Objects* and *Entities* properties

- **Geometric *OBJECTS* (CAD)**
 - Full freedom, objects might overlap, duplicate, or be isolated
- **Geometric *ENTITIES* (FEM) once created are**
 - Unique within their type (***Domains, Boundaries, Edges Points***)
 - Ordered and numbered (identified)
 - Are linked by a clear hierarchy: 3D → 2D → 1D → 0D
- **Implications of the Entity creation**
 - Orphan objects not entering the rules are ignored
 - If required, overlapping objects are split to achieve the correct entities

Entity generation obey two rules: *Union* and *Assembly*


- ***Union***
 - intersecting (*Composite* or not) *Objects* are split
 - overlapping *Objects* are merged
 - common Surfaces/Lines/Points are made unique → *Internal Boundaries*
- ***Assembly*** *nothing to do with a classical CAD assembly!*
 - *Composite Objects* and their **internal content** are treated as a ***Union***
 - overlapping **highest order** *Composite Objects* are **NOT** merged
 - adjacent *Boundaries* of *Composite Objects* are duplicated into two overlapping *Internal Boundaries*
 - on demand ***imprints*** (projections) of overlapping *Boundaries* might be made






Usage of the FEM *Entities*

- **Union**
 - *Internal Boundaries* (unique) are respecting *Flux Continuity* **by default**
- **Assembly**
 - *Identity* or *Contact Pairs* must be created to link overlapping boundaries
 - User specific Physics might also link onto overlapping boundaries
- COMSOL Multiphysics defines **all** *Physics*, *Shape functions*, *Materials* properties and *Boundary Conditions* onto its *Entities*
- The **meshing** is also defined on the *Domain Entities* and the mesh elements and nodes **inherit** their properties from the domain or boundaries they are linked to
- All is made for a maximum of flexibility of model generation and **changes**


Example of *Objects*

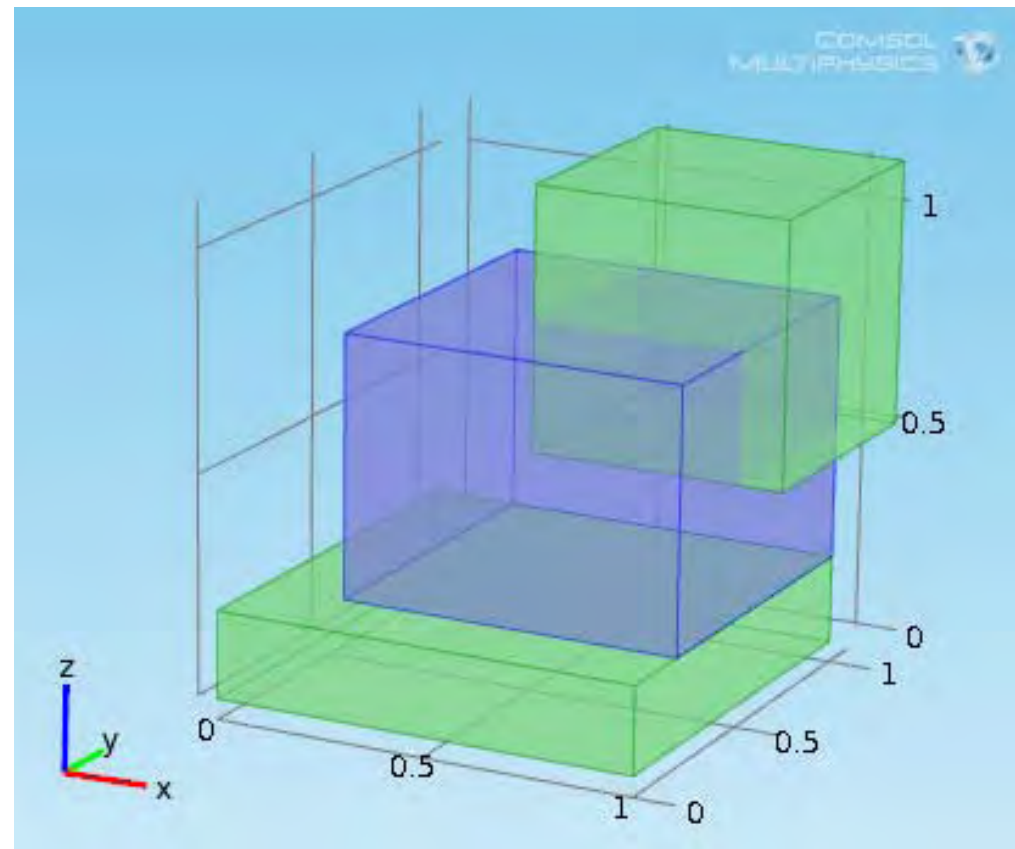
- Three 3D block *Objects* where the two green ones are grouped into a *Composite Object*

 Geometry 1

-  Block 1 (*blk1*)
-  Block 2 (*blk2*)
-  Block 3 (*blk3*)
-  Compose 1 (*co1*)
-  Form Union (*fin*)

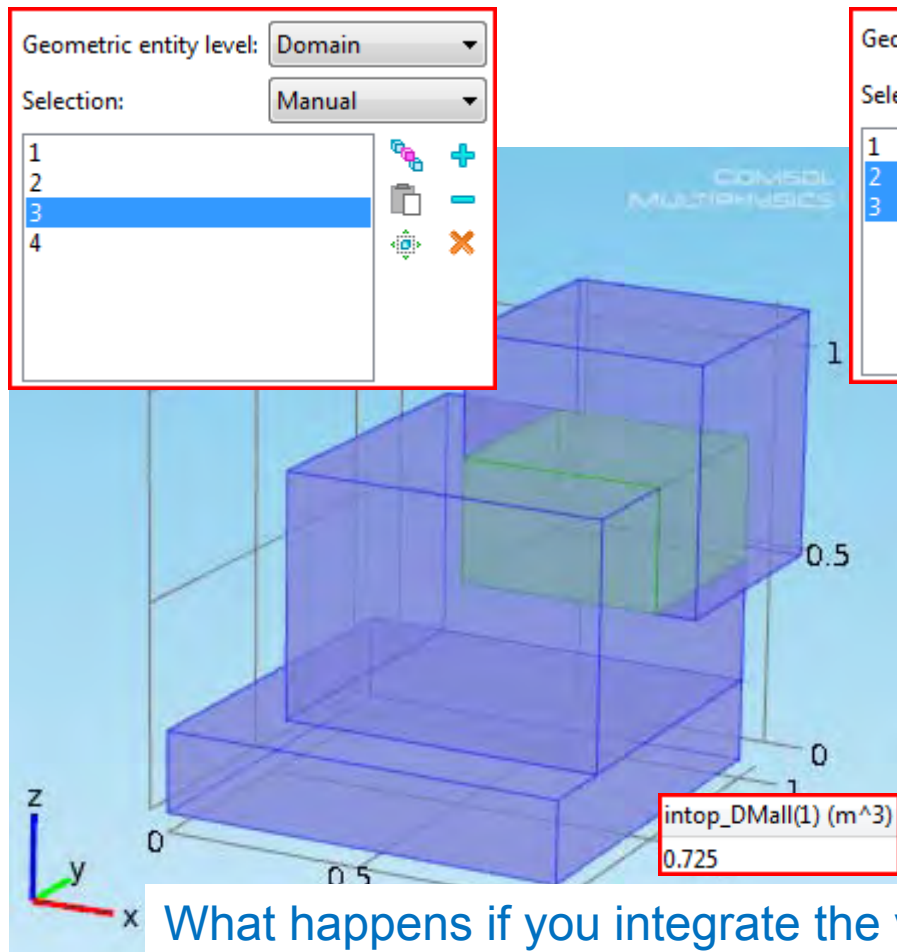
Geometric entity level: Object

co1	+
blk2	-
	×
	

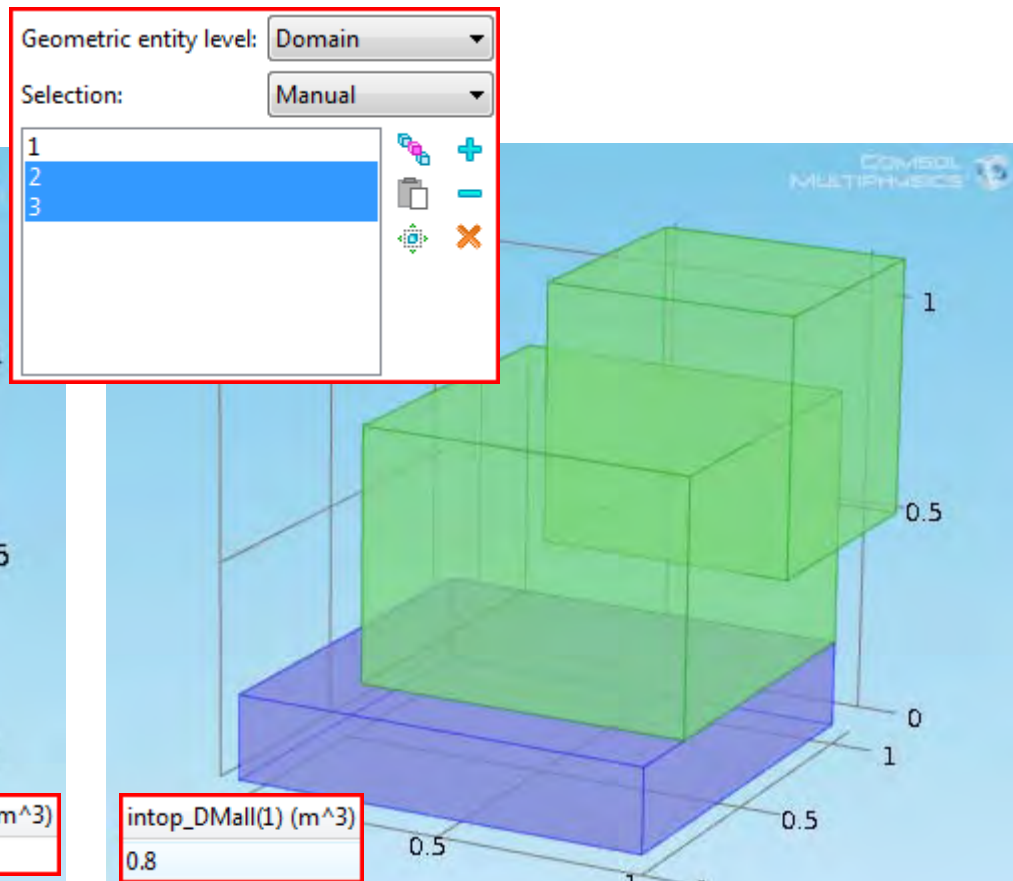


Example of *Entity* creation via *Union* and *Assembly*

- **Union:** creation of 4 *Domains*



- **Assembly:** creation of 3 *Domains*



What happens if you integrate the volume of all Domains in these two cases ?

Understand the different *Frames*

- The latest v4.1 documentation has a good explanation about the different frames used by COMSOL Multiphysics, with their related coordinate system used to express all physics and the results:
 - ***Spatial Frame*** Euler representation (st, ale) x, y, z, r, phi
 - ***Material Frame*** Lagrange representation X, Y, Z, R, PHI
 - ***Mesh Frame*** *Deformed Geometry* (dg) Xm, Ym, Zm, Rm, PHIm
- *Frames* are created “on need” as well as their coordinate naming
- *Structural* (st) create by default a *Spatial frame* driven by the deformations, ***new!***
- *Definitions-node operators* act on a specific, selected *Frame*, while post-processing operators act on the *Frame* defined in the *Results - Data Set - Solution* node

What are these funny units, & why didn't they tell us ?

- Let us take a *Structural (st) Body Load* example

$$Fb = m G = \rho V G$$

- Newton's law is saying:

$$Fb [N] = m [kg] * G [m/s^2]$$

$$Fb := \sum_{i=1}^n \int_0^{X_i} \int_0^{Y_i} \int_0^{Z_i} \rho_i G dx dy dz$$

- A *Load* is defined at our macroscopic object level as a force **Fb** acting on the mass via the acceleration.
- But FEM deals with small “finite” volumes (mesh elements) that adds up to form the domains.
- COMSOL Multiphysics wants the BC's to be defined at the FEM element level, not in our macroscopic world level.

Body Load

Domains

Selection: Selection 1

1	$\sum_{i=1}^n$	+
2		
3		

Equation

Coordinate System Selection

Coordinate system: Global coordinate system

Force

Load type: Load defined as force per unit volume

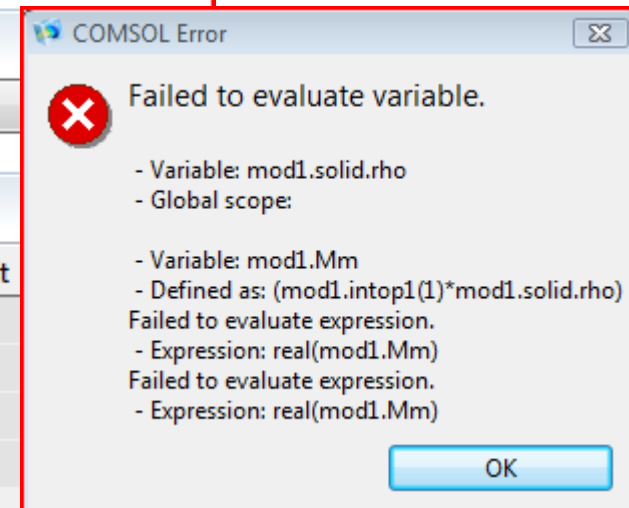
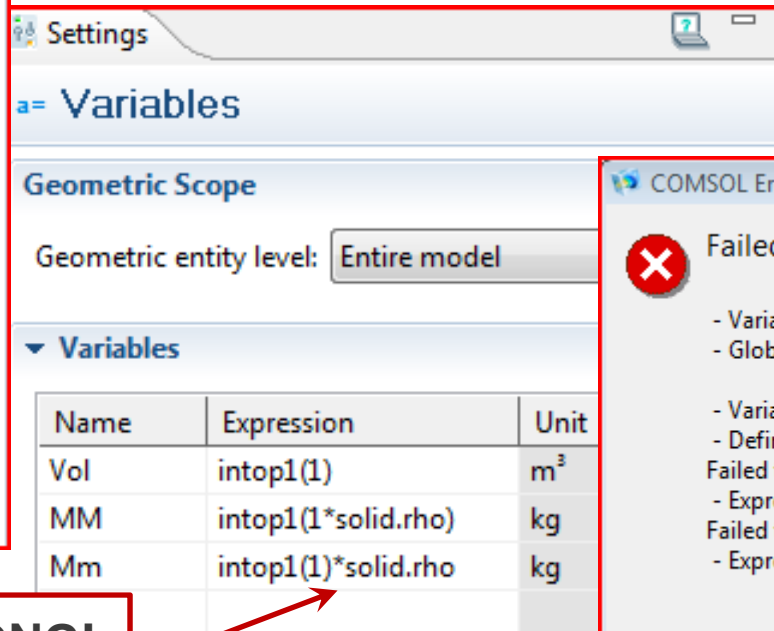
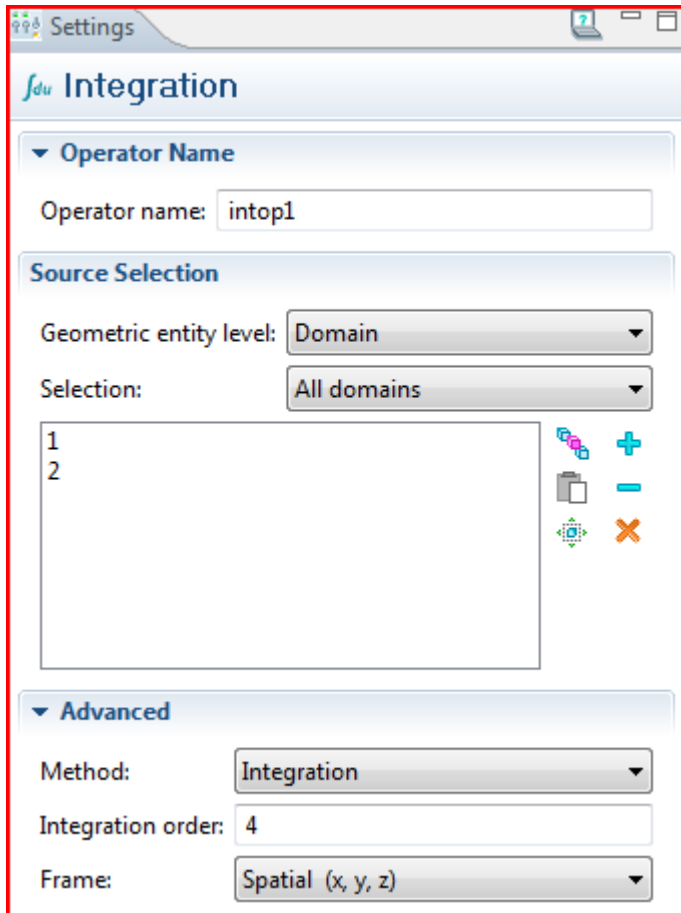
User defined		
0	x	
0	y	N/m ³
-g_const*solid.rho		

$dx dy dz$

Example of implicit sum and operators (on *Frames*)

Two *Domains* with different materials:

- The Volume **V** and total mass **MM** are OK
- The last variable **Mm** triggers an error: **solid.rho** being outside of the operator it is not uniquely defined, the implicit “_i” is back



WRONG!

3D, 2D, ... models

- **COMSOL Multiphysics always calculates in 3D**
- A 2D model is a 3D model where the depth “**z**” (the out-of-paper direction) has a default thickness (**solid.d**) of 1 [m]
some physics consider other default values, be aware!
- The physics equations of such a model simplifies accordingly
- For 2D-axi the thickness is replaced by the “loop length” $2*\pi*r$ [m]
there is a normalization change between v3.5a and v4, be aware!
- Note: for *Boundary Condition* definitions, this thickness must often be used

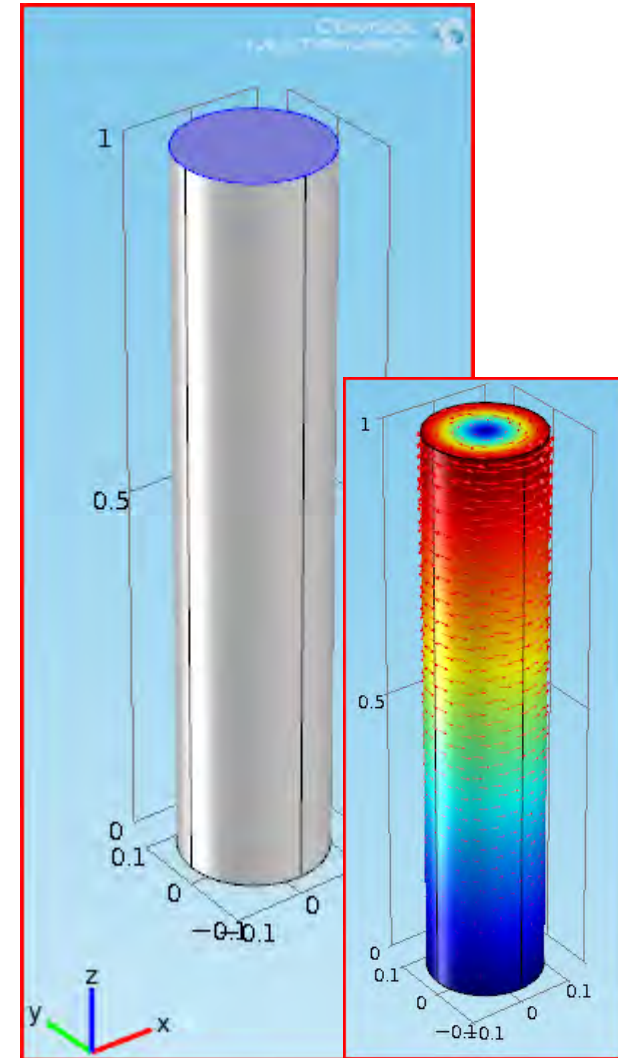
Moment load on a 3D solid model

- A moment loads M_z [N*m] can be added via:
- The new **Rigid Body Connector**, but this implies that the boundary becomes “rigid” which is not always desired
- A *Boundary Load* condition, as for the following 3D example applying a moment to the end 2D area of a cylinder defined in the x - y plane:

- F_x [N] = $- 2*(y-Y_0) / R1*M_z$ [N*m] / R1
- F_y [N] = $+ 2*(x-X_0) / R1*M_z$ [N*m] / R1

Where (X_0, Y_0) is a point defining the axis and $R1$ [m] is the full cylinder radius. Again by implicitly integrating these force values over the circular section we find back the total moment M_z [N*m]. A cylindrical coordinate system could also be used.

- If the moment is applied to the rim (to the edge or to a certain height along the cylinder edge) then the factor “2” above should be dropped!



Driven Moment load example

- You know **where** to apply a moment (or a force) load, but **not** its final value
- You know the resulting moment **MzLoad** [N*m] you want to see at a given interface location.
- If this location has a Fixed BC you can use the weak expression and the resulting *Lagrange Multipliers* **lm** to precisely extract the moment seen and have COMSOL multiphysics doing a means square optimization of the load for you to match the desired output value.
- The **Boundary Load** case can be expressed as on previous slide.
- The resulting **measured** moment **Mzm** [N*m] on the **Fixed Boundary** is obtained by integrating the *Lagrange Multipliers* defined as a *Variable*:
 - **$Mzm [N*m] = \text{intop1}(u_lm * y - v_lm * x)$**
- Less precise results will be achieved by using the surface traction forces **solid.Tax** [Pa] instead of **u_lm** [Pa] (respectively **solid.Tay** [Pa] ...).

Cont...

Driven Moment load example

cont.

- We have defined the desired load **MzLoad** [N*m] as a **Parameter**, the **Boundary Load** condition **Fx**, **Fy** [N], the measuring **Variable Mzm** [N*m].
- Remains to define a **Global Equation** in the solid physics to allow COMSOL Multiphysics to generate and to control the load moment **Mz** [N*m] (as used for the load case above) and defined by the equation **Mzm – MzLoad = 0**.

Parameters

Name	Expression	Value
MzLoad	1000[N*m]	1000 N·m
R1	0.1[m]	0.1 m

Variables

Geometric Scope: Entire model

Name	Expression	Unit	Descrip
rx	$\sqrt{x^2+y^2}$	m	
Thz	$-0.5[\text{rad}] \cdot (u_y - v_x)$		
MMe	$\text{intop_load}(-(\text{smsld.Tax} \cdot y - \text{smsld.Tay} \cdot x))$	N·m	
MMf	$\text{intop_fix}((\text{smsld.Tax} \cdot y - \text{smsld.Tay} \cdot x))$	N·m	
Mzm	$\text{intop_fix}(-(\text{v_lm} \cdot x - \text{u_lm} \cdot y))$	m ³	

Boundary Load

Boundaries: Selection: Manual

4

Equation

Coordinate System Selection: Global coordinate system

Force: Load type: Total force

Total force:

$-2 \cdot y / R1 \cdot \text{mod1.Mz}[\text{N} \cdot \text{m}] / R1$	x	
$2 \cdot x / R1 \cdot \text{mod1.Mz}[\text{N} \cdot \text{m}] / R1$	y	N
0	z	

Global Equations

Global Equations

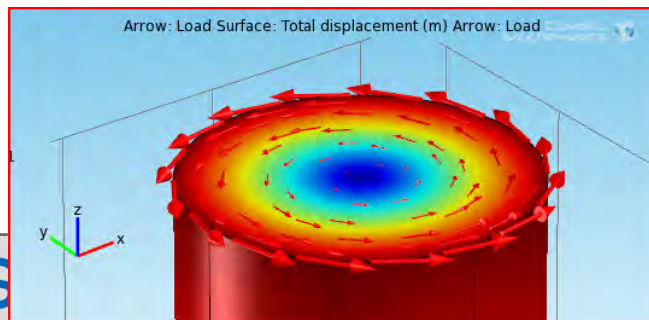
$f(u, u_t, u_{tt}, t) = 0, u(t_0) = u_0, u_t(t_0) = u_{t0}$

Name	f(u, u_t, u_{tt}, t)	Initial value (u_0)	Initial v
Mz	$Mzm - MzLoad$	MzLoad	0

Moment applied for a given rotation

- What if we do not know the true moment to be applied, but we have the desire to achieve a given rotation?
- A metrology of the desired angle **ThetaMeasured** must be performed:
 - via the *curl* of the deformation field $\mathbf{Thz} = 0.5 \text{ [rad]} * (\mathbf{vX} - \mathbf{uY})$, or
 - an $\mathbf{arctan2}(\mathbf{v}, \mathbf{u})$, or even
 - a **Rigid Body Connection** would do
- Introduce the new *Global Equation*, linking **Mz** [N*m], the applied moment, to the difference between achieved and desired angle

$$\mathbf{ThetaMeasured} - \mathbf{ThetaDesired} = 0$$



Global Equations				
Global Equations				
$f(u, u_t, u_{tt}, t) = 0, u(t_0) = u_0, u_t(t_0) = u_{t0}$				
Name	f(u, ut, utt, t)	Init...u_0)	Initi..._t0)	D...n
Mz	ThetaMeasured - ThetaDesired	0	0	

Domain reduction to CoG with a Point mass & load

- What if one could tell COMSOL Multiphysics:
“Select these **Domains** and reduce them to a point mass and inertia at the centre of gravity (CoG), and link rigidly, or softly, to the interface **Boundaries**.”
- Such a case must be set up by user-written equations, or by linking in a 2D beam model, where *Point Mass*’ and *Point Loads* are defined, in addition to rotations, **and** rotations at solid model level are required to link to the beam.
- Simplest is to add a **Point** on an existing boundary. If the point is well defined on a surface in the **Geometry** node, it will appear as a “hard node point” on the **Boundary Entity**. Thereafter one can apply a **Boundary Point Load** corresponding to the point mass m [kg] times the desired acceleration.
- Even better, to avoid the singularities of a **Point Load**, the load can be distributed over a surface via a **Boundary Load**.

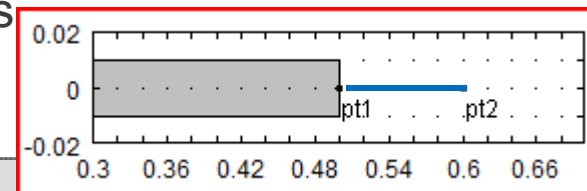
Cont...

Point load in 2D solid model

- But, one must not forget that a load will **not be** considered in an eigenfrequency, nor for a frequency sweep, analysis.
Because the internal physics equations of COMSOL Multiphysics are transformed and external forces are ignored if not directly related to the eigenfrequency variable ***lambda* [1/s]** respectively frequency ***freq* [Hz]** (cleaner: define your own *MyLambda*).
- Workaround:** Add a **Boundary Load** depending on the eigenfrequency ***lambda*** such as: $(m * g_const) - lambda^2 * m * u$
- Works also for the stationary study as COMSOL set ***lambda*=0** (just as ***t*=0**)
- Continuing in this way, one can further add to **Fx** some *Parameters* (or *Variables*) such as damping (**damp [m/s]**) and any solid spring constant (**stiffx [N/m]**) to the above equation, and defining the full boundary load along “**x**” as:
$$(m * g_const) + (stiffx + lambda * damp - lambda^2 * m) * u$$
- To adapt the above equations to a *Frequency Sweep*, one can add a simple *Global Variable* stating: ***lambda* = -i*2*pi*freq** Note: variables cannot be *disabled*, today
- I have **not** discussed inertia loading, this requires solid rotations and is left as exercise.

What if the CoG of the load is outside of any Domain?

- What if the reduced model *Point* of the CoG is **not** situated on any 3D *Boundary*, and is **outside** of any *Domain*?
- Any single point in space **not** belonging to any *Domain* will be considered as an **orphan** and will **not** be included in the *Entities list*, so one cannot add any equations to these orphan points.
- **General** or **Coefficient Form PDE** physics (v4.1) is restricted to *Domains*, not allowing us to select single points.
- **Weak Form Point PDE (wp)** physics accept points. However, v4.1 **wp** does not accept any eigenfrequency analysis, which means we are restricted to stationary cases.
- Each Point could be added as a Load case fully via equations, but then there is no handy geometrical point to visualize the results
- Remains: linking *beams* to *solid* physics



Future, nice-to-have features

- My wish list is long, here are some items:
 - “Soft link” BC’s, as complement to the Rigid Boundary Connector
 - Automatic domain reduction to their corresponding Point Mass and Point Load singular items, both positioned at the selected domain’s CoG.
 - To couple in inertia loads, one need a better access to the rigid body rotations and their derivatives.
- Today the closest one can get to rotations is to take the curl of the displacement field, which for **small angles** corresponds nicely to the rotations:
$$\text{Thx} = 0.5 \text{ [rad]} * (wY-vZ)$$
$$\text{Thy} = 0.5 \text{ [rad]} * (uZ-wX)$$
$$\text{Thz} = 0.5 \text{ [rad]} * (vX-uY)$$
- Note the uppercase **X**, **Y**, **Z** required if the **Spatial Frame** is active as it is by default in v4.

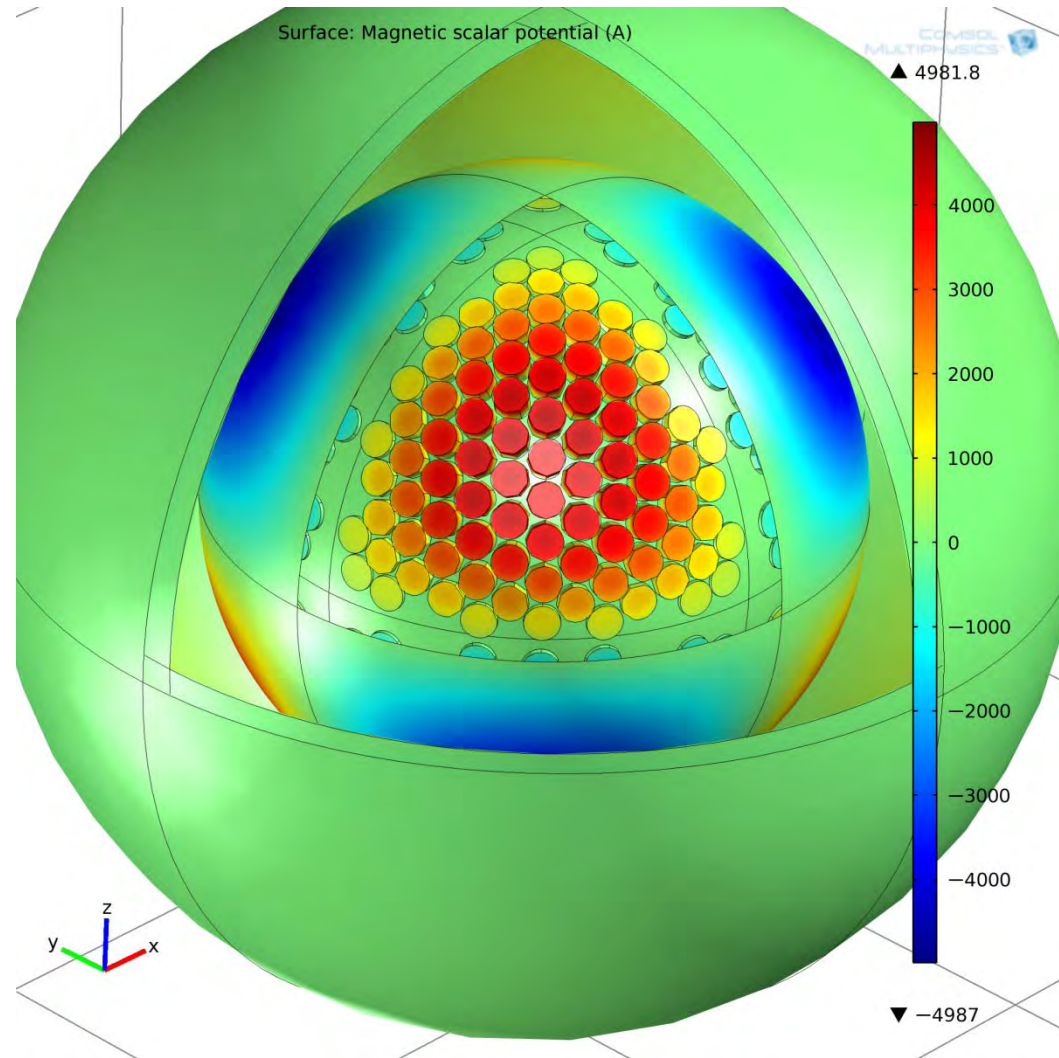
New and Future, nice-to-have features

cont...

- New in v4: the **mass participation factors** have appeared for eigenfrequency analysis (see **Study - Solver Configuration – Eigenvalue Solver - Output sub-node**)
Mass participation factors are an essential tool to analyze and to hierarchies eigenfrequencies, in particular for automation and control applications
- **Participation Factors** applies also to the rotations such to sort eigenmodes that are of importance to rotational degrees of freedom, not covered today.
- **Model validation and verification**: to have a direct access (and printout) of the model partial and total mass, centre of gravity, and full inertia tensor for different coordinate systems; the minimum being the CoG and the default coordinate systems.
Systematic comparison between the CAD values and the meshed values of these variables ensure that the FEM model has correct material definitions and scale.
- Today, all these values must be defined as user variables and post-processed separately which is time consuming and error prone.

What we can already do today in v4.1

- Spherical rotor made up from 725 Domains all in COMSOL Multiphysics geometric v4.1 CAD builder
- More than 690 small permanent magnets of different shapes, aligned on a spherical iron core part
- Arbitrary 3D rotation via quaternion's notation
- In preparation: 20 stator coils and closing field iron
- Solves in a few minutes, but model tree navigation is slow
- For details: see the poster of Leopoldo ROSSINI, CSEM sa



Conclusions

- COMSOL Multiphysics is today *the unique* tool allowing mixing many types of different physical models under the same software environment.
- The new v4 environment is particularly powerful and allows for very efficient modeling
- Engineers with a long background in older FEM tools, need some time to find their marks, and today, one has to learn-back how to write out the physics equations for many load cases, even in the classical domain of structural FEM physics.
- COMSOL Multiphysics latest release v4.1 has incorporated several useful new features, also for the structural domain, and I have been told that more is in preparation.
- Efficient *Engineering Structural Physics* is a must when COMSOL Multiphysics will really start to take shares from the classical FEM market, as most engineers have a long-term background in structural analysis, but need today to go far beyond.

Have fun COMSOL-ing !

Thank you for your attention!