

Version 2020

# Opera-3d User Guide

Version 2020

Dassault Systèmes UK Ltd  
Network House, Langford Locks  
Kidlington, Oxfordshire, OX5 1LH, UK

June 2020

# Legal Notices

**SIMULIA Opera 2020** is © 1984-2020 Dassault Systèmes UK Limited

This page specifies the patents, trademarks, copyrights, and restricted rights for **SIMULIA Opera 2020**:

## Trademarks

Opera, 3DEXPERIENCE, the Compass icon, the 3DS logo, CATIA, SOLIDWORKS, ENOVIA, DELMIA, SIMULIA, GEOVIA, EXALEAD, 3D VIA, BIOVIA, NETVIBES, IFWE and 3DEXCITE are commercial trademarks or registered trademarks of Dassault Systèmes, a French "société européenne" (Versailles Commercial Register # B 322 306 440), or its subsidiaries in the United States and/or other countries. **All other trademarks are owned by their respective owners.** Use of any Dassault Systèmes or its subsidiaries trademarks is subject to their express written approval.

DS Offerings and services names may be trademarks or service marks of Dassault Systèmes or its subsidiaries.

## Third-Party Copyrights Notices

Certain portions of SIMULIA Opera contain elements subject to copyright owned by the following entities:

© Intel Corporation

Copyright © Mark J. Kilgard, 1994, 1995, 1996, 1998, 2000, 2006, 2010

© Magnequench Neo Powders, PTE. Ltd.

© Sumitomo Electric Industries, Ltd,

Some procedures are based on routines in Numerical Recipes: The Art of Scientific Computing, published by Cambridge University Press, and are used by permission."

and

First Edition FORTRAN :"Copyright(C)1986 Numerical Recipes Software"

First Edition C :"Copyright(c)1987,1988 Numerical Recipes Software"

Second Ed.FORTRAN :"Copyright(c)1986,1992 Numerical Recipes Software"

Second Ed.C :"Copyright(c)1987-1992 Numerical Recipes Software"

Intel Python Distribution is copyright © 2018 Intel Corporation, and provided under the following terms:

Use and Redistribution. You may use and redistribute the software (the "Software"), without modification, provided the following conditions are met:

- \* Redistributions must reproduce the above copyright notice and the following terms of use in the Software and in the documentation and/or other materials provided with the distribution.
- \* Neither the name of Intel nor the names of its suppliers may be used to endorse or promote products derived from this Software without specific prior written permission.
- \* No reverse engineering, decompilation, or disassembly of this Software is permitted.

Limited patent license. Intel grants you a world-wide, royalty-free, non-exclusive license under patents it now or hereafter owns or controls to make, have made, use, import, offer to sell and sell ("Utilize") this Software, but solely to the extent that any such patent is necessary to Utilize the Software alone. The patent license shall not apply to any combinations which include this software. No hardware per se is licensed hereunder.

Third party and other Intel programs. "Third Party Programs" are the files listed in the "third-party-programs.txt" text file that is included with the Software and may include Intel programs under separate license terms. Third Party Programs, even if included with the distribution of the Materials, are governed by separate license terms and those license terms solely govern your use of those programs.

**DISCLAIMER.** THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT ARE DISCLAIMED. THIS SOFTWARE IS NOT INTENDED FOR USE IN SYSTEMS OR APPLICATIONS WHERE FAILURE OF THE SOFTWARE MAY CAUSE PERSONAL INJURY OR DEATH AND YOU AGREE THAT YOU ARE FULLY RESPONSIBLE FOR ANY CLAIMS, COSTS, DAMAGES, EXPENSES, AND ATTORNEYS' FEES ARISING OUT OF ANY SUCH USE, EVEN IF ANY CLAIM ALLEGES THAT INTEL WAS NEGLIGENT REGARDING THE DESIGN OR MANUFACTURE OF THE MATERIALS.

**LIMITATION OF LIABILITY.** IN NO EVENT WILL INTEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH

DAMAGE. YOU AGREE TO INDEMNIFY AND HOLD INTEL HARMLESS AGAINST ANY CLAIMS AND EXPENSES RESULTING FROM YOUR USE OR UNAUTHORIZED USE OF THE SOFTWARE.

No support. Intel may make changes to the Software, at any time without notice, and is not obligated to support, update or provide training for the Software.

Termination. Intel may terminate your right to use the Software in the event of your breach of this Agreement and you fail to cure the breach within a reasonable period of time.

Feedback. Should you provide Intel with comments, modifications, corrections, enhancements or other input ("Feedback") related to the Software Intel will be free to use, disclose, reproduce, license or otherwise distribute or exploit the Feedback in its sole discretion without any obligations or restrictions of any kind, including without limitation, intellectual property rights or licensing obligations.

Compliance with laws. You agree to comply with all relevant laws and regulations governing your use, transfer, import or export (or prohibition thereof) of the Software.

Governing law. All disputes will be governed by the laws of the United States of America and the State of Delaware without reference to conflict of law principles and subject to the exclusive jurisdiction of the state or federal courts sitting in the State of Delaware, and each party agrees that it submits to the personal jurisdiction and venue of those courts and waives any objections. The United Nations Convention on Contracts for the International Sale of Goods (1980) is specifically excluded and will not apply to the Software.

\*Other names and brands may be claimed as the property of others.

SIMULIA Opera may include open source software components. Source code for these components is available upon request. The original licensors of said open source software components provide them on an "as is" basis and without any liability whatsoever to customer (or licensee).

IP Asset Name	IP Asset Version	Copyright Notice
<b>Under Other License Terms</b>		
Libzip2	1.0.6	Copyright © 1996-2007 Julian R Seward. All rights reserved.
OpenGLUT	0.6.3	Copyright © 2004 The OpenGLUT Contributors. All Rights Reserved.

IP Asset Name	IP Asset Version	Copyright Notice
QUADPACK	None	<p>Copyright © 1987 R. Piessens, Elise de Doncker</p> <ul style="list-style-type: none"> <li>• the names of the Copyright holders or contributors may not be used to endorse or promote any derived Software without prior permission;</li> <li>• the Quadpack Software is provided "as is", without warranties;</li> <li>• and we deny any liability for situations resulting from the use of this Software</li> </ul>
Under Boost Software License 1.0 (BSL-1.0) <a href="https://opensource.org/licenses/BSL-1.0">https://opensource.org/licenses/BSL-1.0</a> :		
Boost	1.68.0	Copyright © 2018 Boost Community
Boost	1.71.0	Copyright © 2019 Boost Community
Under 2-clause BSD License (BSD-2-Clause) <a href="https://opensource.org/licenses/BSD-2-Clause">https://opensource.org/licenses/BSD-2-Clause</a> :		
Pygments	2.2.0	Copyright © 2006-2017 by the Pygments team
Sphinx	2.2.1	Copyright © 2007-2019 by the Sphinx team. All rights reserved.
Under 3-clause BSD License (BSD-3-Clause) <a href="https://opensource.org/licenses/BSD-3-Clause">https://opensource.org/licenses/BSD-3-Clause</a> :		
ARPACK	1996	Copyright (c) 1996-2008 Rice University. Developed by D.C. Sorensen, R.B. Lehoucq, C. Yang, and K. Maschhoff. All rights reserved.
conda-pack	0.4.0	Copyright © 2017, Jim Crist and contributors

IP Asset Name	IP Asset Version	Copyright Notice
hwloc	1.11.6	<p>Copyright © 2004-2006 The Trustees of Indiana University and Indiana University Research and Technology Corporation. All rights reserved.</p> <p>Copyright © 2004-2005 The University of Tennessee and The University of Tennessee Research Foundation. All rights reserved.</p> <p>Copyright © 2004-2005 High Performance Computing Center Stuttgart, University of Stuttgart. All rights reserved.</p> <p>Copyright © 2004-2005 The Regents of the University of California. All rights reserved.</p> <p>Copyright © 2009 CNRS</p> <p>Copyright © 2009-2016 Inria. All rights reserved.</p> <p>Copyright © 2009-2015 Université Bordeaux</p> <p>Copyright © 2009-2015 Cisco Systems, Inc. All rights reserved.</p> <p>Copyright © 2009-2012 Oracle and/or its affiliates. All rights reserved.</p> <p>Copyright © 2010 IBM</p> <p>Copyright © 2010 Jirka Hladky</p> <p>Copyright © 2012 Aleksej Saushev, The NetBSD Foundation</p> <p>Copyright © 2012 Blue Brain Project, EPFL. All rights reserved.</p> <p>Copyright © 2015 Research Organization for Information Science and Technology (RIST). All rights reserved.</p> <p>Copyright © 2015-2016 Intel, Inc. All rights reserved.</p>
QtSolutions	N/A	Copyright © 2013 Digia Plc and/or its subsidiary(-ies).
Under HDF5 (Hierarchical Data Format 5) License <a href="https://support.hdfgroup.org/ftp/HDF5/releases/COPYING">https://support.hdfgroup.org/ftp/HDF5/releases/COPYING</a> :		
HDF5	1.8.18	<p>HDF5 (Hierarchical Data Format 5) Software Library and Utilities</p> <p>Copyright 2006-2016 by The HDF Group.</p> <p>NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities</p> <p>Copyright 1998-2006 by the Board of Trustees of the University of Illinois.</p> <p>All rights reserved.</p>
Under ICU License <a href="https://github.com/unicode-org/icu/blob/master/icu4c/LICENSE">https://github.com/unicode-org/icu/blob/master/icu4c/LICENSE</a> :		

IP Asset Name	IP Asset Version	Copyright Notice
ICU	56.1	Copyright © 1995-2015 International Business Machines Corporation and others All rights reserved.
Under MIT License <a href="https://opensource.org/licenses/MIT">https://opensource.org/licenses/MIT</a> :		
dxfgrabber	1.0.0	Copyright © 2012, Manfred Moitzi
Ezdx	0.10.2	Copyright © 2011-2018, Manfred Moitzi
JSON for Modern C++	3.6.1	Copyright © 2013-2019 Niels Lohmann
Pytest	5.0.1	Copyright © 2004-2019 Holger Krekel and others
Under OpenSSL License <a href="https://www.openssl.org/source/license.html">https://www.openssl.org/source/license.html</a> :		
openSSL	1.0.2m	Copyright © 1998-2017 The OpenSSL Project. All rights reserved. Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved.
openSSL	1.1.1a	Copyright © 1998-2018 The OpenSSL Project. All rights reserved. Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved.

The following components are distributed and licensed under the terms of their original licenses:

Under GNU Lesser General Public License version 2.1 (LGPL-2.1) <a href="https://opensource.org/licenses/LGPL-2.1">https://opensource.org/licenses/LGPL-2.1</a> :		
Numerical diagonalization of 3x3 matrices	12-Mar-12	Copyright Joachim Kopp Numerical diagonalization of hermitian 3x3 matrices arXiv.org preprint: physics/0610206 Int. J. Mod. Phys. C19 (2008) 523-548
nose	1.3.7	Copyright © 2015 Jason Pellerin

Qwt	6.1.4	<p>Copyright © 1997 Josef Wilgen Copyright © 2002 Uwe Rathmann</p> <p>The Qwt library and included programs are provided under the terms of the GNU LESSER GENERAL PUBLIC LICENSE (LGPL) with the following exceptions:</p> <ol style="list-style-type: none"> <li>1. Widgets that are subclassed from Qwt widgets do not constitute a derivative work.</li> <li>2. Static linking of applications and widgets to the Qwt library does not constitute a derivative work and does not require the author to provide source code for the application or widget, use the shared Qwt libraries, or link their applications or widgets against a user-supplied version of Qwt. If you link the application or widget to a modified version of Qwt, then the changes to Qwt must be provided under the terms of the LGPL in sections 1, 2, and 4.</li> <li>3. You do not have to provide a copy of the Qwt license with programs that are linked to the Qwt library, nor do you have to identify the Qwt license in your program or documentation as required by section 6 of the LGPL.</li> </ol> <p>However, programs must still identify their use of Qwt.</p>
Under GNU Lesser General Public License version 3 (LGPL-3.0) <a href="https://open-source.org/licenses/LGPL-3.0">https://open-source.org/licenses/LGPL-3.0</a> :		
Lime Report	1.4.7	Copyright © 2015 by Alexander Arin (arin_a@bk.ru)
Qt	5.12.4	Copyright © 2019 The Qt Company

Other license terms:

libbz2:

This program, "bz2", the associated library "libbz2", and all documentation, are Copyright (C) 1996-2007 Julian R Seward. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above Copyright notice, this list of conditions and the following disclaimer.
2. The origin of this Software must not be misrepresented; you must not claim that you wrote the original Software. If you use this Software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original Software.
4. The name of the author may not be used to endorse or promote products derived from this Software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ""AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS

SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

#### OpenGLUT:

Copyright (c) 2004 The OpenGLUT Contributors. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies or substantial portions of the Software.

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE OPENGLUT CONTRIBUTORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of The OpenGLUT Contributors shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from The OpenGLUT Contributors.

## Restricted Rights

This clause applies to all acquisitions of Dassault Systèmes Offerings by or for the United States government, or by any prime contractor or subcontractor (at any tier) under any contract, grant, cooperative agreement or other activity with the United States government. The software, documentation and any other technical data provided hereunder is commercial in nature and developed solely at private expense. The Software is delivered as "Commercial Computer Software" as defined in DFARS 252.227-7014 or as a "Commercial Item" as defined in FAR 2.101(a) and as such is provided with only such rights as are provided in Dassault Systèmes standard commercial end user license agreement. Technical data is provided with limited rights only as provided in DFAR 252.227-7015 or FAR 52.227-14, whichever is applicable. The terms and conditions of the Dassault Systèmes standard commercial end user license agreement shall pertain to the United States government's use and disclosure of this software, and shall supersede any conflicting contractual terms and conditions. If the DS standard commercial license fails to meet the United States government's needs or is inconsistent in any respect with United States Federal law, the United States government agrees to return this software, unused, to DS. The following additional statement applies only to acquisitions governed by DFARS Subpart 227.4: "Restricted Rights - use, duplication and disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252-227-7013.

**SIMULIA Opera** © 1984-2020 Dassault Systèmes UK Limited.

# Contents

---

Legal Notices .....	2
Contents .....	11
<b>Chapter 1 Introducing Opera-3d .....</b>	<b>24</b>
Introduction .....	24
Opera Manager, Optimizer and Licensing .....	25
Opera-3d Modeller Tabbed Menu Interface .....	26
Opera-3d Post-Processor Tabbed Menu Interface .....	27
Opera-3d Analysis Programs .....	28
Magnetostatic, Electrostatic and Current Flow .....	28
Harmonic, Transient and Fixed Velocity Electromagnetic .....	28
Transient Electromagnetic with Motion .....	28
Magnetization .....	28
Quench .....	28
Charged Particle .....	28
Harmonic and Modal High Frequency .....	29
Static and Modal Stress .....	29
Static and Transient Thermal .....	29
Coupled Analysis .....	29
Multiphysics .....	31
Structure of the User Guide .....	33
Dialog Layout and Results in the User Guide Examples .....	34
<b>Chapter 2 Getting Started .....</b>	<b>35</b>
Introduction .....	35
The Model .....	35
Starting the Opera Manager .....	37
Starting the Modeller .....	37
The Modeller Window .....	38
Building the Geometry .....	39
The Permanent Magnet .....	39
Controlling the View .....	40
Properties of the Magnet .....	41

---

Building the Pole and Shim .....	42
Building the Frame .....	45
Surrounding Air and Boundary Conditions .....	50
Mesh Control Cylinder .....	50
Analysis Type .....	50
Exploiting Symmetry .....	50
Saving the Model .....	51
Creating the Model Body .....	52
Building the Finite Element Mesh .....	53
Surface Meshing .....	53
Volume Meshing .....	54
Defining Material Properties .....	55
Creating the Analysis Database .....	60
Running the Analysis .....	63
Post-Processing .....	65
Loading the Solution .....	65
Displaying the Model with Replications .....	65
Displaying Results .....	67
Results on the Geometry .....	67
Fields at a Point .....	68
Force Between Poles .....	69
Field Values on a Plane .....	69
Further work .....	72
<b>Chapter 3 Geometric Modeller Features .....</b>	<b>73</b>
Introduction .....	73
Topology Definitions .....	75
Introduction .....	75
Geometric Entities .....	75
Geometric Model Information .....	77
Bodies and Cells .....	77
Sketching Primitives .....	79
2D Sketching with Wire-Edges .....	89
Creating Edges and Sheet Faces from existing Points .....	90
Picking and Hiding Entities .....	94
Picking Bodies .....	95
Picking Other Entities .....	95
Drilling Down .....	96
Hiding Entities .....	97
Reset Picked Entities .....	98
Keyboard Commands while Picking .....	98

---

Picking Bodies by Name or Unique Name .....	99
Rapid Picking of Entities .....	100
Operations on Picked Entities .....	102
Tabbed Interface .....	102
Context Menus .....	102
Boolean Operations .....	104
Union with Regularization .....	105
Union without Regularization .....	106
Intersection with Regularization .....	107
Subtraction with Regularization .....	107
Subtraction without Regularization .....	109
Trim Overlap .....	109
Cutaway Overlap .....	112
Transforming and Copying Objects .....	113
Displacement .....	113
Changing the Shape of a Body .....	114
Making Replications of a Body .....	115
Names, Unique Names and Labels .....	115
Copying Cells, Faces and Edges .....	116
Sweeping Faces .....	117
Sweep a Face through a Distance .....	117
Sweep a Face along a Vector .....	119
Sweep a Face by Rotation .....	120
Sweep a Face along a Path .....	120
Lofting Between Surfaces .....	123
Face Tangent Parameters .....	124
Shell and Offset .....	127
Shell .....	127
Offset .....	127
Blend and Chamfer .....	129
Blending Exterior Edges .....	129
Chamfering Exterior Edges .....	130
Blending Interior Edges .....	130
Chamfering Interior Edges .....	132
Non-Manifold Edges .....	132
Morphing Operations .....	134
TWIST Command .....	136
STRETCH Command .....	138
BEND Command .....	139
MORPH Command .....	140
Local Coordinate Systems .....	143
Global Coordinate System .....	143

---

Rotated Coordinate System .....	143
Displaced Coordinate System .....	144
Moving the WCS .....	145
Mesh Control .....	147
Defining and Editing Conductors .....	153
Different Types of Conductors .....	155
Uses of Volume Data .....	160
Analysis Options and Database Creation .....	164
AC Eddy Current Analysis with Harmonic Electromagnetic solver .....	164
Database Creation .....	164
Add to existing Database .....	166
Transient Eddy Current Analysis with Transient Electromagnetic solver .....	167
Parameterizing Models and Rebuilding .....	169
Parts Libraries and CAD Files .....	172
CAD Files .....	172
Building a Model from Parts .....	173
<b>Chapter 4 Plate Capacitor Electrostatic Example .....</b>	<b>175</b>
Building and Running the Model .....	176
Creating the Geometry .....	176
Creating and Meshing the Model .....	189
Setting the Analysis Type and Boundary Conditions .....	189
Saving the Model File, Creating the Database and Running the Analysis .....	190
Post-Processing .....	192
Loading and Displaying the Model .....	192
Computing the Capacitance .....	192
Model Variants .....	194
<b>Chapter 5 Radiation Screen Transient EM Example .....</b>	<b>196</b>
Introduction .....	196
Building and Running the Model .....	197
Creating the Geometry .....	197
Setup the Analysis Options .....	205
Solving the Model .....	209
Post-Processing .....	211
Displaying Eddy Current Density .....	211
Computing Forces .....	213
Further work I - Improving the Model .....	217
Discussion .....	217
Further Work II - Extracting Field Data During the Simulation .....	219
Python script .....	219

---

Running the simulation .....	220
Simulation outputs and post-processing .....	220
<b>Chapter 6 Permanent Magnet Synchronous Machine .....</b>	<b>222</b>
Introduction .....	222
Magnetostatic model .....	223
Creating the geometry .....	223
Material Properties .....	233
Solver Settings .....	236
Post-Processing .....	237
Magnetostatic model at different rotor positions .....	240
Transient motion model .....	244
Model update .....	244
Post-processing .....	253
<b>Chapter 7 Winding Tool .....</b>	<b>256</b>
Introduction .....	256
Definitions and Terminology .....	257
Slots and coils per pole per phase .....	257
Average coil pitch .....	257
Basic winding .....	257
Winding feasibility .....	258
Winding Classification .....	259
Overlapping and non-overlapping: .....	259
Single and double layer: .....	259
Full-pitch and fractional pitch: .....	259
Integral and fractional slot: .....	259
Distributed and concentrated windings: .....	260
Winding Harmonics .....	261
Winding Factor (kw) .....	261
Winding MMF .....	262
Harmonic Order .....	262
Star of Slots .....	264
Basic Usage .....	265
Launching the winding tool .....	265
Use case 1: Fixed number of slots and poles .....	265
Use case 2: Range of number of slots .....	270
<b>Chapter 8 Induction Heating: Thermal and Multiphysics .....</b>	<b>273</b>
Introduction .....	273
The Model .....	273

---

Building the Thermal Model .....	275
Building the Plate Geometry and Mesh .....	275
Static Thermal Analysis .....	280
Setting up the Thermal Analysis .....	280
Setting the Thermal Properties .....	280
Save Model and Create the Database .....	282
Post-Processing the First Thermal Analysis .....	283
Thermal Results .....	283
Multiphysics Analysis .....	285
Mesh Generation and Database Creation .....	296
Post-Processing the Coupled Example .....	297
<b>Chapter 9 A Quench Example .....</b>	<b>301</b>
Introduction .....	301
The Model .....	302
Building the Model .....	304
Building the Solenoid Coil .....	304
Material and Volume Properties .....	309
Nonlinear Functions .....	309
Setting the User Defined Variables .....	310
Setting the Material and Wire Properties .....	311
Setting the Boundary Conditions .....	315
Defining the Circuit .....	317
Quench Analysis .....	320
Setting up the Quench Analysis .....	320
Running the Analysis .....	322
Post-Processing the Quench Analysis .....	323
Log of Analysis .....	323
Thermal Results .....	325
Advanced Quench Control .....	328
Coupled Quench Analysis .....	329
Modify the Original Model .....	329
Setting up the Coupled Solution .....	332
Setting the Additional Material Properties .....	333
Simplified Circuit Setting .....	334
Setting up the Analysis .....	334
Running the Analysis .....	338
Post-Processing the Coupled QUENCH Solution .....	339
Log of Analysis .....	339
Thermal and Eddy Current Results .....	340

---

<b>Chapter 10 A Charged Particle Example .....</b>	<b>343</b>
Introduction .....	343
Constructing the Model .....	345
Building the Geometric Model .....	345
Charged Particle Simulation .....	357
Boundary Conditions .....	357
Assigning Voltage Boundary Conditions .....	360
Specification of Emitters .....	361
Model Symmetry .....	363
Mesh Generation .....	364
Running the Charged Particle Simulation .....	366
Post-Processing, Analyzing the Results .....	367
Loading and Displaying the Model in the Post-Processor .....	367
Displaying Trajectories .....	367
Secondary Emission .....	376
Lossy Dielectrics .....	379
Beam Magnetic Fields .....	380
External Magnetic Fields .....	380
<b>Chapter 11 RF Cavity and Waveguide Modal and Harmonic HF Example .....</b>	<b>382</b>
Introduction .....	382
Cavity Modelling Using the Modal HF Solver .....	382
Waveguide Modelling Using the Harmonic HF Solver .....	383
Cavity Modelling Using the Modal HF Solver .....	385
Building the Model Geometry .....	385
Setting Options for the Modal HF Analysis .....	389
Model Body Creation and Meshing .....	393
Preparing and Running the Modal HF Analysis .....	394
Analysing the Results in the Post-Processor .....	397
Waveguide Modelling with Harmonic HF .....	401
Building the Model Geometry .....	401
Setting Options for Harmonic HF Analysis .....	404
Meshing and Creating the Analysis Database .....	409
Model Variant - Walled Waveguide .....	414
Post-Processing Harmonic HF Results .....	420
<b>Chapter 12 Stress Examples and Multiphysics .....</b>	<b>443</b>
Introduction .....	443
Cantilever Beam .....	444
Building the Geometry .....	444
Boundary Conditions .....	444

---

Material Properties .....	448
Exploiting Symmetry .....	448
Creating the Mesh and Running the Simulation .....	449
Displaying Results .....	451
Steel Plate Deformed by Energized Solenoid .....	458
Building the Geometry .....	458
Stress Analysis Considerations in the Magnetostatic Model .....	463
Materials .....	463
Constraints .....	465
Adding the Coil .....	468
Building the Mesh and Starting the Multiphysics Simulation .....	469
Post-Processing .....	470
<b>Chapter 13 An Example Using the Pre-Processor .....</b>	<b>476</b>
A Simple Model of an Inductor .....	476
The Base Plane of the Model .....	477
Starting the Pre-Processor .....	477
Defining the Base Plane Points .....	477
Defining the Base Plane Facets .....	484
Defining the Base Plane Subdivisions .....	487
Extruding in the Third Dimension .....	489
Objects in the 3D Model .....	491
Defining Materials and Potentials .....	491
Magnetic Boundary Conditions .....	499
Displaying the 3D Model .....	502
Defining the Conductor .....	506
Examining the Model with the 3d Viewer .....	510
Saving TOSCA Data .....	513
Creating the TOSCA Database File .....	513
Writing the Pre-Processor Data File .....	517
Leaving the Pre-Processor .....	518
Running TOSCA and Starting the Post-Processor .....	519
Examining the Basic Solution from TOSCA .....	520
Viewing the Model .....	520
2D Surfaces in the 3D Model Space .....	521
Evaluation of Solution at Points and Along Lines .....	523
Examining the Complete Device from TOSCA .....	526
Loading the Model .....	526
ELEKTRA Worked Example .....	529
Writing the ELEKTRA Database File .....	529
Writing the Pre-Processor Data File .....	534

---

Leaving the Pre-Processor .....	535
Running ELEKTRA and Starting the Post-Processor .....	536
Examining the Basic Solution from ELEKTRA .....	537
Loading and Displaying the Model in the Post-Processor .....	537
<b>Chapter 14 Application Notes .....</b>	<b>539</b>
Introduction .....	539
Basic Functionality .....	539
Electrical Machines .....	540
Other Applications .....	540
Python Applications .....	540
Meshing in the Modeller .....	541
Introduction .....	541
Element Types .....	542
Mesh Control .....	545
Layering .....	546
Mesh-Size Transition .....	548
Identifying Errors .....	550
Meshing - Typical Problems .....	551
Fixing a Meshing Error .....	552
Guidelines to Improve Mesh Generation .....	553
Periodicity in the Modeller .....	557
Rotational Symmetry around Z .....	559
Advanced Methods for Periodicity .....	560
Analysis of Parts of a Geometric Model .....	565
Static Current Flow and Magnetostatic Analyses .....	565
Using Packing Factor to Represent Laminations .....	570
Grain Oriented Materials .....	571
Complex Material Properties .....	572
Introduction .....	572
Complex Permeability Model .....	572
Hysteresis Losses .....	573
Restrictions of the Model .....	573
Complex Materials in Harmonic EM and Harmonic HF .....	574
Complex Permeability in Nonlinear Models .....	574
Tabular Functions .....	576
Tabular Function Files .....	576
Using Tabular Function Files to Create Functional Variables .....	579
Interpolation of Tabular Functions .....	580
Display of Tabular Functions .....	580
Simulation of Magnetization .....	581

---

Introduction .....	581
Using Magnetostatic solver: Packing Factor and Material Orientation .....	581
Limited Approach using Magnetostatic solver .....	582
Command Script .....	587
Using the Magnetization Analysis Module .....	589
Demagnetization In Service .....	593
Using Hysteretic Materials .....	596
Introduction .....	596
Example: An H-frame dipole .....	597
Boundary Conditions for Thin Gaps .....	603
Introduction .....	603
Specifying the Electric Insulator Boundary Condition .....	604
Segmented Surface Magnet Example .....	605
Thermal Contact Boundary Conditions .....	608
Using Surface Impedance Boundary Conditions in Harmonic Electromagnetic .....	610
Using Surface Impedance Boundary Conditions in Transient Electromagnetic .....	613
Setting up a Transient EM model with an SIBC material .....	613
Limitation of SIBC with "DC" fields .....	615
Limitations with neighbouring electrically conducting regions .....	617
Using Surface Impedance Boundary Conditions in Multiphysics (Electromagnetic and Thermal) Simulations .....	621
Thin Plate Approximation for Magnetostatic Simulations .....	625
Introduction .....	625
Example ship magnetic signature model .....	625
A second example: Magnetic shielding of an MRI magnet .....	631
Guidelines and limitations .....	633
Using Current Source Boundary Conditions .....	635
Power and Energy Calculation in Harmonic solutions .....	639
Power Calculations .....	639
Energy Calculations .....	639
General .....	640
Inductance Calculations in Opera-3d .....	641
Single Coil Model .....	641
Multiple Coil Models – Linear Materials .....	641
Multiple Coils – Nonlinear Materials .....	642
Flux Linkage in Meshed Conductors .....	643
Legendre Polynomials .....	645
Introduction .....	645
Legendre Polynomial Coefficients .....	645
Setting up the Model for Accurate Field Results .....	647
Results .....	649
Circuits in Harmonic and Transient EM simulations .....	651

---

Time Dependent Drive Functions .....	656
Example Circuits .....	659
Example 1: 3-Phase Transformer Circuit .....	659
Example 2: 3-Phase Drive Circuit for a Rotating Machine .....	662
Example 3: Hysteresis chopping .....	666
Example 4: A 12-Pulse Transformer Circuit using a 3-Phase Voltage Supply .....	667
Bulk Conductors .....	670
Introduction .....	670
Operation .....	670
Model .....	672
Circuit Operation .....	674
Simulation Results .....	674
Discussion .....	678
Restarting Simulations .....	681
Introduction .....	681
Applicable simulation types .....	682
Machine Analysis using Motional EM solver .....	684
Introduction .....	684
Group Labels .....	685
Gap Regions .....	685
Motion Control .....	692
Control File and Command File Editor .....	693
Logging of Variables .....	693
Generator Example .....	693
Periodic Models .....	694
Linear Actuator Example .....	698
Example Control File .....	705
Example 1 .....	705
Example 2 .....	707
Discussions .....	712
Opera in Simulink® System Analysis .....	713
Opera Analysis Block .....	714
Estimating Iron Loss in Opera-3d .....	717
Introduction .....	717
Methods for Quantifying Iron Loss .....	718
Preparation/Pre-Processing .....	722
Post-Processing .....	722
Modelling DC Current flow .....	726
Introduction .....	726
DC Current Flow in a Conducting Medium .....	726
Using Field Values from Current Flow Simulations .....	732
Lossy Dielectric Simulation .....	734

---

<b>Multiphysics Analysis .....</b>	<b>739</b>
Coupled Thermal and Stress .....	739
Coupled Electromagnetic and Stress (Magnetostriction) .....	746
Using Biot-Savart Coils in Multiphysics (Magnetostatic and Stress) Simulations .....	748
Combined Magnetic and Electric Fields .....	753
Q and g (R/Q) Factors from a Modal HF Solution .....	755
Definitions .....	755
Command Files to Compute Q-factor and Geometric Impedance .....	756
Coupled EM and Thermal Analysis in Opera-3d .....	758
Introduction .....	758
Implementation .....	758
Induction Heating Example .....	759
Scripts provided .....	760
Particle Collision and Scattering in the Charged Particle solver .....	762
Surface Secondary Emission .....	763
Volume Secondary Emission .....	768
Plasma Free Surface - Type 103 Emitter .....	772
Plasma Emitter .....	776
Parameterized Models in the Pre-Processor .....	785
Geometric Modeller .....	785
Pre-Processor .....	785
Single Phase Inductor, featuring 3 types of losses .....	787
<b>Chapter 15 Supplied Python Functionality .....</b>	<b>791</b>
Utility Classes .....	791
Use .....	791
Contents .....	791
Database Index .....	793
Purpose .....	793
Terminology .....	793
Available classes .....	793
Typical usage: Opera-3d/Solver .....	793
Typical usage: Opera-3d/Post .....	794
Typical usage: general .....	795
Notes .....	795
Database Extraction .....	796
Purpose .....	796
Terminology .....	796
Available classes .....	796
Typical usage: Opera-3d/Solver .....	797
Typical usage: Opera-3d/Post .....	798

---

Database Update .....	799
Purpose .....	799
Terminology .....	799
Available classes .....	799
Typical usage: Opera-3d/Solver .....	800
Notes .....	800
Steady-State Detection .....	801
Purpose .....	801
Methods: Variance Test .....	801
Methods: Mean Test .....	801
Parameters .....	802
Available classes .....	803
Typical usage: Opera-3d/Solver .....	803
Debugging .....	805
Loss Calculation .....	806
Purpose .....	806
Terminology .....	806
Available classes: time averaged losses .....	807
Available classes: iron losses .....	807
Available classes: loss parameters and movement .....	808
Available classes: multiple losses .....	808
Typical usage: Opera-3d/Solver .....	808
Typical usage: Opera-3d/Post .....	810
<b>Index .....</b>	<b>812</b>

# *Chapter 1*

## Introducing Opera-3d

### Introduction

---

Opera-3d is the pre and post-processing system for electromagnetics, thermal and stress solutions.

Finite element discretization forms the basis of the methods used in these analysis programs. This widely applicable technique for the solution of partial differential equations requires special enhancements to make it applicable to electromagnetic field calculations. Access to these features is supported by the **Geometric Modeller** and **Pre-Processor**. These programs provide facilities for the creation of finite element models, specification of complicated conductor geometry, definition of material characteristics (including nonlinear, anisotropic and hysteretic materials) and graphical displays for interaction with and examination of the data.

Similarly, the **Post-Processor** provides facilities necessary for calculating electromagnetic, temperature and displacement fields, displaying them as graphs and contour maps. The Post-Processor can also calculate and display many derived quantities and can plot particle trajectories through electromagnetic fields.

## Opera Manager, Optimizer and Licensing

---

All the programs within the Opera Suite can be started using the **Opera Manager**. This is an enhanced file browser which recognises the file types created by Opera and has menus and toolbars to access all the programs, options, User Guides and Reference Manuals.

The Opera Manager incorporates a **Batch Processor** for controlling the execution of the Opera analysis programs. It has a queuing system which prioritises analyses and allows multiple computers on a local area network to share the load.

The Opera Manager includes an **Optimizer** which can be used to run multiple designs and search for an optimal design which meets the specified criteria.

In Opera-3d, an analysis program may be run using a **Multi-threaded** version. The Opera Manager gives control over the number of threads that will be used by analyses, as a part of options for the Batch Queue. Opera-2d analyses always run with a single thread.

Opera software is licensed. The Opera Manager can list the licenses available, but the licenses are managed by the **Opera License Utility**. This is a separately installed package which must be installed and run on any computer which is used as a license server.

Full details of the Opera Manager, the Optimizer and the Opera License Utility can be found in the ***Opera Manager User Guide***.

## Opera-3d Modeller Tabbed Menu Interface

---

The Opera-3d Modeller Tabbed Menu Interface has 3 Tabs of commands:

- **Work:** contains commands for opening and saving files, reviewing and changing previously run commands, defining user variables and displaying models.
- **Modelling:** contains commands for creating, modifying and manipulating model geometries.
- **Analysis:** contains commands to define analysis specific data, generate the mesh and create analysis databases.

**Display buttons** are displayed at the bottom of the window and include commands to manipulate the display and perform other common operations.

## Opera-3d Post-Processor Tabbed Menu Interface

---

The Opera-3d Post-Processor Tabbed Menu Interface has 2 Tabs of commands:

- **Work:** contains commands for database access, defining user variables, model display, global options and units.
- **Post-Processing:** contains commands for visualizing results and deriving further information.

**Display buttons** are displayed at the bottom of the window and include commands to manipulate the display and perform other common operations.

## Opera-3d Analysis Programs

---

There are currently 9 analysis programs:

### Magnetostatic, Electrostatic and Current Flow

Solves nonlinear magnetostatic or electrostatic fields and current flow in three dimensions. It has been in use for many years and is being continually improved to increase its accuracy and efficiency. It uses a formulation based on total and reduced scalar potentials.

### Harmonic, Transient and Fixed Velocity Electromagnetic

Analyses time dependent electromagnetic fields, including the effects of eddy currents. There are 3 analysis options: the time variation can be transient, steady-state AC or eddy currents can be induced in moving conductors with a specified linear or fixed rotational velocity in the presence of a static field. Fixed Velocity can be applied to situations where the motion does not change the geometry, e.g. infinitely long rails or rotating disks.

### Transient Electromagnetic with Motion

Analyses transient electromagnetic fields in linear and rotating machines, with the option of mechanical coupling to determine the speed of the moving parts.

### Magnetization

Computes the magnetization of permanent magnet materials by time varying electromagnetic fields in three dimensions including the effects of eddy currents.

### Quench

Simulates a quench in a superconducting magnet. It models transient thermal and electromagnetic fields including heat sources and magnetic fields produced by conductors driven by time varying circuit currents.

### Charged Particle

Analyses electrostatic fields taking into account the effects of space charge created by beams of charged particles.

## Harmonic and Modal High Frequency

Analyses high frequency electromagnetic fields. There are two analysis options: steady-state AC or eigenvalue extraction.

## Static and Modal Stress

Simulates mechanical loading and deformation of a model. There are 2 analysis options:

- **Static Stress** calculates the strains, stresses and static deformations of the model. Loads can be applied using internal and external forces and pre-strains, either applied directly to the model or imported from other electromagnetic or thermal Opera analyses.
- **Modal Stress** calculates the natural modes of vibration of a body at rest. No internal or external loads or pre-strains are allowed in.

## Static and Transient Thermal

Analyses static and transient thermal fields arising as a result of electromagnetic heating and external heat sources. The calculated temperatures can be fed back to electromagnetic analyses in order to vary material properties.

## Coupled Analysis

Several of the Opera analysis programs do coupled analysis in which more than one physical process is modelled concurrently. The Opera-3d environment allows other types of coupled analysis in which the results of one model can be used as data for another.

## Transient Electromagnetic with Motion

Transient Motion analysis solves the motion of rigid bodies coupled with electromagnetic fields:

- the electromagnetics provides forces for the rigid body motion analysis, and
- the rigid body motion analysis provides position and motion to the electromagnetics.

## Lossy dielectrics

The lossy dielectric option can be selected in **Charged Particle**, **Electrostatic** and **Current Flow** solvers to calculate the electric fields from charges which can move in materials which have a very small conductivity.

## Circuits

All the transient electromagnetic analysis programs can model circuits consisting of current or voltage sources, resistors, capacitors, inductors, diodes and switches. Some components are not available in **Harmonic Electromagnetic**.

## Quench

Quench solves three systems concurrently:

- thermal analysis which uses the heat generated from currents calculated by the electromagnetic analysis and critical currents in the superconducting coils which depend on the local flux density;
- electromagnetic analysis which uses conductivities that vary as a function of the temperature calculated by the thermal analysis; and
- currents in circuits.

## Eddy current analysis

The eddy current analysis programs can import temperature from thermal analyses to make use of conductivities which vary with temperature.

## Charged Particle

The **Charged Particle** analysis program calculates particle trajectories through a combination of electric and magnetic fields. The magnetic fields can be provided by

- currents in coils,
- the particles themselves or
- a separate analysis using **Magnetostatic** or one of the other electromagnetic analysis programs.

## Stress

The **Static Stress** analysis program calculates the deformation, stress and strain of a body which is subjected to internal and external forces. The forces can be provided by

- directly applied volume or boundary conditions,
- electromagnetic loads,
- pre-strains calculated from magnetic or electric fields (magnetostriction and electrostriction), or
- pre-strains calculated from a background temperature change or from an imported thermal solution.

## Magnetostatic

The **Magnetostatic** analysis program can calculate source fields from currents in coils or can import sources from a separate analysis.

## Multiphysics

In addition to the Coupled Analysis features described above, Opera-3d can also support a sequence of analyses in a single **Multiphysics** run. A Multiphysics run will consist of two or more consecutive analyses (usually using different Analysis Programs) where the results from any previous analysis are available to subsequent analyses in the sequence.

For example, if pre-stressing due to temperature rise must be included in a **Static Stress** analysis, a **Thermal** analysis will be the first analysis in the sequence and the calculated temperatures will be automatically passed to the **Static Stress** analysis to use in conjunction with the material thermal expansion coefficient. An example of this is shown in [Multiphysics Analysis \[page 739\]](#).

In addition to the solved variables, such as:

- magnetic scalar potential in magnetostatic analysis
- temperature in thermal analysis
- deflection in mechanical analysis

derived values, such as magnetic flux density, heat flow and stress/strain, will also be available to subsequent analyses after they are calculated by a particular analysis in the sequence. Most commonly, these quantities will be used to express material properties as a function that is dependent on the results from another physics domain.

It should be noted that if the same type of physics (electromagnetic, thermal or mechanical) is included more than once in the sequence of analyses, the most recent calculation of the stored values will be used in subsequent analyses.

### Other analysis values

The electromagnetic analyses will also automatically compute two further quantities that will be automatically used as "source" terms in subsequent thermal or mechanical analyses.

All electromagnetic analyses will calculate body force density, which can be used as a load in mechanical stress analyses.

Harmonic analyses will calculate loss intensity due to eddy currents in conducting electrical materials.

### Displaced nodes

If a **Static Stress** analysis is included in the multiphysics sequence, the displaced position of the nodes in the mesh may be used in subsequent electromagnetic or thermal analyses, allowing the effect of the deflection of a structure on its magnetic or thermal performance to be assessed.

### Types of analysis

Static, harmonic and transient analysis programs can be included in a Multiphysics simulation. In the case of transient simulations, the fields available at the last time-output will be passed on to

subsequent analysis programs. Time-averaged fields can also be computed, using Python, from a transient solution and used to inform subsequent analyses.

The Modal High Frequency analysis may be included as part of the sequence, providing the simulation is constrained to only calculate results at a single eigenfrequency.

The complete list of programs which may be included as part of a Multiphysics sequence is:

- Magnetostatic
- Electrostatic
- Static Current Flow
- Harmonic Electromagnetic
- Transient Electromagnetic
- Fixed Velocity Electromagnetic (Linear)
- Fixed Velocity Electromagnetic (Rotational)
- Transient Electromagnetic with Motion
- Charged Particle
- Harmonic High Frequency
- Modal High Frequency
- Static Thermal
- Static Stress

Further examples of Multiphysics simulations may be found in [Induction Heating: Thermal and Multiphysics \[page 273\]](#) and [Stress Examples and Multiphysics \[page 443\]](#)

## Structure of the User Guide

---

The Opera-3d User Guide shows the user how to use the software. It is structured into the following chapters.

- **Getting Started [page 35]**

This chapter gives a detailed description of how a model is prepared and analysed. New users are encouraged to spend some time going through this chapter, as it will answer many questions that can otherwise arise when using the software.

- **Geometric Modeller Features [page 73]**

This chapter describes in more detail the features of the Modeller which can be used to build complex geometries more efficiently.

- **Tutorials**

A series of examples using the software is included. Each attempts to high-light a typical application using various analysis modules.

- [Plate Capacitor Electrostatic Example \[page 175\]](#)
- [Radiation Screen Transient EM Example \[page 196\]](#)
- [Permanent Magnet Synchronous Machine \[page 222\]](#)
- [Induction Heating: Thermal and Multiphysics \[page 273\]](#)
- [A Quench Example \[page 301\]](#)
- [A Charged Particle Example \[page 343\]](#)
- [RF Cavity and Waveguide Modal and Harmonic HF Example \[page 382\]](#)
- [Stress Examples and Multiphysics \[page 443\]](#)
- [An Example Using the Pre-Processor \[page 476\]](#)

- **Application Notes [page 539]**

This chapter contains a number of useful techniques that can be used for performing various tasks. If a question arises as to how to use the software in a particular way, this chapter should first be consulted in case an answer is presented. Note that Application Notes assume basic knowledge of the workings of Opera, which can be gained by going through the Getting Started example.

All the features of Opera-3d are described in detail in the ***Opera-3d Reference Manual***.

## Dialog Layout and Results in the User Guide Examples

---

The dialogs, shown in the worked examples and application notes in this document, may have been improved or contain additional features in the latest version of the Opera-3d software. However, each dialog will always retain the functionality illustrated, and all the examples can be performed as shown.

Additionally, there are improvements to the mesh generator, which will lead to more accurate solutions for some examples. As a consequence of this, the number of nodes and elements may be different and derived results may not be exactly the same as values given in the User Guide.

# **Chapter 2**

## **Getting Started**

### **Introduction**

---

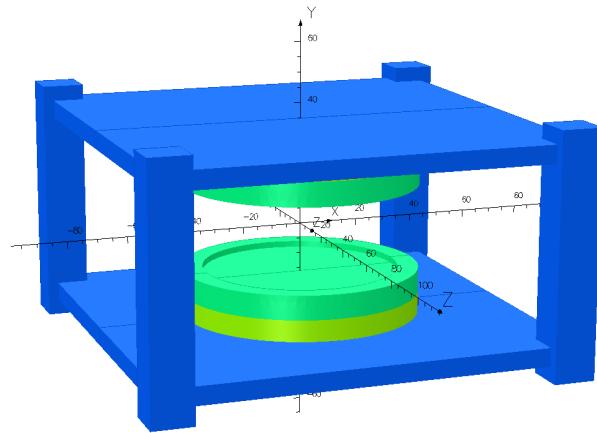
In this chapter, the most important concepts for pre-processing, analysis and post-processing of Opera-3d models are introduced. These are illustrated by a complete simulation of a Magnetic Resonance Imaging (MRI) magnet system. Included in the example are:

- building the model geometry using the Modeller;
- exploiting symmetry to reduce the size of the model to be solved;
- generating the finite element mesh;
- setting material characteristics;
- choosing the type of simulation and the solver module to be used;
- creating a model database and launching the simulation;
- using the Post-Processor to extract and process information from the solved database.

A model file, **mri.opc**, containing the example model used in this chapter, is provided in the examples folder provided with Opera. This can be accessed from the Opera Manager using **File -> Open Examples Folder**. Before it is used it should be copied to another folder; the examples folder is read-only. However, it is recommended that you follow the instructions below to build the model yourself.

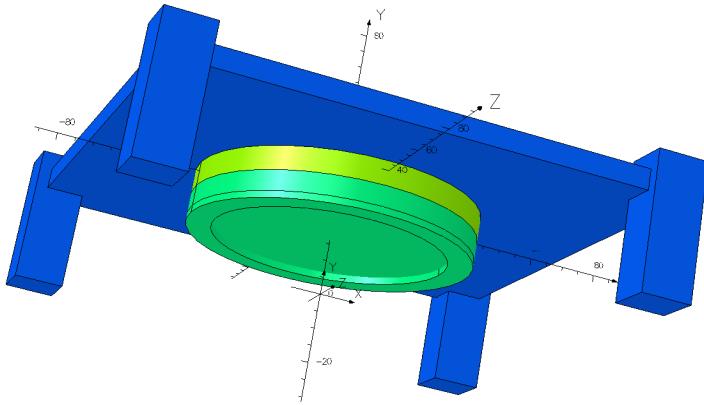
### **The Model**

The model to be created in this introduction is shown in [Figure 2.1](#). It represents a simplified permanent magnet MRI system. The frame is constructed from steel and acts as the return path for the flux. Attached to the frame are two cylindrical permanent magnets, magnetized to produce a uniform field in the region between the magnet poles. The poles are also cylindrical and include shims of the same material (rings added to the pole face) to improve the field quality. They are made from good quality steel.



*Figure 2.1 The MRI magnet geometry*

Because of the symmetry of the model, it is sufficient to solve the magnetic fields using only one quarter of the geometry of the MRI system. However, a half model of the geometry will be constructed initially as this is easier, both to build and to visualize. The required symmetry operations will then be implemented automatically as part of the model creation function of the Modeller. [Figure 2.2](#) shows the half model, which represents the upper pole and magnet and the top half of the frame. The symmetry of the one quarter model will be implied by appropriate boundary conditions. The model is constructed in cm<sup>1</sup>.



*Figure 2.2 The half model to be constructed in the Modeller*

---

<sup>1</sup>Various length units can be used: micron, mm, cm, m, inch.

This example is focussed on building and setting up the model for analysis.

This analysis will be nonlinear, requiring the user to specify magnetic characteristics (**BH** curves) for the magnet and the steel.

During post-processing, the magnetic flux density will be examined and the force between the poles calculated.

The model created is not optimized for field accuracy, and some additional control of the mesh should be made to improve the field solution, particularly in the air gap. General details on how to control the mesh further can be found in [Meshing in the Modeller \[page 541\]](#).

In MRI applications, the homogeneity of the dipole field is important. Calculating the homogeneity of the dipole field using Legendre Polynomial fitting is shown in [Legendre Polynomials \[page 645\]](#), which also discusses accurate field recovery.

## Starting the Opera Manager

The Opera Manager is the main interface to the different components within the Opera suite of programs.

### Microsoft Windows

On Microsoft Windows systems the Opera Manager can be started using the Start button on the Desktop as follows:

#### **Start -> All Programs -> Opera 2020**

Once started, the Opera Manager can be opened from the system notification area by clicking on the Opera icon as shown here: 

### Linux

On Linux systems, the Opera Manager is started from the menus using:

#### **Applications -> Other -> Opera 2020**

or from the command line by typing:

`$vfdir/bin/opa_manager`

where `$vfdir` is a shell variable set to the directory where the software is installed.

## Starting the Modeller

The Modeller can be started from the Opera Manager. However, before starting the Modeller it is usual to define a project folder, in which all files that are generated during the build, simulation and post-processing of a device will be stored. To do this, navigate to the required folder in the Opera

Manager and press the toolbar button ; alternatively, the project folder may be set from the folder view context menu, which is activated by clicking the right mouse button on the required folder name.

Having set the project folder select

### **Opera-3d -> Modeler**

from the Manager menubar, or by clicking on the **Modeler** icon  on the toolbar.

Full details of the features of the Opera Manager are given in the ***Opera Manager User Guide***.

## The Modeller Window

When the Modeller starts, the user is presented with a display showing the 3d axes (Figure 2.3). Note that the display will look slightly different on a Linux platform, but the functionality will be the same.

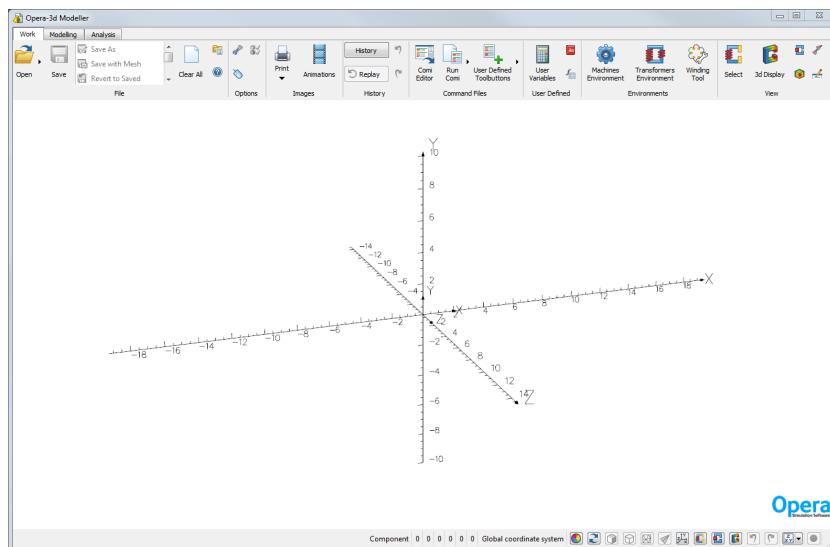


Figure 2.3 The initial display of the Modeller installed on a Microsoft Windows operating system

Many of the functions and features of the Modeller will be illustrated in this example; a more complete description may be found in **Geometric Modeller Features [page 73]**. The console window, showing the commands issued to the software, may be displayed by right clicking anywhere in the tab area. On the resulting context sensitive menu, select **Show/Hide** items and click on the tick box for **Console**.

## Building the Geometry

### The Permanent Magnet

The first part of the geometry to be created is the magnet. This is constructed from a single cylinder of 40 cm in radius and 7 cm in height. The lower circular face of the magnet, which will touch the pole, is 22 cm from the symmetry plane. The symmetry plane will be at Y = 0.

The Modeler allows entities to be built by sketching<sup>1</sup> - in which mouse click and drag operations are used to create the required shape. However, it will be a little easier to follow the model build process in this example if sketching is deactivated. Sketching may be toggled on or off by clicking on the



**Sketching Active** toolbutton. If the toolbutton is shown as depressed, i.e. it is high-lighted and surrounded by a shadowed frame, click on it to deactivate sketching.



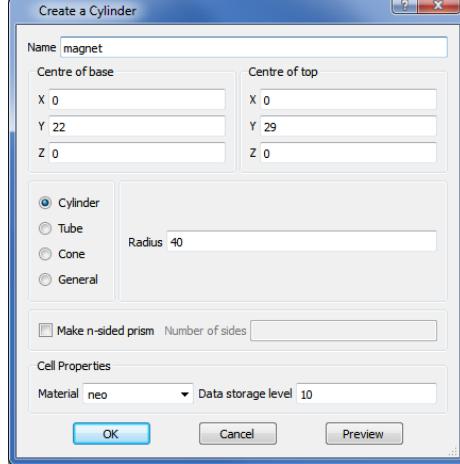
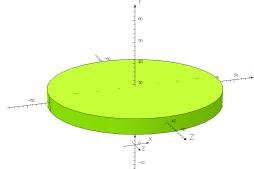
To create the cylinder, click on the **Cylinder** toolbutton on the toolbar. This brings up a dialog box in which the properties of the cylinder may be defined. Note that **Cylinder** is a generic name - the options in the dialog also allow construction of elliptic cylinders, tubes, cones and n-sided prisms.

To enter data in a dialog box, either select the field using the mouse, or move to the field using the **tab** key on the keyboard. Pressing the **Enter** key is equivalent to selecting the **OK** button. The user may close the dialog box without making any changes to the model by pressing **Cancel** (equivalent here to using the **Esc** key).

The cylinder is defined by completing the dialog box as shown below. Note that a **Material** name and **Data storage level**<sup>2</sup> may be entered here or edited later. In this example, we will call the material of the magnet **neo** and use a data storage level of **10** so that the properties of the magnet are preserved in subsequent operations. The **Preview** button allows the user to see a preview of the new cylinder. The data entry dialog is still active, allowing immediate modification of the cylinder parameters if required. Pressing **OK** or **Enter** creates the cylinder and closes the dialog. The cylinder is shown below, in the right-most column.

<sup>1</sup>Sketching is described in [Sketching Primitives \[page 79\]](#). The current state of **Sketching Active** is stored in the registry.

<sup>2</sup>**Data storage level** is used to determine properties when two or more bodies overlap. The properties from the entity with the highest data storage level are used.

 <b>Create Cylinder</b>	<p>Select the <b>Cylinder</b> option. Give the cylinder the <b>Name</b> <b>magnet</b>, and set: <b>Centre of base</b> <b>(0,22,0)</b>, <b>Centre of top</b> <b>(0,29,0)</b>, <b>Radius</b> <b>40</b>, <b>Material</b> name <b>neo</b> and <b>Data storage level</b> <b>10</b>.</p>  <p>Select <b>OK</b> to close the dialog.</p>	
---	---	---

## Controlling the View

The cylinder can be viewed from any direction by using the mouse. Move the mouse into the graphics window, hold down the left mouse button and move the mouse. This allows the view of the cylinder to be rotated. Now hold the shift and right button of the mouse down and move the mouse. This gives translation allowing the cylinder to be moved around the screen. If a three-button mouse is being used, holding the centre button allows the view to be zoomed in and out by up and down movements of the mouse. The above assumes that the mouse button operations are set to the default functions. The functions assigned to each button of the mouse may be changed by clicking on the **Mouse Buttons**

 toolbutton.

The view can be re-initialized by selecting the **Initial View** toolbutton from the toolbar . This will adjust the view so that all the visible<sup>1</sup> geometry can be seen. It can be useful if the view is modified in such a way that it is not easy to return to an overall view of the geometry.

Other toolbuttons on the toolbar also control the view. The geometry can be viewed along any of the major axis directions, using  etc. Toolbuttons  and  respectively toggle the solid and out-

---

<sup>1</sup>Here, "visible" is used to mean those entities selected for display. For more information, see [Geometric Modeler Features \[page 73\]](#).

line (edge) views of the model. Pressing the  toolbutton cycles visibility of the coordinate axes between all axes (major and triad), no axes, major axes, and the triad at the origin.

## Properties of the Magnet

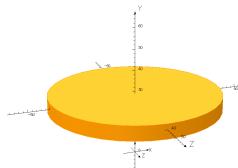
The properties of the cylinder which have been defined so far are:

Material `neo`<sup>a</sup> - which was defined when the cylinder was built

Potential type `Default`

Element type `Linear` - default setting

These and other cell properties can be edited as required. To do this, the cylinder must be selected as a **Cell** (for a discussion on **Cells** and **Bodies**, see [Bodies and Cells \[page 77\]](#)).

 <b>Pick Entity</b>   <b>Pick Cells</b>	<p>Ensure pick entity and pick cells are both selected. Move the mouse (without pressing the buttons) so that the cursor lies over the cylinder. The outline of the cylinder will be high-lighted. Then either double-click or right-click and the solid cylinder will be high-lighted, indicating that it has been picked.</p> <p>After double-click an operation may be selected from the menu bar; after right-click, the program displays the context menu.<sup>b</sup></p> <p>Note that a right-click only allows a single entity to be picked at a time. If multiple entities are to be picked, a double-click should be performed on each.</p>	
---	---	---

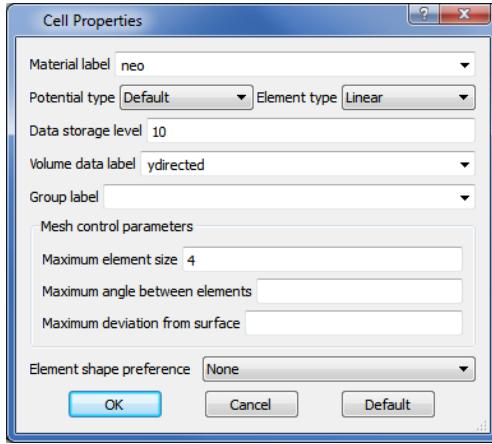
The properties of the picked cell can then be set. From the top menubar or from the context menu as appropriate.

<sup>a</sup>For this example, the magnet has been given the material name `neo`. This is only a label but as yet the material properties have not been defined, and remain at the default values of  $\mu = \mu_0$ ,  $\epsilon = \epsilon_0$ ,  $\sigma = 0$ . Redefinition of the physical material properties will be described later in this example.

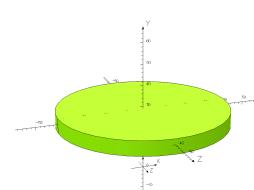
<sup>b</sup>Use of the right mouse button to activate the context menu assumes that the functions of the mouse buttons are set to their default values. Use the **Mouse Buttons**  option to change the functions assigned to each button.

**Cell Properties**

Set **Volume data label** to **ydirected**, **Maximum element size** to 4 and check that the **Data storage level** is 10.



Select **OK** to close the dialog and apply the properties.



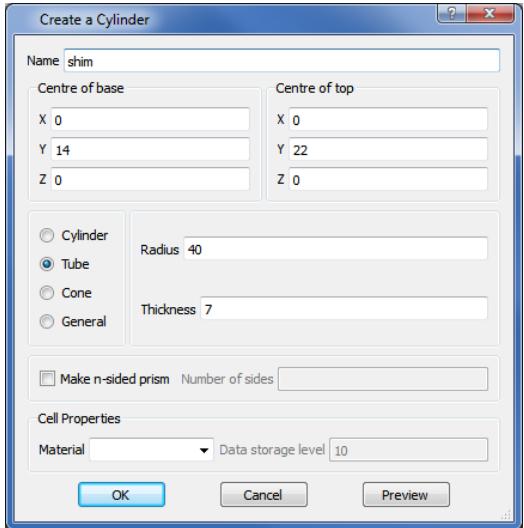
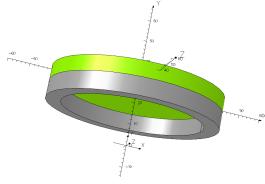
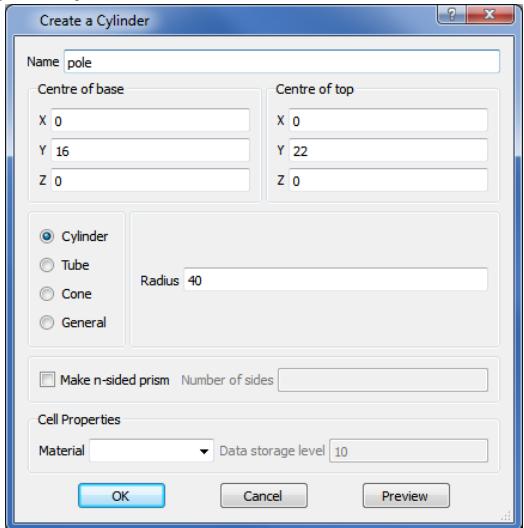
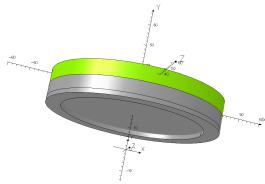
The **Volume data label**, set here to **ydirected**, will be used later to specify the easy direction of magnetization for the magnet. The **Maximum element size** (4) specifies that no mesh element in the magnet will be larger than 4 cm along any of its edges. Press **Enter** or the **OK** button to apply these properties.

## Building the Pole and Shim

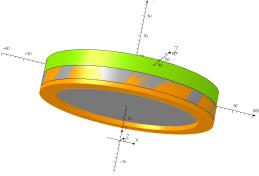
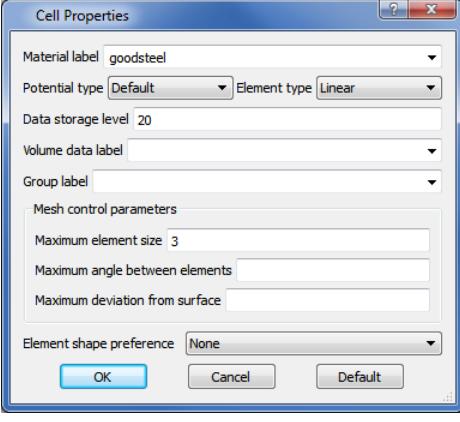
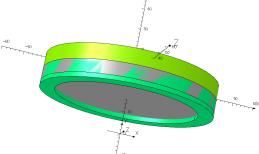
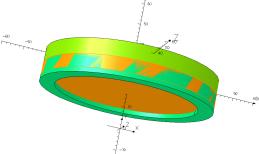
The pole and shim are constructed using two cylindrical bodies. The first body is a cylindrical disk that is used to represent the geometry of a 7 cm wide, 2 cm deep shim. The second is a disk representing the main bulk of the magnet pole.

When generating the model body for analysis, these 2 bodies are combined using a union operation without regularization. In this way, internal faces are kept, allowing the pole face to be represented using 3 cells. Although not necessary for modelling the geometry, by having this structure we have more control over the mesh. In this case, we can generate a regular mesh of hexahedral elements within the shim region, giving a more accurate solution.

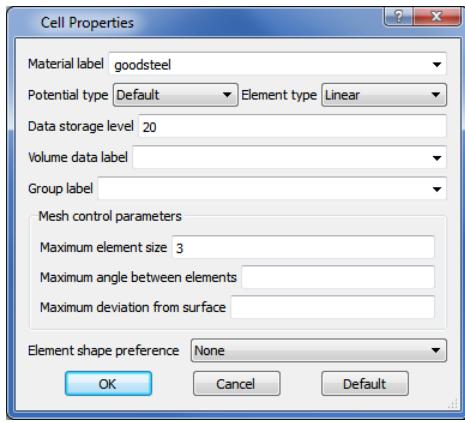
The **Material** name for the pole and shim will be defined later, so the **Material** field has been left blank in both of the two following dialogs.

 <b>Create Cylinder</b>	<p>Set <b>Name</b> to be <b>shim</b>, with <b>Centre of base</b> at (0,14,0), the <b>Centre of top</b> at (0,22,0). Select <b>Tube</b>, with a <b>Radius</b> 40 and <b>Thickness</b> 7. Note that when a tube is being defined, <b>Radius</b> is the outer radius of the tube.</p>  <p>Select <b>OK</b> to close the dialog.</p>	
 <b>Create Cylinder</b>	<p>Set <b>Name</b> to be <b>pole</b>, move <b>Centre of base</b> to (0,16,0) and leave <b>Centre of top</b> at (0,22,0).</p>  <p>Select <b>OK</b> to close the dialog.</p>	

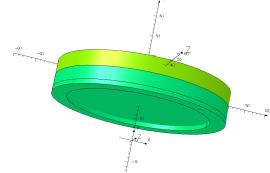
The properties of the pole and shim volumes can now be defined.

 <b>Pick Cells</b>	<p>High-light the shim volume by placing the mouse cursor over it, and right-click to pick the cell and display the context menu in a single operation.</p> <p>Note that because the volumes overlap, the display of the picked cell may show some of those the faces of the unpicked body that are at the same location.</p>	
<p>From the context menu, select: <b>Cell Properties</b></p>		
	<p>Set <b>Material label</b> to <b>goodsteel</b>, <b>Data storage level</b> to 20, and <b>Maximum element size</b> to 3.</p>  <p>Select <b>OK</b> to close the dialog.</p>	
 <b>Pick Cells</b>	<p>High-light the pole volume and right-click to pick the cell and display the context menu.</p>	
<p>From the context menu, select: <b>Cell Properties</b></p>		

**Set Material label to goodsteel, Data storage level to 20, and Maximum element size to 3.**



Select **OK** to close the dialog.

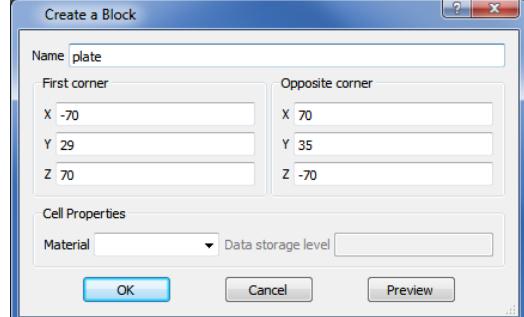


## Building the Frame

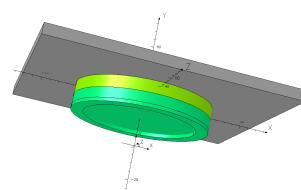
The frame is constructed from two blocks - one to make the back plate and the second to make one of the (half) legs. The leg block is then copied to make four (half) legs. The back plate will be constructed as a block using two diagonally opposite corners.

**Create Block**

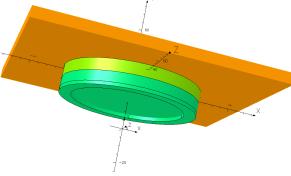
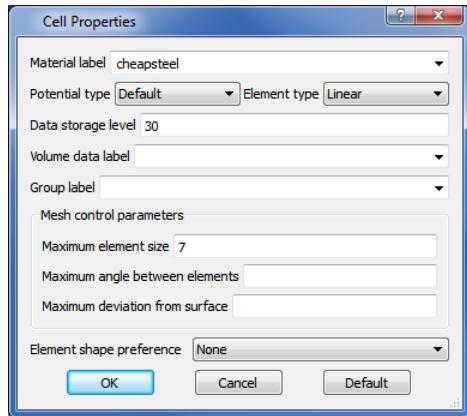
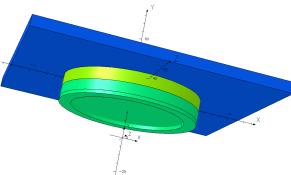
**Set Name to be plate, with First corner at (-70, 29, 70), the Opposite corner at (70, 35, -70).**



Select **OK** to close the dialog.



The cell properties can then be defined.

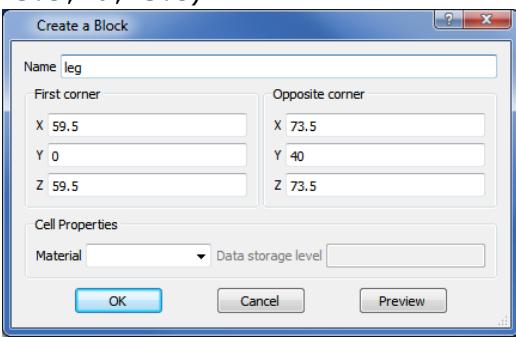
 <b>Pick Cells</b>	<p>High-light the block and right-click to pick the cell and display the context menu.</p>	
<p>From the context menu, select: <b>Cell Properties</b></p>		
	<p>Set <b>Material label</b> to <b>cheapsteel</b>, <b>Data storage level</b> to 30, and <b>Maximum element size</b> to 7.</p>  <p>Select <b>OK</b> to close the dialog.</p>	

The **Data storage level** is set higher than the default (0). When the plate and the legs are combined later to make a single volume using a Boolean union operation, with regularization, the cell with the highest data storage level will define the properties that the combined body retains.

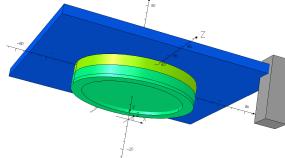
The first (half) leg is also defined as a block.

 **Create Block**

Set **Name** to be **leg**, with **First corner** at (59.5, 0, 59.5), the **Opposite corner** at (73.5, 40, 73.5).



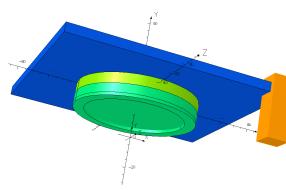
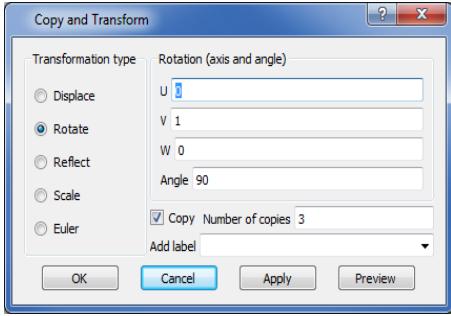
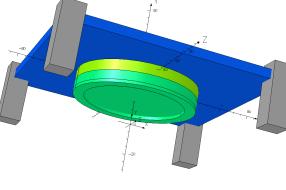
Select **OK** to close the dialog.



Note that objects can be hidden from the display (by first selecting the **Hide Entity** toolbar button  or picked for a subsequent operation (using the **Pick Entity** toolbar button ). See [Picking and Hiding Entities \[page 94\]](#) for more information.

## Making copies

Three copies of the leg will be made by rotation around the y-axis. Copying is an **Operation** that can be performed on cells or bodies.

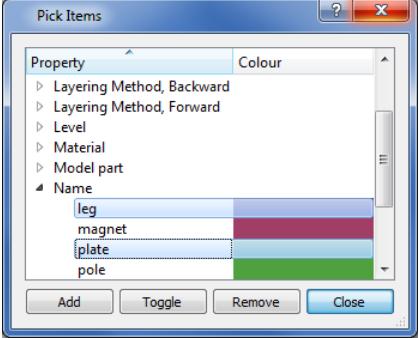
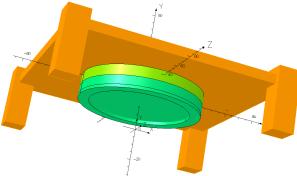
 <b>Pick Bodies</b>	<p>Pick the leg using a right-click on the mouse.</p>	
 <b>Copy and Transform</b>	<p>Select the <b>Rotate</b> option, enter a <b>Rotation</b> axis of <math>(0, 1, 0)</math>, and <b>Angle</b> of 90 degrees. Tick the <b>Copy</b> option and set the <b>Number of copies</b> to 3.</p> <p></p> <p>Select <b>OK</b> to close the dialog.</p>	

Note that the axes are specified in local coordinates, labelled **U**, **V** and **W**. By default, the local coordinate system is aligned with the global system, and has not been changed in this model. Hence a rotation about **V** is equivalent to a rotation about the global y axis.

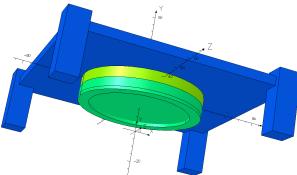
## Combining bodies

The back plate and the legs are to be combined into a single volume using a Boolean union operation. This requires several bodies to be picked. While this may be performed by mouse clicks on each body, it is sometimes more convenient to pick by one or more of their attributes. This may be

achieved by using  **Pick by Property**.

 <b>Pick by Property</b>	<p>Expand the <b>Name</b> category to view the list of names of bodies in the model.. Click on <b>leg</b>. Then press and hold the <b>Ctrl</b> key while clicking on <b>plate</b>. Click on <b>Add</b>, followed by <b>Close</b>.</p> 	
--	---	---

Perform the **Union** operation using **Combine Bodies**, which may be accessed from the **Operations** menu or from the context menu.

<b>Union, with regularization</b>	<p>Note that the legs turn to the same colour as the back plate, indicating they have used the data from the <b>plate</b> cell as it had a higher data storage level. Other results from using <b>regularization</b> are discussed in more detail in <a href="#">Boolean Operations [page 104]</a>.</p>	
-----------------------------------	---	--

This completes the geometry of the model.

## Surrounding Air and Boundary Conditions

The previous steps have defined the geometry of the magnet. However, the magnetic field also exists in the free space that surrounds the magnet, and this must also be included as part of the model. In the real world, this space extends to infinity, but in a finite element model, the free-space region must be terminated at a finite distance from the geometry. This distance must be such that the termination does not significantly affect the accuracy of the solution.

The free-space background region will be terminated about 180 cm from the centre of the magnet in the horizontal plane (X and Z directions) and about 120 cm in the vertical (Y) direction. In this example, the background region will be generated as part of the process of defining the symmetry of the model.

### Mesh Control Cylinder

Additional cells can be included in a model to give better control of the element sizes in areas of interest. In this model, an additional **Cylinder** of air should be defined under the pole to reduce the element size here. Use the same toolbuttons and dialogs as before to define a cylinder with **Material** air with **Centres of base** and **top** at (0,0,0) and (0,14,0) and a **Radius** of 40.

Pick the cylinder and change its **Cell properties** so that the **Maximum element size** is 10.

### Analysis Type

The operations that have been performed up to this point, i.e. definition of the model geometry, are independent of the type of analysis that is to be performed. However, some of the remaining tasks in this example will use features that differ between analysis types, so before continuing, set this to be **Magnetostatic**. This affects the items available on the **Analysis** tab, and also what options are available in various dialogs.

Note that the analysis type may be selected or changed at any time during the model creation process. However, if performed after, for example, model symmetry, material properties and boundary conditions have been defined, then these may then need to be modified to suit the selected analysis type.

### Exploiting Symmetry

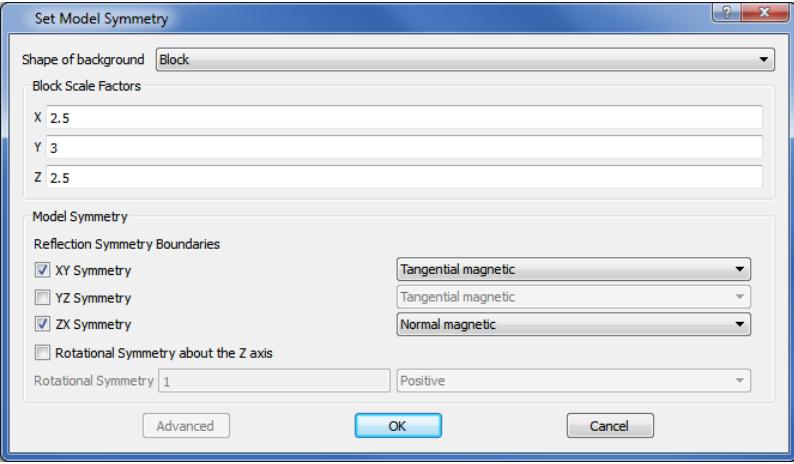
As explained at the beginning of this example, the symmetry of the model means that only 1/4 of the complete magnet needs to be solved. The 1/4 model will be created from the geometry that has been defined by using **Model Symmetry**; this will also allow the boundary conditions on the outer faces of the model to be defined.

**Model Symmetry** also allows a background volume to be created automatically. Geometry that lies within the background volume takes precedence over the background, i.e. the background volume fills only the parts of the space that are not already occupied.

**Model Symmetry**



Set the **Shape of background** to be **Block**, and set the **Scale Factors** to be 2 . 5 in X and Z, and 3 in Y.  
 Set the **Reflection Symmetry Boundaries** to be **Tangential magnetic** on the XY plane, and **Normal magnetic** on the ZX plane.



Select **OK**. Note that nothing changes on the display until the Model Body is created later.

The use of **Block** as the **Shape of background** will create a rectangular block around the geometry that is centred on the centre of the bounding box of the geometry, and is **Scale Factor** times larger than the bounding box in each direction. The block is then trimmed by the **Reflection Symmetry Boundaries** so, in this case, it only occupies the quadrant formed by positive y and z.

The above has set the required electromagnetic boundary conditions on the symmetry planes. The boundary conditions on the outer boundaries are left as default<sup>1</sup> in this example.

## Saving the Model

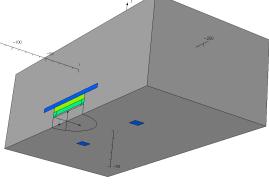
The model can be saved to a file at any time by selecting **Save**  or typing **Ctrl-S**. A name can be given in the browser window that follows. The model can be subsequently saved at any other time without giving the file name again.

---

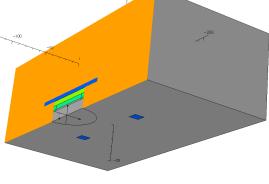
<sup>1</sup>For the Magnetostatic solver the default boundary condition on the outer boundaries is Tangential magnetic.

## Creating the Model Body

While the geometry of the model is being defined, the model is in **Component** mode. In this mode, any feature of the model may be modified, and any number of bodies can exist in the model. Before the finite element mesh can be generated, the model, including the background region, must be made into a single body, with the properties of overlapping regions cells, and the defined symmetry conditions must be applied. These operations are performed during the **Create Model Body** process (see also [Analysis of Parts of a Geometric Model \[page 565\]](#)). This operation is fully reversible, and **Component** mode may be restored by selecting **Delete Model Body** in the **Model** menu.

 <b>Create Model Body</b>	<p>The properties of the background volume have taken effect, with only the 1/4 model now being displayed. The geometry of the complete half MRI system, as built, is still held in the model but will not be used. Selecting <b>Delete Model Body</b> from the <b>Model</b> menu will revert to <b>Component</b> mode.</p>	
---	---	---

Where the faces of the MRI geometry and background region are coincidental, the faces of the geometry are displayed rather than the faces of the background. However, the faces of the MRI geometry have inherited the boundary condition labels applied to the background volume. This can be verified by examining the properties of the faces on the symmetry planes.

Right mouse click to view context menu, select: <b>List</b>		
 <b>Pick Faces</b>	<p>Right mouse click on any face on the symmetry planes to view the context menu. Selecting <b>List</b> shows that the selected face is labelled as a symmetry plane. Note that the faces of the background volume on the symmetry planes are no longer simple rectangles or squares, but fit round the MRI geometry.</p>	

## Building the Finite Element Mesh

Creation of the finite element mesh is performed in two stages; a surface mesh is produced first, followed by a volume mesh.

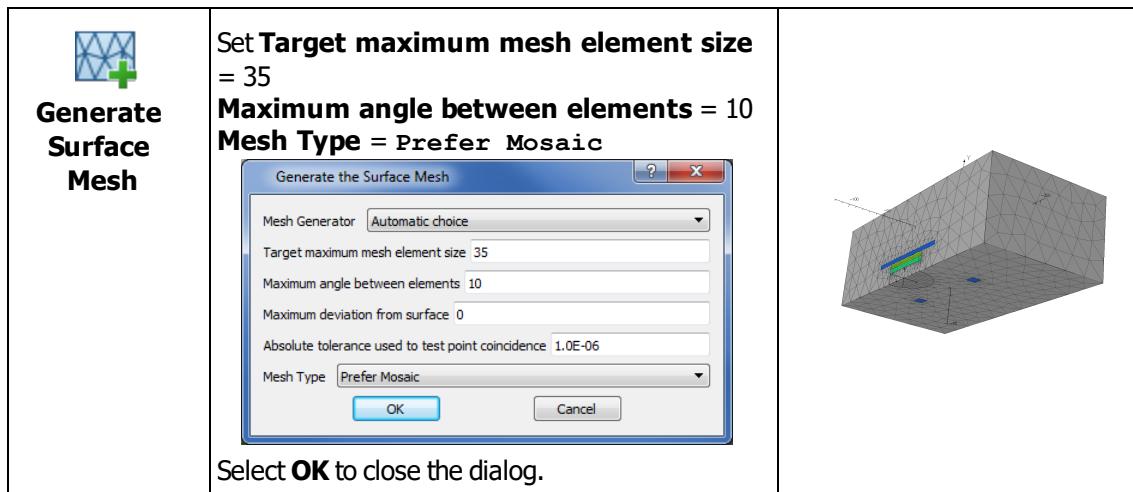
The surfaces of the model are initially meshed with triangles or quadrilateral facets. This includes internal surfaces between cells of the model. After the surface mesh is complete, the volume mesh can be created.

Where the geometry of a cell is regular, it is possible to create a regular mesh of hexahedral elements or of triangular prism elements. In this present example, the cells of the shim and pole are such that they can all be regularly meshed. However, the background region has an irregular shape, and will be meshed with tetrahedra.

The use of the mixed element shapes within a single mesh is called "mosaic" meshing. See [Meshing in the Modeller \[page 541\]](#).

### Surface Meshing

The parameters of the surface mesh are selected to suit the geometry of the model.



The **Target maximum mesh element size** will be applied to those regions where a smaller mesh size has not been defined. In this model, the maximum mesh size in the plate and legs was set to 7 cm, while that in the magnet was set to 4 cm, and in the pole and shim to 3 cm. The concentration of surface elements on these parts is a consequence of these settings. Note that there is a gradual transition between the regions of fine and coarse meshing. However, it may be noted that in some regions, the mesh appears to be relatively large compared with the local geometry. This point is commented on at the end of the present example.

The **Maximum angle between elements** affects the mesh size on curved surfaces. If the angle between the normals at the centroids of adjacent elements exceeds this value, the element size will be reduced.

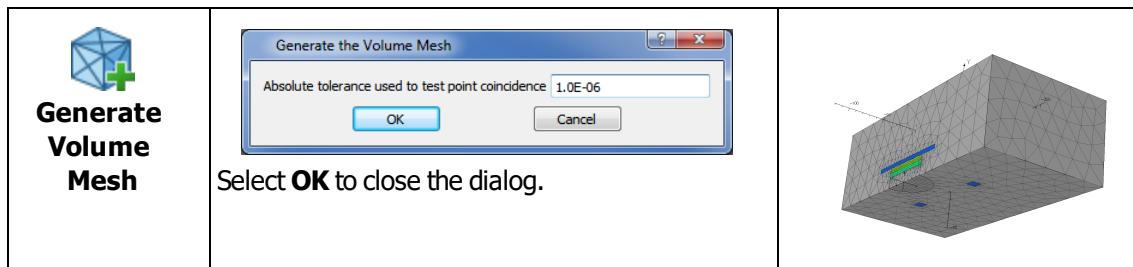
Setting **Mesh Type** to **PREFER MOSAIC** will result in a hexahedral or prismatic mesh in cells where it is possible, and where tetrahedral meshing has not been specified.

Generation of the surface mesh should only take a few seconds. More complex models may take longer to mesh. In such cases, progress can be monitored in the bottom status bar. If necessary, The

meshing process can be aborted by hitting the **Cancel** toolbutton .

## Volume Meshing

Once the surface mesh is complete, the volume mesh can be generated.



As each cell is meshed it is high-lighted by its outline. Constructing the volume mesh normally takes somewhat longer than the surface mesh; for this model volume meshing should take about one minute, depending on computer performance. The resulting mesh should contain about 80,000 elements and about 15,000 nodes - the exact numbers will depend on the computer hardware in use. This mesh data can be obtained by hovering the cursor over the text, **Volume mesh**, in the information bar at the bottom-right of the Modeller window, until a message window appears.

Note that more nodes will be added to the mesh when it is written to the analysis database because quadratic elements will be used to model the curved surfaces. The total number of nodes in the analysis will be about 33,000.

## Defining Material Properties

Up to this point, only material labels have been defined. Material properties will now be assigned to these labels. It should be noted that material properties can be defined at any stage during the model build process, after the analysis type had been chosen.

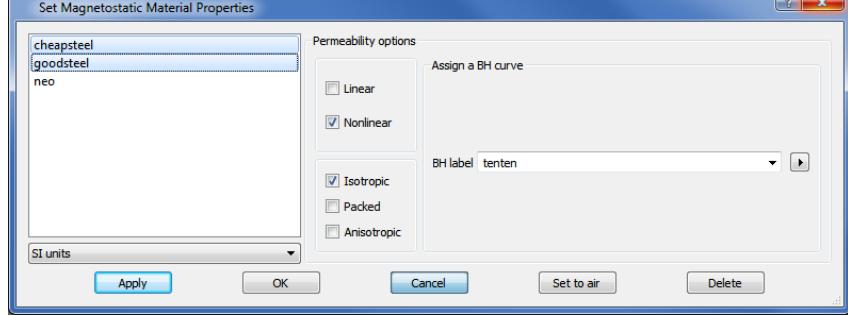
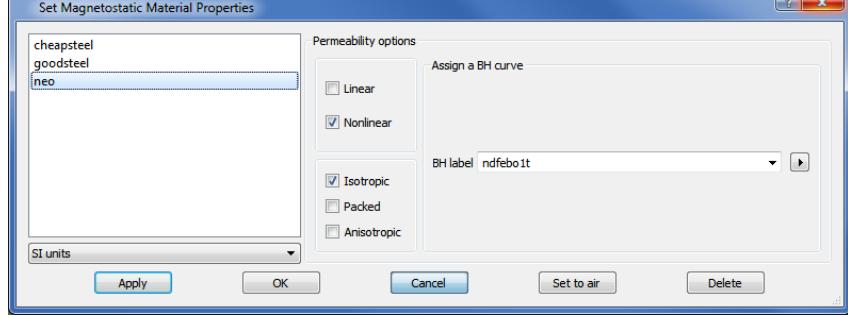
In this example both `goodstee1` and `cheapstee1` will be given the same magnetic characteristics. The permeability of the magnetic materials will be defined using a nonlinear characteristic curve and a nonlinear analysis will be performed. In the analysis, the characteristic curve for the material data will be used to match the permeability of each element to the flux density. Note that a linear analysis could also be performed, in which case, the initial slope of the curve would be used to define the material permeability.

In this example, the materials will be treated as isotropic.



Material properties are assigned to material labels defined for each cell. Using **Material Properties**, each material label may be assigned properties, which may include a nonlinear BH curve. Each BH curve is defined with its own label, and this label is assigned to the material label.

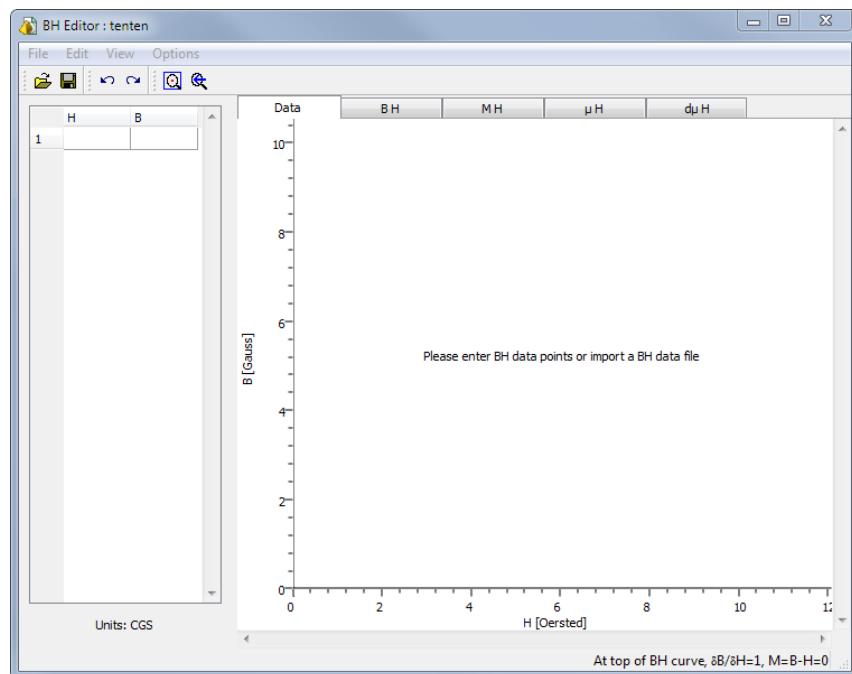
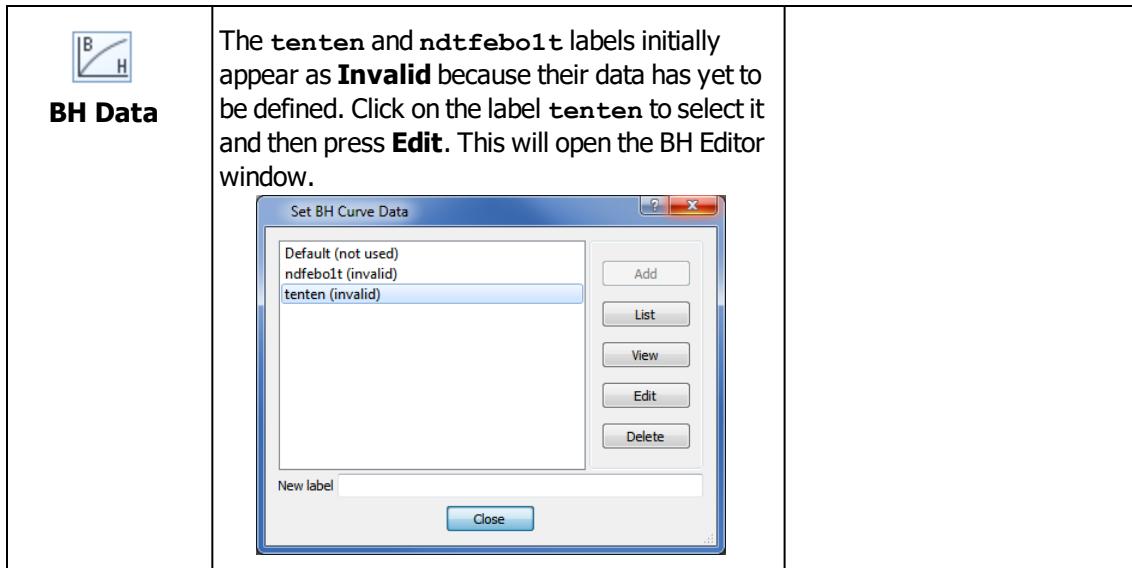
The material `cheapstee1` is assigned the BH curve label `tenten`. The BH curve itself will be defined later. Similarly, `goodstee1` is assigned the label `tenten`, and `neo` is assigned the label `ndfebo1t`

 <b>Material Properties</b>	<p>Select both <b>cheapsteel</b> and <b>goodsteel</b> together by holding down the <b>Shift</b> key while clicking on them. Select <b>Nonlinear</b>, and give a BH label <b>tenten</b>.</p>  <p>Save this selection using the <b>Apply</b> button</p>
	<p>Select <b>neo</b>, select <b>Nonlinear</b>, and give the BH label <b>ndfebo1t</b></p>  <p>Click on <b>OK</b> to save the selections and leave the dialog.</p>

## Set BH curve properties

Initially the BH curve labels will be undefined. It is necessary to assign BH curves to each BH label

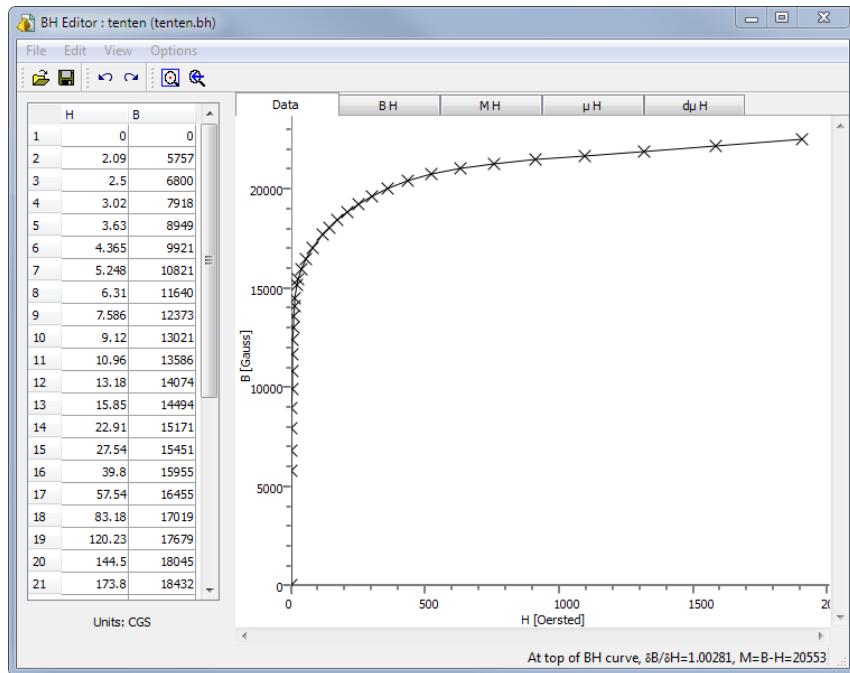
being used in the model. The list of labels currently being used can be obtained using the  **BH Data** toolbutton.



A user-defined BH curve can then be created by typing data into the data section in the left panel. A minimum of five (**H**, **B**) data points is required to create a valid curve. The values for **H** and **B** are entered in the units system indicated at the bottom of the pane. The BH Editor **Options** menu may be used to switch between CGS (gauss and oersted) and SI (tesla and amp m<sup>-1</sup>) units.

The two columns used for data entry have some of the characteristics of common spreadsheet software: data can be copied to and from spreadsheets; if the columns are too narrow, the values will appear as # characters.

Alternatively, an existing BH curve can be used by loading the appropriate file, a number of which are supplied with Opera. Use the BH Editor **Import File** toolbutton, , browse to the file, located in the installation directory **%VFDIR% \bh \tenten.bh**, and select **Open**.



To help in checking the quality of the BH curve, other tabs in the BH curve editor allow the data to be viewed in different ways. For more information refer to the **Opera-3d Reference Manual**.

Select **File -> Close** to close the BH Editor.

Repeat the above process to specify a BH curve for the label **ndfebo1t**, loading the file **%VFDIR% \bh \ndfebo1t.bh**. Then **Close** the **BH Data** dialog.

## Set volume properties

The next action is to specify the magnetization direction for the permanent magnet. The default direction is along Z, but in this model the magnet is to be magnetized through its thickness i.e. in the global Y direction. The desired magnetization direction may be achieved by setting the properties of the **Volume data label**, **ydirected**, that was assigned when the magnet was created.

In general, these labels are used to associate various physical properties - velocity, charge density etc. - with a particular volume or volumes of the mesh. For a Magnetostatic analysis, the **Volume**

**data label** may be used to reorientate the local volume properties with respect to the global coordinate system (and to define a packing factor when simulating laminated materials).

The properties of the **ydirected** label may be set using  **Volume Properties**. The orientation of the local coordinate system is controlled by Euler angles, **Theta**, **Phi** and **Psi**, as in the dialog below. Commonly used local orientations may also be selected from the **Local orientation** drop-down menu. In this case, the local Z coordinate of the magnet volume data - i.e. the magnetization direction - needs to be aligned with the global Y direction. Hence we can use one of the menu options, namely, **Local XYZ = global ZXY**.

 <b>Volume Properties</b>	<p>Click on the <b>ydirected</b> label and choose <b>Local XYZ = global ZXY</b> from the <b>Local orientation</b> drop-down menu. This is equivalent to setting <b>Theta=90</b>, <b>Phi=90</b>, <b>Psi=180</b>.</p> <div style="border: 1px solid #ccc; padding: 5px; width: fit-content; margin-left: auto; margin-right: auto;"><p>Set Magnetostatic Volume Properties</p><table border="1" style="width: 100%; border-collapse: collapse;"><tr><td style="width: 30%;">ydirected</td><td style="width: 70%;"><p>Volume orientation</p><p>Local orientation <b>Local XYZ = global ZXY</b></p><p>Other orientation angles</p><p>Theta 90</p><p>Phi 90</p><p>Psi 180</p><p>Electromagnetic properties</p><p>Packing factor</p></td></tr><tr><td colspan="2" style="text-align: center; padding-top: 5px;">OK Cancel Apply Clear Delete</td></tr></table></div> <p>Click on <b>OK</b> to specify the orientation and leave the dialog.</p> <p>Note that the orientation can be made visible by switching on the display of the volume vectors by selecting the  <b>Vectors</b> button. Select the <b>Volume orientation</b> radio button and, for this example, set the <b>Scale factor</b> to 10.</p>	ydirected	<p>Volume orientation</p> <p>Local orientation <b>Local XYZ = global ZXY</b></p> <p>Other orientation angles</p> <p>Theta 90</p> <p>Phi 90</p> <p>Psi 180</p> <p>Electromagnetic properties</p> <p>Packing factor</p>	OK Cancel Apply Clear Delete	
ydirected	<p>Volume orientation</p> <p>Local orientation <b>Local XYZ = global ZXY</b></p> <p>Other orientation angles</p> <p>Theta 90</p> <p>Phi 90</p> <p>Psi 180</p> <p>Electromagnetic properties</p> <p>Packing factor</p>				
OK Cancel Apply Clear Delete					

## Creating the Analysis Database

There are two steps to creating the analysis database. First, the user sets appropriate analysis information for the type of analysis program used to solve the field equations. The second step is to create the database itself.

### Analysis settings

The **Analysis Settings** window allows the user to choose a linear or nonlinear analysis, the number of nonlinear iterations, the type of nonlinear algorithm and other analysis specific information.

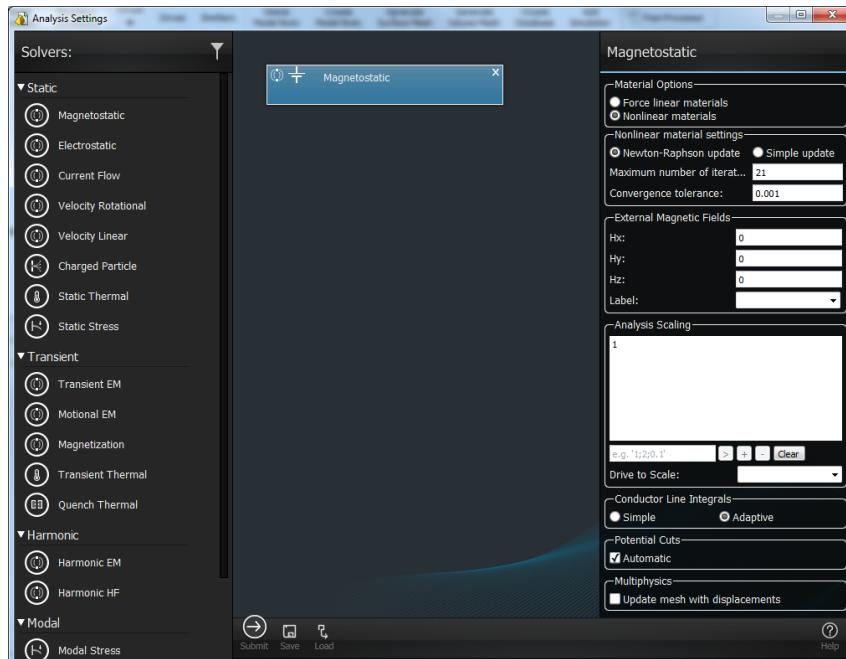


Figure 2.4 Analysis settings dialog for Magnetostatics

Select **Use nonlinear properties**, and leave other settings at their default values. Click on **Submit**.

Note that the analysis settings may be specified at any time in the model build process, up to the point where the database is created.

### Database



The second step is to create the analysis database. Select **Create Database** to display the dialog.

In this example, the database is created after the surface and volume meshes have been created. Consequently, the action offered as default in the **Operations** field is **Create database**. With this selected, the options to define the **Surface mesh parameters** and **Volume mesh parameters** are unavailable. Had the **Create Database** dialog been opened before the model had been created or meshed, the default action would have been to perform the model body creation, meshing and database creation in a single step. The mesh parameters would then have been editable.

The name for the database is entered in the **Database** field. Browsing is available by clicking on the button at the right of the filename box. In this case, the name **mri** is entered; the extension, **op3** does not need to be included because database files are given the extension **op3** automatically. A Modeller data file of the same name, but with extension **opc**, is created with the **op3** file.

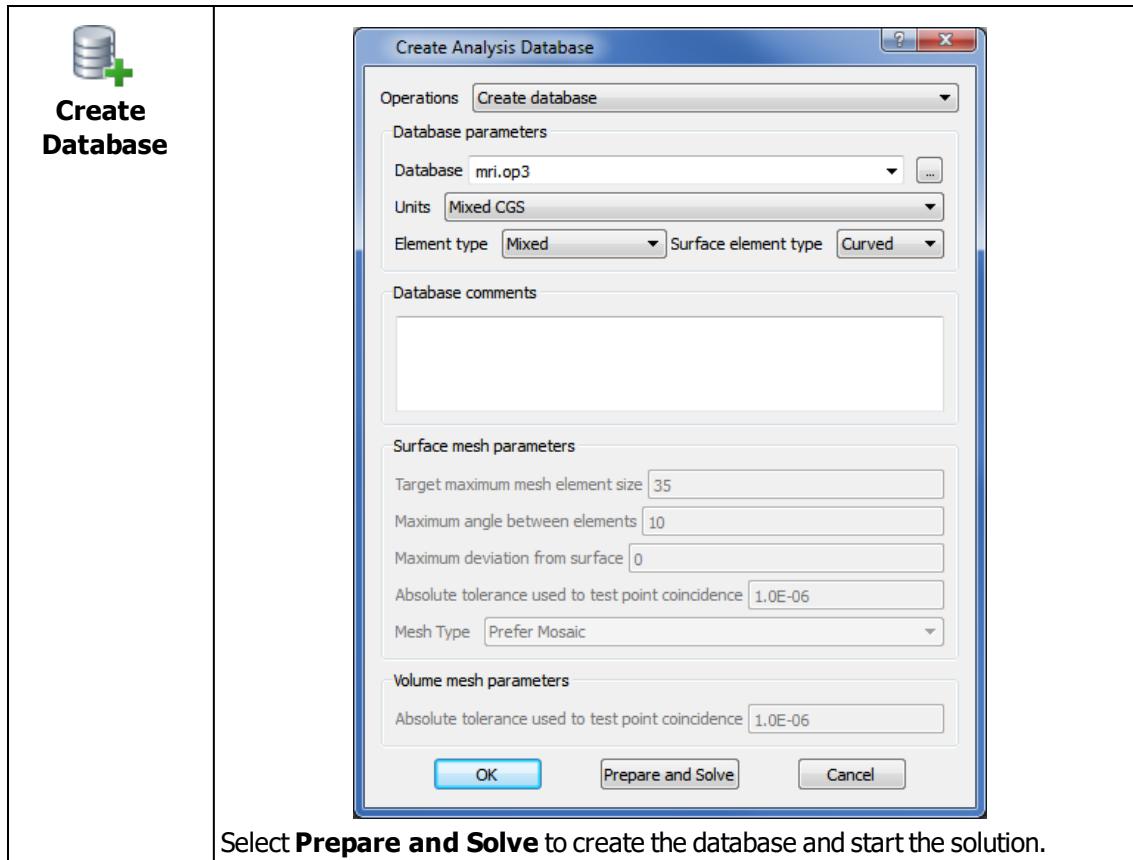
The **Units** in which the model has been constructed are selected at this time. By default, the units are **Mixed CGS**.

When a model is built, the type of mesh element in each cell and on each face in the model may be defined as Linear or Quadratic. When meshed, regions with hexahedral mesh elements will be meshed with eight-noded hexahedra if the element type is linear, and with twenty-noded hexahedra if the element type is quadratic. Similarly, linear tetrahedral elements will have four-noded tetrahedra, while quadratic elements will have ten nodes. When creating the database, **Element type** and **Surface element type** allow these setting to be preserved (**Mixed**) or overridden globally (**Linear** or **Quadratic**). Selecting **Curved** for the **Surface element type** is similar to **Mixed**, but those elements on curved surfaces default to quadratic.

Note that in this description, mesh **Element type** (linear or quadratic) should not be confused with the mesh **Element shape** (hexahedral or tetrahedral). Similarly, the use of linear to describe an element type should not be confused with its use to describe linear material properties.

The user may enter a multi-line description of the contents of the database in the **Database comments** field.

In common with other dialogs in Opera, pressing **F1** when the dialog is in focus will display the dialog help information. This provides additional information on the database parameters.



This will create the **op3** database and submit it to the Opera Manager batch queue to be solved.

It is sometimes convenient to create a database but not solve it immediately. This is achieved by selecting **OK** rather than **Prepare and Solve**. The database may then be solved by launching the simulation from the Opera Manager.

The Modeller can now be closed.

# Running the Analysis

---

## Opera Manager

The analysis of the **op3** database is controlled through the Opera manager batch queue, more details of which can be found in the **Opera Manager User Guide**.

If **Prepare and Solve** is selected when the database is created, the database will be loaded in the Opera Manager batch queue for solution. The solution will begin once the task reaches the top of the queue.

If **OK** is selected, which prepares but does not solve, the **op3** database can be put into the batch queue from within the Opera Manager. To start the analysis, either right-click on the file (in this case **mri.op3**) and select **Solve** from the menu, or select and drag the database into the batch pane<sup>1</sup>.

## Monitoring progress

Once started, a solver window will report progress of the Magnetostatic solution.

The window reports when major stages of the analysis are complete and the amount of CPU and elapsed time used at each stage.

To obtain a nonlinear solution, as required in this example, an iterative method is used. Consequently, the same stages are repeated until convergence of the solution is obtained.

The solver window displays important details from the previous iteration, allowing the progress towards convergence to be monitored. If convergence is not achieved at a particular iteration, a new set of equations will be formed by updating the permeability distribution from the previous solution. It then solves the new equations and determines the over (or under) relaxation factor that will minimize the residual. Finally, it determines the convergence indicators (the residual error in the solution to the set of equations, the *rms* relative change in solution over the whole model and the maximum relative change in the solution on any degree of freedom in the model) to determine whether the solution has converged.

With the default nonlinear convergence tolerance of 1E-3 this model should converge in less than 10 iterations. A summary message of the total CPU and elapsed time is reported at the end of the analysis, after which post-processing can begin.

The displayed information is also written to a file with the same name as the database, but with the extension **res**.

---

<sup>1</sup>By default, the Opera Manager batch queue will automatically run one job at a time. This can be changed using the **Options -> Batch Options** menu in the Opera Manager.

## Finish the analysis

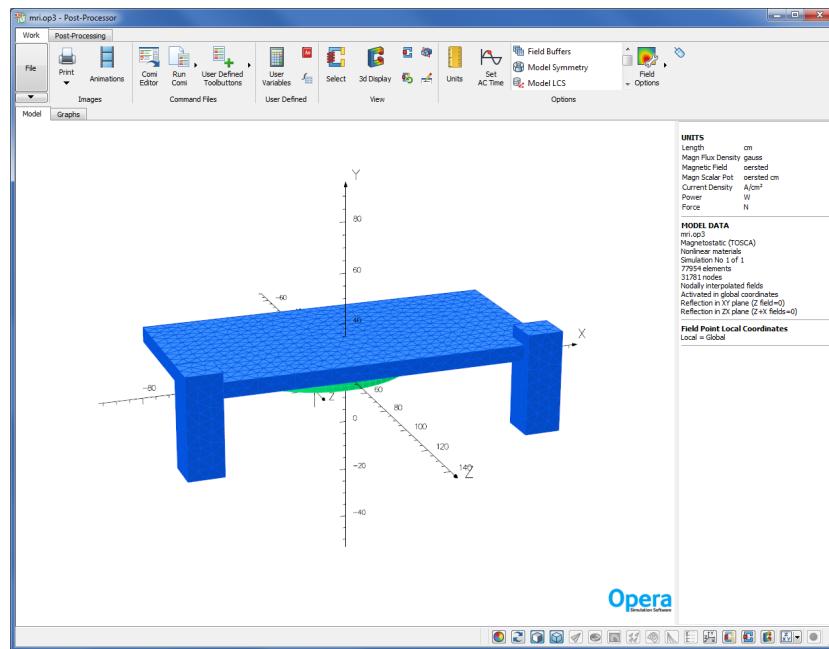
When the analysis has finished, select the **Post-Process** option to close the solver window, and start the Post-Processor.

## Post-Processing

The post-processing features shown here are typical of those that may be appropriate when assessing the performance of a magnet system. Many other features are available in the Post-Processor, and these are described in other worked examples in this **User Guide**.

### Loading the Solution

When the Post-Processor is started from the solver window **Post-Process** button, the database is automatically activated and loaded, and the display shown in [Figure 2.5](#) is obtained.



*Figure 2.5 Initial Post-Processor window*

The post-processing features are accessed from the tabbed menu in the same way as in the Modeler. The toolbuttons that have similar functions in the two programs also have similar icons. The console window, showing the commands issued to the software, may be displayed by right clicking anywhere in the tab area. On the resulting context sensitive menu, select **Show/Hide** items and click on the tick box for **Console**.

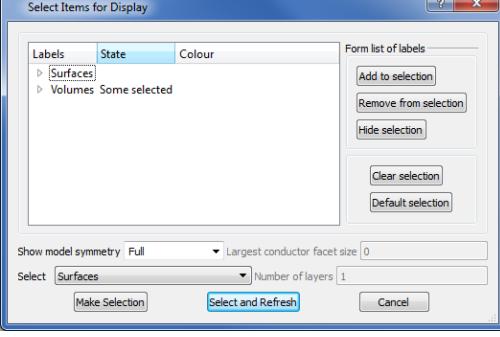
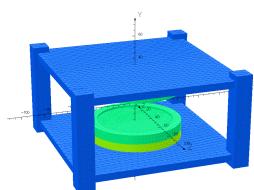
### Displaying the Model with Replications

For computational efficiency, only 1/4 of the MRI and the surrounding free space was solved in Magnetostatic. The Post-Processor allows the user to recreate the model of the complete magnet. Some

of the post-processing calculations need the complete magnet to be selected for display.

The symmetry for the model was set within the Modeller, and the Post-Processor has access to this information. However, symmetry copies of the geometry are not initially displayed. To show the full

geometry, use the  **Select** toolbutton .

 <b>Select</b>	<p>Select <b>Full</b> from the <b>Show model symmetry</b> drop-down list.</p>  <p>Click on <b>Select and Refresh</b> to close the dialog and update the display.</p>	
--	---	---

The operation of the 3d-Viewer in the Post-Processor has the same functionality as in the Modeller. For example, for some models, the view of the displayed geometry may not be convenient, in which case the  **Initial View** toolbutton may be used to centre the view on the centre of the bounding box of the selected geometry, and apply an appropriate zoom.

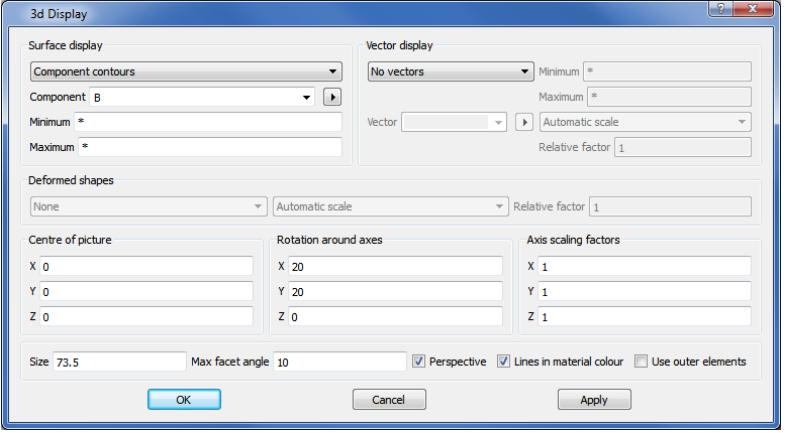
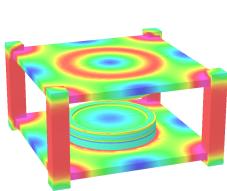
Since it may take an appreciable time to perform the display operations on a complex model, a progress bar is provided in the bottom-right of the Post-Processor window to show how the selection and display process is proceeding.

## Displaying Results

### Results on the Geometry

Results from the simulation, such as the components of the magnetic flux density,  $B$ , can be super-

imposed on the geometry by using the **3d Display** toolbutton .

 <b>3d Display</b>	<p>Select the <b>Component contours</b> option from the <b>Surface display</b> drop-down list, and change the <b>Component</b> to the required field component. For example, enter <b>B</b> to display the modulus of the magnetic flux density.</p>  <p>The required field component may be typed in the entry field. As is common with other entry fields in Opera, arithmetic operations may be performed on components. Alternatively, the component may be selected using the right arrow adjacent to the <b>Component</b> option . Note that only commonly used components are presented in the list of options; it is not necessarily exhaustive.</p>
 <b>Outline view of model</b>   <b>Axes</b>	<p>In addition to the contours, the surface finite element mesh is initially displayed over the geometry. The mesh can be toggled on and off, as can the coordinate system axes and contour labels (i.e. the scale on the left hand side of the Post-Processor window).</p> 

The scale to the side shows the magnitude of the flux density in gauss. The user should check that the flux density values are about as expected. The maximum value of the flux density should be about 19 kilogauss.

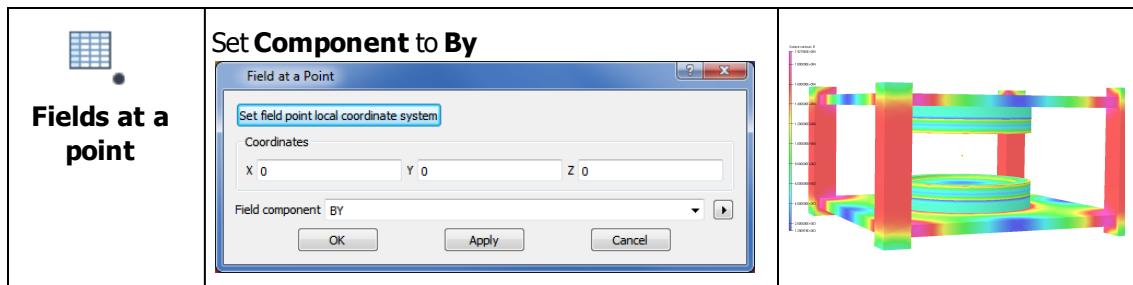
The user should also check the field display to ensure that the model has been set up correctly. It may also be necessary to check that the direction of the field is correct in the symmetry copies of the model in order to verify that the correct boundary conditions have been applied on the symmetry planes. This can be achieved visually using options in the **3d Display**:

- Select an individual field component, **Bx**, **By** or **Bz**, as the **Surface display** field component.
- Select **Vectors** from the **Vector display** drop-down list and enter the required component (**B**) in the **Vector** entry field.
- Select **Automatic scale** from the drop-down list.

If the magnetic fields do not show the expected field pattern, then the reflections in the coordinate planes were not entered correctly when the model symmetry was specified (see [Surrounding Air and Boundary Conditions \[page 50\]](#)).

## Fields at a Point

Fields at specific points, along lines and on patches within the model space can be calculated. As an example, to calculate the Y component of magnetic flux density at, say, the origin, use the **Fields at a Point** toolbutton.



The position of the chosen point, at  $(0, 0, 0)$ , is high-lighted. The value returned should be about +1910 Gauss (the value may vary slightly due to minor differences in meshes, although the value will converge to a more accurate value as the number of elements is increased in the model).

The field at any point within the model space can be evaluated.

The symmetry settings for the model can also be checked by testing field points in any reflected copy. For example, in this model, the Y-component of flux density at any of the set of 4 symmetrically disposed sample points, corresponding to each  $(x, \pm y, \pm z)$ , should have the same value at each point.

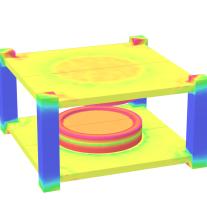
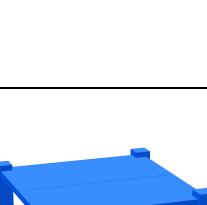
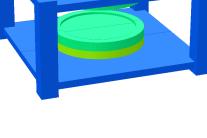
## Force Between Poles

Generally, the forces on magnetic objects may be computed by integration of the full Maxwell stress tensor over a surface surrounding the object. However, for a symmetric magnet, such as this one, it

$$(B_y)^2$$

is sufficient to compute the integral of  $\frac{(B_y)^2}{2\mu_0}$  on the mid-plane between the poles. This formula is valid for SI units, and since the model was created in CGS, the system of units must be changed. This may be performed simply in the Post-Processor.

### Changing the unit system

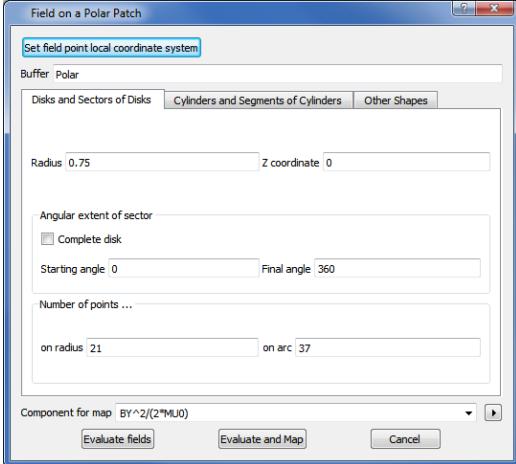
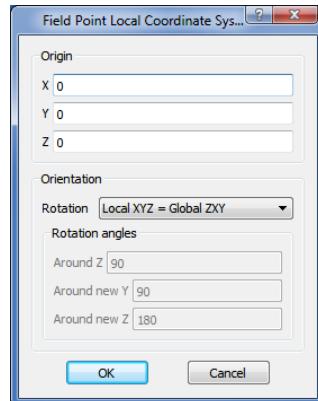
 <b>Units</b>	Click on <b>SI Units</b> followed by <b>OK</b> . Note that the units in the Units panel are updated (top right in the Post-Processor window).	
 <b>Initial view</b>	The view needs to be initialized to the new unit system.  Note that $B_y$ is now displayed on the surface, since that is the component last requested.	
 <b>3d Display</b>	Restore the material colour display by selecting <b>Material colours</b> .	

### Field Values on a Plane

It is important that the integration captures essentially all the flux that is passing between the poles across the mid-plane. Given the geometry of the magnet, a circular patch on the ZX plane extending nearly to the legs of the device will achieve this.

The fields on the patch are calculated at a regular grid of points in the polar coordinate system. The discretization of the grid must be sufficiently fine to adequately capture the variation of field over the area being mapped.

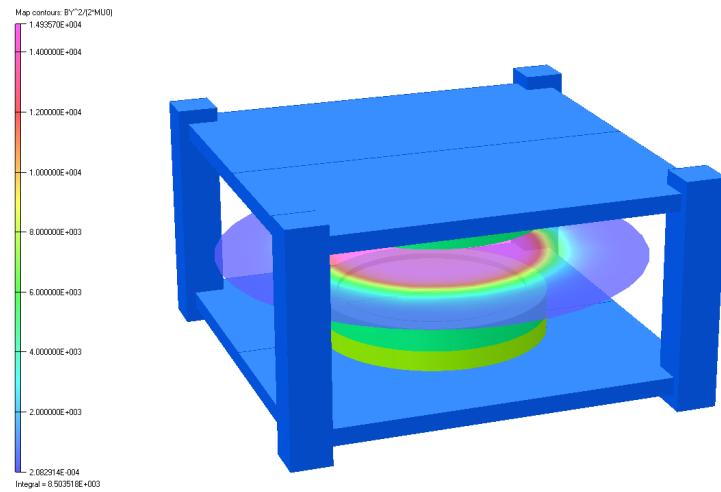
Use the  **Polar Patch** toolbutton to define the patch.

 <b>Polar Patch</b>	<p>Select the <b>Disk and Sectors of Disks</b> tab. Set <b>Radius</b> to be <math>0.75</math>, and the <b>Z</b> coordinate to <math>0</math>. Select the option for <b>Complete disk</b>. Set the <b>Number of points on radius</b> to <math>21</math>, and <b>on arc</b> to <math>37</math>. Set <b>Component for map</b> to <math>BY^2/(2*\mu_0)</math></p>  <p>The circular patch is defined in local polar <math>(r, \theta, z)</math> coordinates. In this example, the patch must lie in the global ZX plane, so a local coordinate system defining the plane of the patch must be set with its local Z-axis along the global Y-axis, and its origin coincident with the global origin<sup>a</sup>.</p> <p>Press the <b>Set field point local coordinate system</b> button; set the <b>Origin</b> to <math>(0, 0, 0)</math> and <b>Rotation</b> to <b>Local XYZ = Global ZXY</b>, followed by <b>OK</b>.</p>  <p>Click on <b>Evaluate and Map</b> on the main dialog.</p>
---	---

The stress on the patch,  $(B_y)^2/(2\mu_0)$ , is plotted as a contour (zone) map within the 3d display as shown in [Figure 2.6](#).

---

<sup>a</sup>When a local coordinate system is being used, it affects only the coordinates of the field points. The field components are still in global coordinates.



*Figure 2.6 Stress through the centre of the air gap*

The integral of the stress, i.e. the force will also be calculated. This integral (about 8500 N), is displayed in the status bar at the bottom of the graphics screen.

**Close** the Post-Processor session.

## Further work

---

The accuracy of the simulation will depend to a large extent on the size and quality of the mesh elements. As noted previously, in some regions, the mesh sizes chosen for this example model are relatively large compared with the details of the local geometry; the accuracy of the simulation may therefore be comparatively low. It is recommended that the user experiments with changing the mesh sizes to explore their effect on accuracy.

This concludes the present introductory example.

# **Chapter 3**

# **Geometric Modeller Features**

## **Introduction**

---

This chapter provides a more complete description of the features in Modeller, with simple examples.

### **Defining geometry**

- Topology Definitions [page 75]
- Sketching Primitives [page 79]
- 2D Sketching with Wire-Edges [page 89]

### **Manipulating geometry**

- Picking and Hiding Entities [page 94]
- Rapid Picking of Entities [page 100]
- Operations on Picked Entities [page 102]
- Boolean Operations [page 104]
- Transforming and Copying Objects [page 113]
- Sweeping Faces [page 117]
- Lofting Between Surfaces [page 123]
- Shell and Offset [page 127]
- Blend and Chamfer [page 129]
- Morphing Operations [page 134]
- Local Coordinate Systems [page 143]

### **Creating data for Opera-3d**

- Mesh Control [page 147]
- Defining and Editing Conductors [page 153]

- [Uses of Volume Data \[page 160\]](#)
- [Analysis Options and Database Creation \[page 164\]](#)

## **Advanced modelling**

- [Parameterizing Models and Rebuilding \[page 169\]](#)
- [Parts Libraries and CAD Files \[page 172\]](#)

# Topology Definitions

## Introduction

This section introduces commonly used terms and their meaning within the Modeller.

## Geometric Entities

### Primitive

A primitive is one of the 3-dimensional geometric shapes from which a model can be built. Primitives can have one of the following shapes: block, cylinder (including tube and polygonal prism), sphere, torus and wire edge. Primitives can be combined and altered in various ways to make more complex shapes.

### Vertex

This is a simple point in space, defined by a position within a cartesian coordinate system. Normally a vertex will mark a sharp change in geometry, e.g. the intersection of 2 or more edges.

### Edge

An edge is a line in space. Typically an edge will have 2 vertices defining its ends. The edge can also have an underlying geometry associated with it, e.g. it could be straight, an arc of a circle, or a more complex underlying spline curve. For some edges, e.g. circles and ellipses, the edge may have a single vertex or no end vertices at all.

### Wire edge

An edge often exists as a boundary to a face, but may exist independently as a wire edge. A wire edge can be created by copying an existing edge or by using the **WIREEDGE** command.

### Face

A set of edges, connected together at their ends, forms a loop. For most cases, such a loop forms the boundary of a face. A face also has an underlying geometry, e.g. the edges could bound part of a plane, part of a sphere or part of a more complex underlying surface definition.

There are also some cases where a face is bounded by more or less than a single loop.

- For some underlying surface geometries, e.g. a sphere or torus, there may be no bounding loop of edges required.

- For structures formed from complex operations, the face may contain internal holes. In this case there may be one or more internal loops of edges bounding interior sections of the surface to create the holes in the face.

## Sheet face

Typically a face will form part of a closed set of faces that completely encloses a volume, but it can exist independently as a sheet face. A sheet face can be created by copying an existing face or by creating a primitive (e.g. a block) with no thickness.

## Cell

A closed set of faces connected at the bounding edges forms a shell. A cell represents an enclosed volume of space bounded by one or more shells. Any point within the cell can be reached from another point within the cell by travelling a path that does not pass through a face. For some complex geometries there may be more than a single shell of faces bounding a cell, e.g. where a section has been removed from the interior of a volume, leaving an internal set of faces restricting the volume occupied by the cell.

## Body

A body is a collection of any of the topological entities discussed above. All objects within a body are topologically linked, so that parts of a body cannot be moved without considering the impact on other parts of the body. For example a body may contain 2 cells. These cells each occupy their own space. Movement of one cell within the body could make the cells overlap which would cause the cells to become invalid.

However, if there are 2 bodies, each with one cell, it is permissible to have the bodies overlapping as there is no connection between the bodies. Before forming a final model, the bodies will be merged into a single body to ensure that the topology forms a single valid structure.

## Local coordinate system

A local coordinate system (LCS) is specified by a position for the origin, and a rotation of the local axes with respect to the global coordinate system.

## Conductor

Special primitives can be defined in the Modeller to define standard conductor shapes. These include racetrack coils, solenoids and others.

## Entity

The term entity is used to include any of the terms defined above.

## Working coordinate system

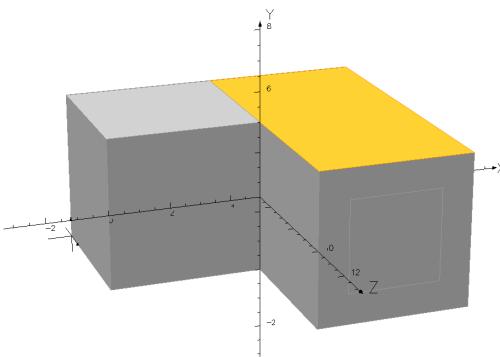
A local coordinate system can be chosen as a Working Coordinate System (WCS). While a WCS is active, values given for operations are in the Working Coordinate System, rather than in the Global Coordinate System.

## Geometric Model Information

The geometric model held within the Modeller consists of a set of bodies, with their component cells, faces, edges and vertices, and Local Coordinate Systems and Conductors.

The information about what is contained within the model is always available in the status bar at the bottom right of the screen. Each number represents the number of entities of a specific type, defined in the current model. Leaving the cursor on any of the entries in the status bar will bring up a tooltip giving more information about its meaning.

For the model, in [Figure 3.1](#), the status bar information at the bottom right of the window, shows that the model consists of 3 bodies; 3 cells; 18 faces (with 1 of them picked); 36 edges; 24 vertices; no conductors.



Component | 3 | 3 | 18(1) | 36 | 24 | 0 | Global coordinate system

*Figure 3.1 Entities displayed by the Modeller*

## Bodies and Cells

A common confusion when first using the Modeller is the misunderstanding of differences between bodies and cells.

When creating a body using a basic building object, e.g. a block, the object created is a single body, containing 1 cell, 6 faces, 12 edges and 8 vertices. The close correspondence between a body and a cell in this instance makes it hard to differentiate between them.

However, after bodies have been combined using non-regular operations, a single body may contain more than 1 cell, and may contain no cells if the single body has only sheet faces.

Whenever performing geometric operations, they are almost always performed on a body rather than on a cell. This is because the topology of connected entities must remain consistent within a body. By trying to move a cell within a body, all connected faces and edges would be affected and may well become invalid.

Similarly, if material properties were defined on a body, only one material would be available for each body, making separation of materials within a body very difficult.

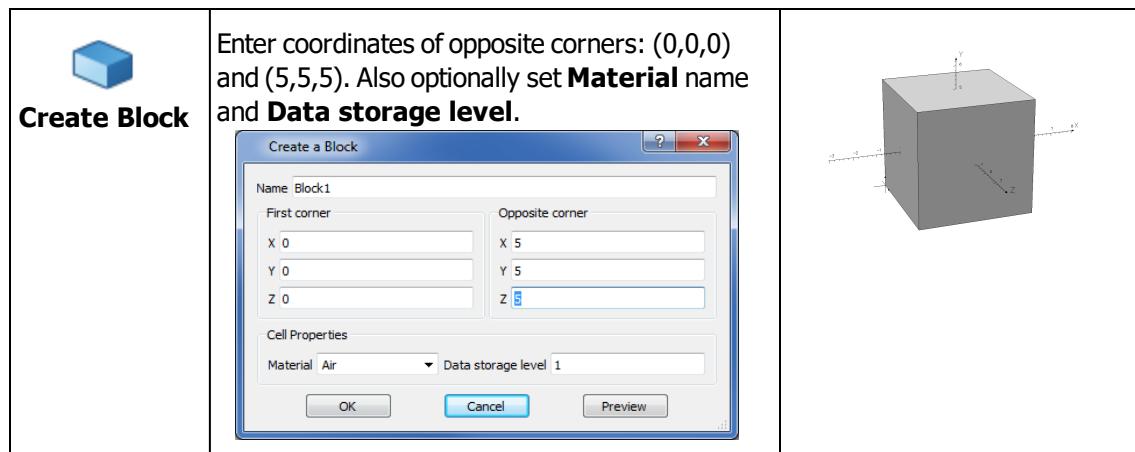
## Sketching Primitives

There are two ways to define basic primitives within the Modeller, either completing the relevant dialog with appropriate values, or using a 2D sketching mode to draw the initial shape interactively, and then make final adjustments to values before drawing.

The following shows how each primitive can be built using the 2D sketching facility. The sketching is based on a 2D plane, oriented on the local XY plane in the current Working Coordinate System.<sup>1</sup>

### Block

A simple Block primitive can be drawn by completing the dialog with the coordinates of the opposite corners, and optionally the material name and a data storage level.

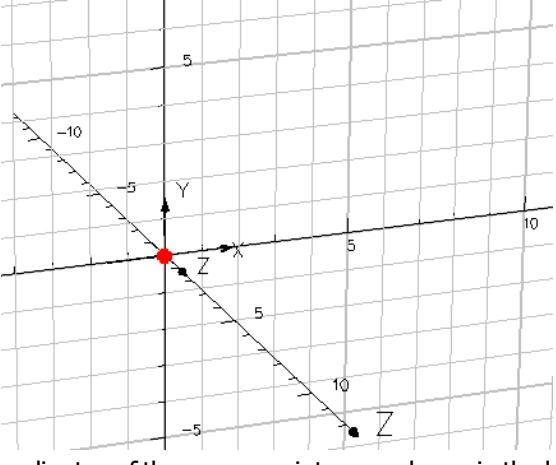
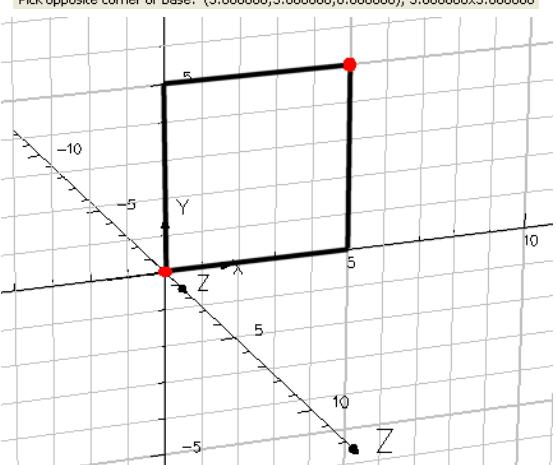


Alternatively the 2D sketching can be used to draw the same primitive interactively. Delete everything done so far by "rewinding the history stream" using one of the following:

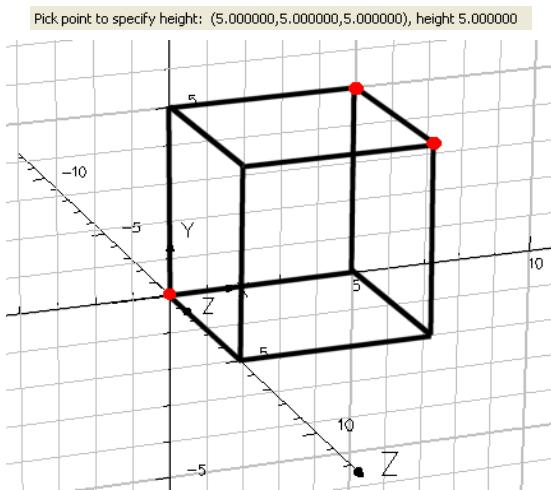
press the **History** button and select **Start of modelling operations**, **Ctrl-z** or the  tool-

<sup>1</sup>For better visibility the edges of the entities in this document have been changed to black, by deselecting the

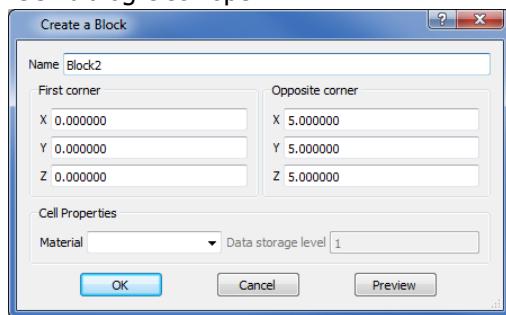
 option **3d Display > View edges in material colour** and setting the edge colour to black in  **Change colours > Standard > Edges**.

 <b>Sketching Active</b>   <b>Create Block</b>	<p>Move the mouse pointer over point (0,0) and double-click the left mouse button to select the first point.</p>  <p>Note that the coordinates of the mouse pointer are shown in the lower left corner of the Modeller window:</p> <div style="background-color: #f0f0f0; border: 1px solid #ccc; padding: 2px; display: inline-block;">Pick first corner of block: (0.000000,0.000000,0.000000)</div>
	<p>Move mouse over point (5,5) and double-click the left mouse button to select the second point. The coordinates and size of the area are again shown in the lower left corner of the Modeller window:</p>  <div style="background-color: #f0f0f0; border: 1px solid #ccc; padding: 2px; display: inline-block;">Pick opposite corner of base: (5.000000,5.000000,0.000000), 5.000000x5.000000</div>

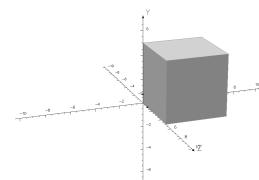
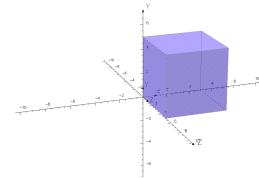
Move mouse to a height of 5 and double-click the left mouse button to select the third point. The height is shown in the lower left-hand corner of the Modeller window:



The **Create a Block** dialog is then presented with the appropriate coordinates inserted. The **Material name** and **Data storage level** can optionally be set at this stage, or changes made to the coordinates if desired. To indicate that changes can still be made to the geometry, a translucent blue colour is used.  
Note that the view of the "preview" block can be moved with the mouse whilst the **Create a Block** dialog is still open.

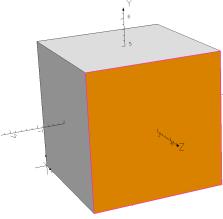
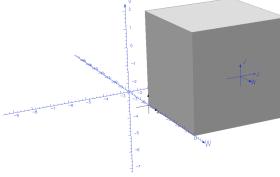
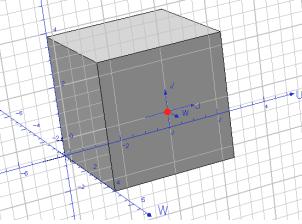
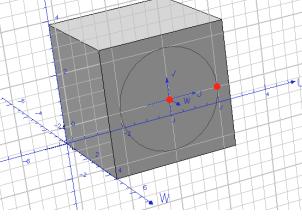


Select **OK** to finally draw the primitive.



## Cylinder

To draw a cylinder primitive on top of one face of the block, proceed as follows:

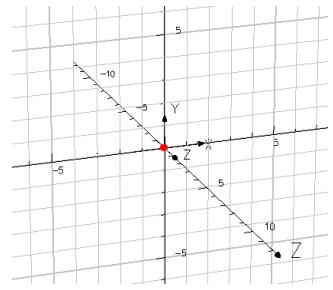
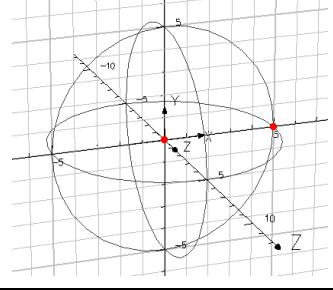
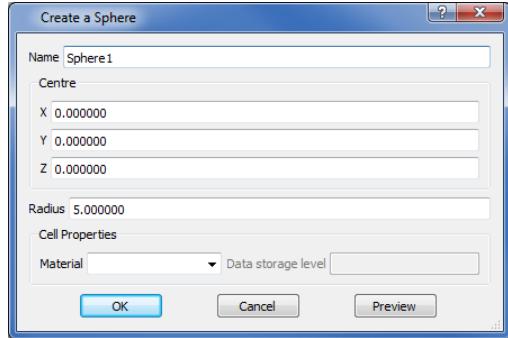
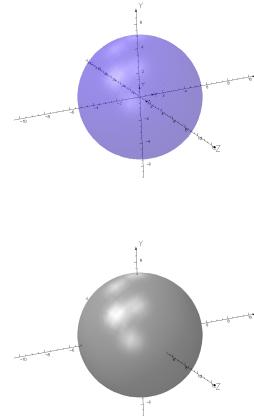
 <b>Pick Faces</b>	<p>Ensure that the  <b>Pick Entity</b> action has been selected and then pick the front face (at z=5) with a double-click.</p>	
 <b>Coordinate Systems &gt; Move WCS</b>		
 Ensure <b>Sketching Active</b> is selected   <b>Create Cylinder.</b>	<p>Move mouse over point (0,0) of the Local Coordinate System and double-click the left mouse button to select the first point. Note that the (local) coordinates of the mouse are shown in the lower left corner of the Modeller window:</p> <p>Pick centre of base of cylinder: (0.000000,0.000000,0.000000)</p>	
	<p>Move mouse over point (2,0) and double-click the left mouse button to select the second point. The coordinates and radius of the base are again shown in the lower left corner of the Modeller window:</p> <p>Pick point to specify radius: (2.000000,0.000000,0.000000), radius 2.000000</p>	

	<p>Move mouse to a height of 6 and double-click the left mouse button to select the third point. The height is shown in the lower left-hand corner of the Modeller window:</p> <pre>Pick point to specify height: (2.000000,0.000000,6.000000), height 6.000000</pre>	
	<p>The dialog is then presented with the appropriate coordinates inserted. Select <b>OK</b> to finally draw the primitive.</p>	

## Sphere

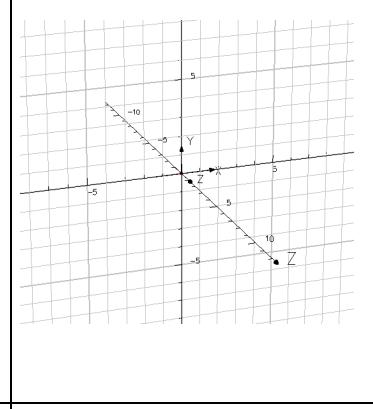
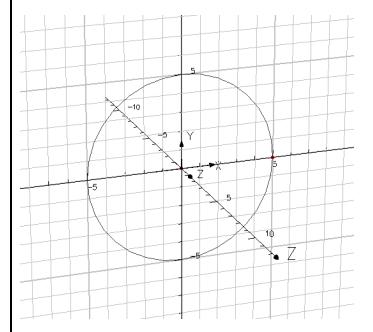
Before drawing a sphere primitive using the 2D sketching mode, reset the Modeller to its initial state

with **Clear All**.

 <b>Ensure Sketching Active</b> is selected   <b>Create Sphere</b>	<p>Move mouse over point (0,0) and double-click the left mouse button to select the first point. Note that the coordinates of the mouse are shown in the lower left corner of the Modeller window:</p> <p>Pick centre of sphere: (0.000000,0.000000,0.000000)</p>	
	<p>Move mouse over point (5,0) and double-click the left mouse button to select the second point. The coordinates and radius of the sphere are again shown in the lower left corner of the Modeller window:</p> <p>Pick point to specify radius: (5.000000,0.000000,0.000000), radius 5.000000</p>	
	<p>The dialog is then presented with the appropriate coordinates inserted. Select <b>OK</b> to finally draw the primitive.</p> 	

## Torus

To draw a torus primitive using the 2D Sketching mode, proceed as follows:

 Ensure <b>Sketching Active</b> is selected	<p>Move mouse over point (0,0) and double-click the left mouse button to select the first point. Note that the coordinates of the mouse are shown in the lower left corner of the Modeller window:</p> <p>Pick centre of torus: (0.000000,0.000000,0.000000)</p>	
	<p>Move mouse over point (5,0) and double-click the left mouse button to select the second point. The coordinates and major radius of the torus are again shown in the lower left corner of the Modeller window:</p> <p>Pick point to specify major radius: (5.000000,0.000000,0.000000), major radius 5.000000</p>	

	<p>Move mouse over point (7,0) and double-click the left mouse button to select the third point. The coordinates and minor radius of the torus are again shown in the lower left corner of the Modeller window:</p> <p>Pick point to specify minor radius: (7.000000,0.000000,0.000000), minor radius 2.000000</p>	
	<p>The dialog is then presented with the appropriate coordinates inserted. Select <b>OK</b> to finally draw the primitive.</p>	

## Wire

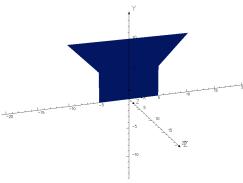
To draw a wire primitive using the 2D sketching mode, proceed as follows:

 <b>Ensure Sketching Active</b> is selected   <b>Create Wire-Edge.</b>	<p>Move mouse over point (-5,0) and double-click the left mouse button to select the first point. Note that the coordinates of the mouse are shown in the lower left corner of the Modeller window:</p> <p>Pick starting point of wire edge: (-5.000000,0.000000,0.000000)</p>	
	<p>Move mouse over point (5,0) and double-click the left mouse button to select the second point. The coordinates of start and end points of the wire are again shown in the lower left corner of the Modeller window:</p> <p>Pick end point of wire edge: start (-5.000000,0.000000,0.000000), end (5.000000,0.000000,0.000000)</p>	
	<p>The dialog is then presented with the appropriate coordinates inserted. Select <b>OK</b> to finally draw the primitive.</p>	

## Wire polygon

Wire edges can be "chained" together during sketching so that the end of one edge becomes the start of the next. If the set of wires forms a closed polygon, it can be covered to form a polygonal sheet face, which can then be extruded to form a body (see [Sweeping Faces \[page 117\]](#)).

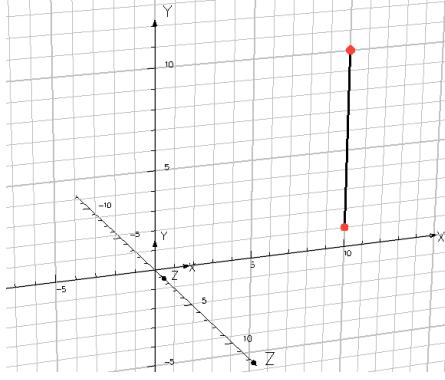
Repeat the previous example, but instead of the final **OK**, select **Continue**. The sketching grid will reappear so that a new point can be selected. Further points can be added in the same way. If the new point is the same as the starting point of the polygon, the polygon will be closed and covered.

 Ensure <b>Sketching Active</b> is selected   <b>Create Wire-Edge</b>	<p>Select the following points; click the dialog <b>Continue</b> button after each one (including the final one):</p> <p>(-5,0) (5,0) (5,5) (10,10) (-10,10) (-5,5) (-5,0)</p>	
---	--	---

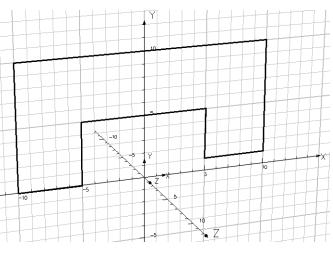
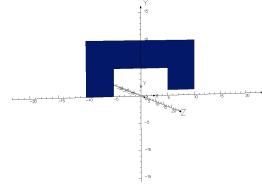
## 2D Sketching with Wire-Edges

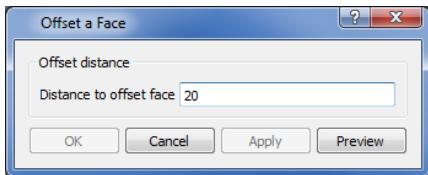
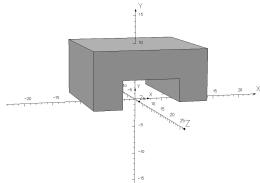
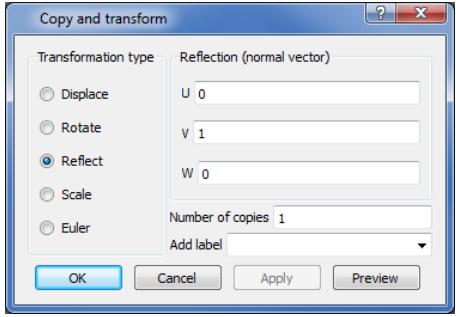
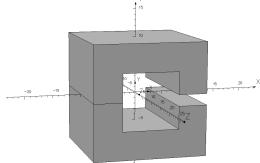
In this example a simple C-Core is drawn with wire edges, to introduce another option of entering geometric data. The **COVER** command is used to form a sheet face with zero thickness and the **OFFSET** command to generate a volume.

Delete everything done so far by "rewinding the history stream":  
**History** and select **Start of modelling operations**.

 Ensure <b>Sketching Active</b> is selected   <b>Create Wire-Edge</b>	<p>Move the mouse pointer over point (10, 1) and double-click the left mouse button to select the first point. Then move the mouse pointer to point (10, 10) and double-click.</p>  <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>Create a Wire-Edge</b></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>Name <input type="text" value="WireEdge"/></td> <td>End point</td> </tr> <tr> <td>Start point</td> <td>X <input type="text" value="10.000000"/> Y <input type="text" value="1.000000"/> Z <input type="text" value="0.000000"/></td> </tr> <tr> <td></td> <td>X <input type="text" value="10.000000"/> Y <input type="text" value="10.000000"/> Z <input type="text" value="0.000000"/></td> </tr> <tr> <td>Edge type</td> <td> <input checked="" type="radio"/> Straight  <input type="radio"/> Arc  <input type="radio"/> Centred  <input type="radio"/> Obtuse centred           </td> </tr> <tr> <td colspan="2" style="text-align: right;">No data is required</td> </tr> <tr> <td colspan="2" style="text-align: right;"> <a href="#">Continue</a> <a href="#">OK</a> <a href="#">Cancel</a> <a href="#">Preview</a> </td> </tr> </table> </div>	Name <input type="text" value="WireEdge"/>	End point	Start point	X <input type="text" value="10.000000"/> Y <input type="text" value="1.000000"/> Z <input type="text" value="0.000000"/>		X <input type="text" value="10.000000"/> Y <input type="text" value="10.000000"/> Z <input type="text" value="0.000000"/>	Edge type	<input checked="" type="radio"/> Straight <input type="radio"/> Arc <input type="radio"/> Centred <input type="radio"/> Obtuse centred	No data is required		<a href="#">Continue</a> <a href="#">OK</a> <a href="#">Cancel</a> <a href="#">Preview</a>	
Name <input type="text" value="WireEdge"/>	End point												
Start point	X <input type="text" value="10.000000"/> Y <input type="text" value="1.000000"/> Z <input type="text" value="0.000000"/>												
	X <input type="text" value="10.000000"/> Y <input type="text" value="10.000000"/> Z <input type="text" value="0.000000"/>												
Edge type	<input checked="" type="radio"/> Straight <input type="radio"/> Arc <input type="radio"/> Centred <input type="radio"/> Obtuse centred												
No data is required													
<a href="#">Continue</a> <a href="#">OK</a> <a href="#">Cancel</a> <a href="#">Preview</a>													

The **Continue** key can be pressed either with the mouse, or with the **Enter** key on a keyboard (the default button in the dialog box).

	<p>Carry on with double-clicks on          (-10, 10)          (-10, 0)          (-5, 0)          (-5, 5)          (5, 5)          (5, 1)</p>	
	<p>To close the polygon, double-click on the first point, (10, 1), again and press the <b>Continue</b> button in the dialog box.</p>	

 <b>Pick Faces</b>  <b>Pick Bodies</b> <b>Offset</b> <b>Copy</b>	<p>Move the mouse pointer over the face. When highlighted, right-click and select <b>Offset</b> from the context menu and enter a value of 20.</p>  <p>Note that the <b>Offset</b> operation on a sheet face creates a volume which extends half the distance in front and half behind the original face.</p>	
 <b>Pick Faces</b>  <b>Pick Bodies</b> <b>Offset</b> <b>Copy</b>	<p>Move the mouse pointer over the body. When highlighted, right-click and select <b>Copy</b> from the context menu. Perform a reflection in a plane with the face normal vector <math>(0, 1, 0)</math>.</p> 	

Save this model for use in a later section with the  **Save As** option and give the file name **C\_Core.opc**.

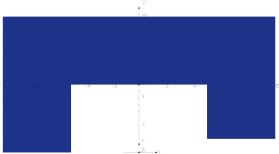
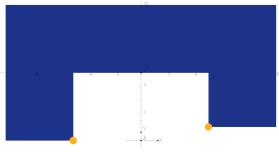
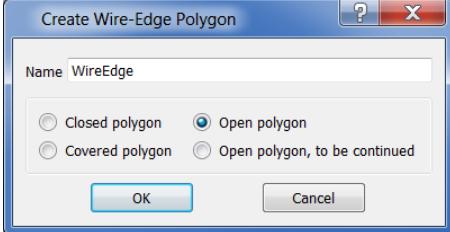
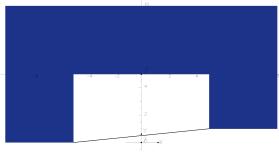
## Creating Edges and Sheet Faces from existing Points

In this example we will modify an earlier created model **C\_Core.opc** to create additional edges in the 2D sketch and show how to convert a group of edges or vertices into a face. If you continue from

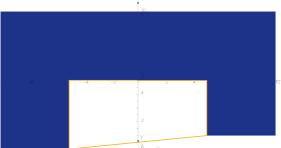
the previous exercise click on **Toggle Sketch Mode**  to turn sketching off.

If you have not conducted the previous exercise, open the **C\_Core.opc** file from the Examples Folder, click the **Replay** button  from the **History** group, select the last replay command from the list and click **OK**. Then press **OK** again in the **Copy and Transform** dialog that opens. This operation will restore the command history.

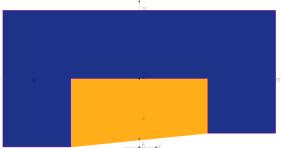
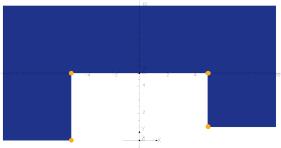
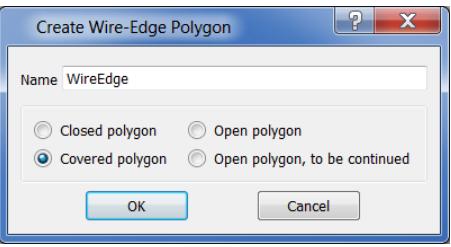
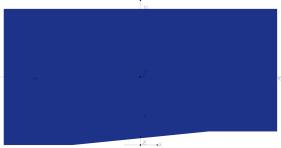
## Creating a new Edge from two existing Vertices

<b>History</b>	Return to the 2D sketch of the C-Core: navigate to the <b>History</b> and select the line after <b>Close and Cover Wire Edge</b> . Press OK.								
 <b>Pick Vertices</b>	Change picking type to vertices or by typing <b>v</b> on the keyboard. Move the mouse pointer over the vertex $(-5, 0)$ . When highlighted, double click to pick or press <b>p</b> on the keyboard. Note the picked vertex changes colour to orange.  Now, move the mouse pointer over the vertex at $(5, 1)$ and pick it as well.								
 <b>Create a wire-edge</b>	Open the <b>Create Wire-Edge Polygon</b> dialog and select <b>Open polygon</b> (if this dialog does not open, make sure that sketching is turned off).   <p>Note that this operation creates a second body: the edge.</p> <table border="1" data-bbox="525 1374 879 1410"> <tr> <td>Component</td> <td>2</td> <td>0</td> <td>1</td> <td>9</td> <td>10</td> <td>0</td> </tr> </table>	Component	2	0	1	9	10	0	
Component	2	0	1	9	10	0			

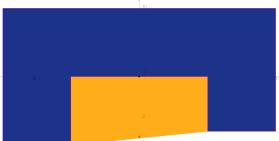
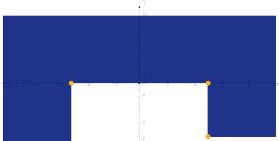
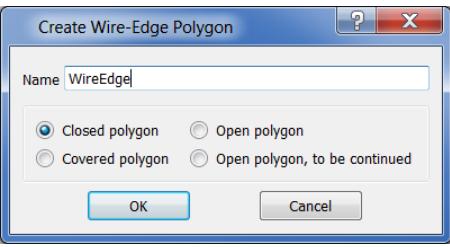
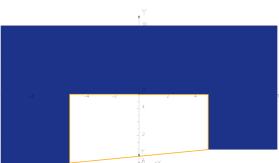
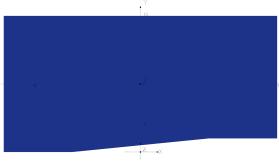
## Creating a new Face from Edges

 <b>Pick Edges</b>	<p>Change picking type to edges or by typing <b>e</b> on the keyboard. Move the mouse pointer over the edge created in the previous step. When highlighted, type <b>p</b> on the keyboard or double-click to pick the edge.</p> <p>Repeat the last step to pick the other edges shown in the figure. Note the picked edges become orange.</p>	
 <b>Cover</b>	<p>Select <b>Cover</b> from the context menu or from the tabbed menu.</p> <p>Note that the model now comprises of two bodies: the original sketch and the new sheet face.</p>	

## Creating a new Face from Vertices - Option 1

 <b>Pick Bodies</b>	<p>Change picking type to bodies or by typing <b>b</b> on the keyboard. Move the mouse pointer over the body corresponding to the new face. When highlighted, right-click and select <b>Delete</b> from the context menu. Press <b>OK</b> to delete the picked body.</p>	
 <b>Pick Vertices</b>	<p>Change picking type to vertices. Pick four vertices <math>(-5, 0)</math>, <math>(-5, 5)</math>, <math>(5, 5)</math>, <math>(5, 1)</math> as shown.</p> <p>The vertices should be picked sequentially in a clockwise or anticlockwise order following the expected edges of the faces.</p>	
 <b>Create Wire-Edge</b>	<p>Open the <b>Create a wire-edge</b> dialog and select <b>Covered polygon</b>.</p> 	

## Creating a new Face from Vertices - Option 2

 <b>Pick Bodies</b>	<p>Change picking type to bodies. Pick and delete the new body of the sheet face.</p>	
 <b>Pick Vertices</b>	<p>Change picking type to vertices. Pick four vertices <math>(-5, 0)</math>, <math>(-5, 5)</math>, <math>(5, 5)</math>, <math>(5, 1)</math> that will form a new face.</p>	
 <b>Create a wire-edge</b>	<p>Open the <b>Create Wire-Edge Polygon</b> dialog and select <b>Closed polygon</b>.</p>  <p>This operation creates a new body consisting of four edges.</p>	
 <b>Pick Bodies</b>  <b>Cover</b>	<p>Change picking type to bodies.</p> <p>Move the mouse pointer over one of the edges of the new body. When high-lighted, press the p letter on the keyboard or double-click to pick the body.</p> <p>Select <b>Cover</b> from the context menu or from the tabbed menu.</p>	

## Picking and Hiding Entities

Picking entities allows the user to pick entities in order to perform operations on single entities or groups, for example geometric operations or assigning material properties. Entities can also be hidden from view to allow views of other parts of a model to be seen.

For these examples a set of 3 bodies is used to demonstrate the principles of picking and hiding. These are created in the following way:

	<p>Before we start, make sure that the 2D Sketching is switched off (the toolbutton is not pressed).</p>	
 <b>Create a Block</b>		<p>This creates the first entity.</p>
 <b>Create a second Block</b>   then initialize the view		<p>This creates the second entity.</p>
 <b>Create a third Block</b>		<p>This creates the third entity.</p>

Note that the third block sits within block two, and as such it is not possible to see it at this stage.

## Picking Bodies

Using the example model that has just been created the bodies can be picked by first selecting the

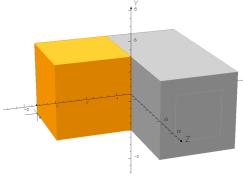
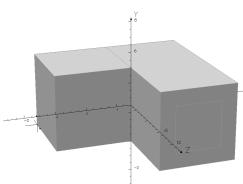


**Pick Entity** button and then selecting the



**Pick Bodies** button.

Now when the mouse is moved over the model, the separate bodies are high-lighted by a coloured edging. Once a body is high-lighted it can be picked by double-clicking the left mouse button.

 <b>Pick Bodies</b>	<p>Moving the mouse over a block high-lights the outline of the body. Double-clicking the left mouse button selects high-highlighted block. High-light and pick block <b>one</b>.</p>	<p>The colour of bodies changes as they are picked.</p> 
	<p>Double-clicking the left mouse button unpicks block <b>one</b>.</p>	

## Picking Other Entities

In exactly the same way the cells, faces, edges and vertices of the bodies can also be picked:

- select the appropriate toolbar icon or, while picking is active, change the type of entity using single character keyboard commands:

Entity Type	Toolbar Icon	Keyboard Command
Bodies		<b>b</b>
Cells		<b>c</b>

Entity Type	Toolbar Icon	Keyboard Command
Faces		f
Edges		e
Vertices		v
Local Coordinate Systems		
Conductors		k

- move the mouse until the required entity is high-lighted;
- double-click the left mouse button to toggle the picked state of the entity (or press **p** or **Enter** as a keyboard command).

The status bar at the bottom of the Modeller window shows the numbers of entities, and in parentheses the number of picked entities. For example, the following status bar indicates that the model consists of 3 bodies, 3 cells, and 18 faces, with 1 of them picked, 36 edges and 24 vertices. Hovering the mouse over one of these numbers will bring up a tool-tip explaining what it means.

Component | 3 | 3 | 18(1) | 36 | 24 | 0 | Global coordinate system

Entities can be picked singly, as has been shown in the above example, or by repeating the picking process more than one entity of the same type can be picked. An example for picking several faces quickly can be found in [Rapid Picking of Entities \[page 100\]](#).

Using multiple picking allows the mesh properties or material labels to be assigned to more than one entity at a time. Also when performing Boolean operations, primitives can be joined to form more complex bodies.

## Drilling Down

One of the three blocks that has been created in the above example is not easily accessible for picking or viewing because it inside another. This can be overcome in two ways: by [Hiding Entities \[page 97\]](#) or by "drilling down".

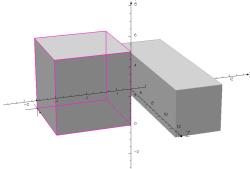
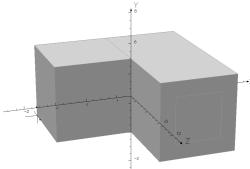
Drilling down can be used as follows. When picking is enabled and the position of the mouse corresponds to several entities which are behind each other, the edge high-lighting normally indicates the entity at the front. Pressing the **<space>** bar will move the high-lighting to the entity behind; pressing **<space>** again will move to the one behind that, if it exists, or back to the front entity. When the required entity is high-lit, double-click can be used to pick the entity in the normal way.

## Hiding Entities

There are two ways in which some entities can be hidden from view.

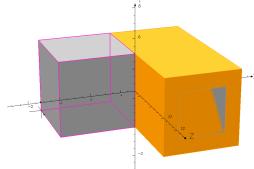
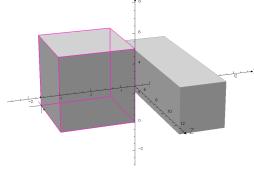
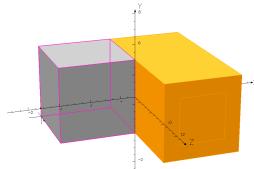
### Entities not involved in an operation

The first method will hide entities not involved in the subsequent operation, so that the entities which are needed for the operation can be picked more easily.

 <b>Hide Entity</b>	<p>Double-clicking on the bodies within the model as they are highlighted will cause the picked body to become hidden from view. Picking block <b>two</b> will now hide it from view making block <b>three</b> visible.</p>	
 <b>Pick Bodies</b>	<p>While block <b>two</b> is hidden, the picking option  can be changed back to <b>Pick Entity</b>, so that block <b>three</b> can be picked.</p>	
 <b>Unhide Entities</b>	<p>By clicking on the  <b>Unhide Entities</b> button all of the hidden entities are made visible again.</p>	

### Hiding picked entities

The second method is to temporarily hide a picked entity until the next action is completed, whereupon the hidden entity will become visible again. This is most useful when having to pick bodies in a particular order to get the correct result from a Boolean operation and will be explained further in the section on Boolean operations.

	If blocks <b>two</b> and <b>three</b> should be picked, first pick block <b>two</b> as before. Note that the front face of block <b>three</b> can be partly seen as it lies on the same surface as the front face of block <b>two</b> .	
		
	Block <b>two</b> is temporarily hidden, but it is still picked.	
	Now block <b>three</b> can be picked. In doing this block <b>two</b> now becomes visible again.	

## Reset Picked Entities

Once a set of entities has been picked and are high-lighted they can be reset by simply pressing the  **Unpick Picked Entities** button .

## Keyboard Commands while Picking

Some keyboard commands have already been described:

- changing the type of entity being picked (filter type);
- drilling down to hidden entities.

Commands can also be used to change the picking function: pick, hide or list information. The following table contains a complete list. The characters can be upper or lower case.

Key	Function
<b>&lt;space&gt;</b>	Cycle through entities which appear on top of each other on the display.
<b>b</b>	Change the filter to body.

Key	Function
c	Change the filter to cell.
e	Change the filter to edge.
f	Change the filter to face.
h	Hide the high-lighted entity.
i	Show information about the high-highlighted entity.
l	The same as i.
p	Pick the high-highlighted entity.
v	Change the filter to vertex.
<enter>	The same as p.

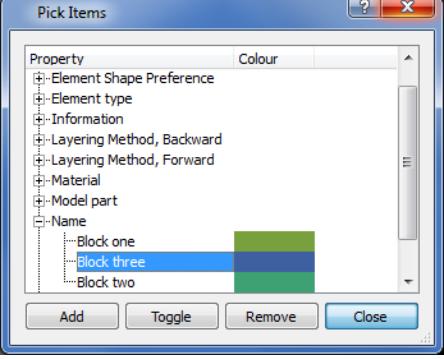
## Picking Bodies by Name or Unique Name

Another method to pick a body which is hidden by other parts of a model, is to pick it by its name, unique name or some other property. In some cases this is faster.

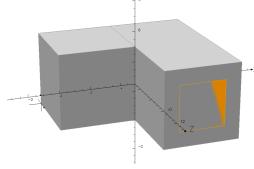


**Pick by Property**

Open the **Name** subset and select block **three**.



Select **Add** and **Quit** to pick the block.

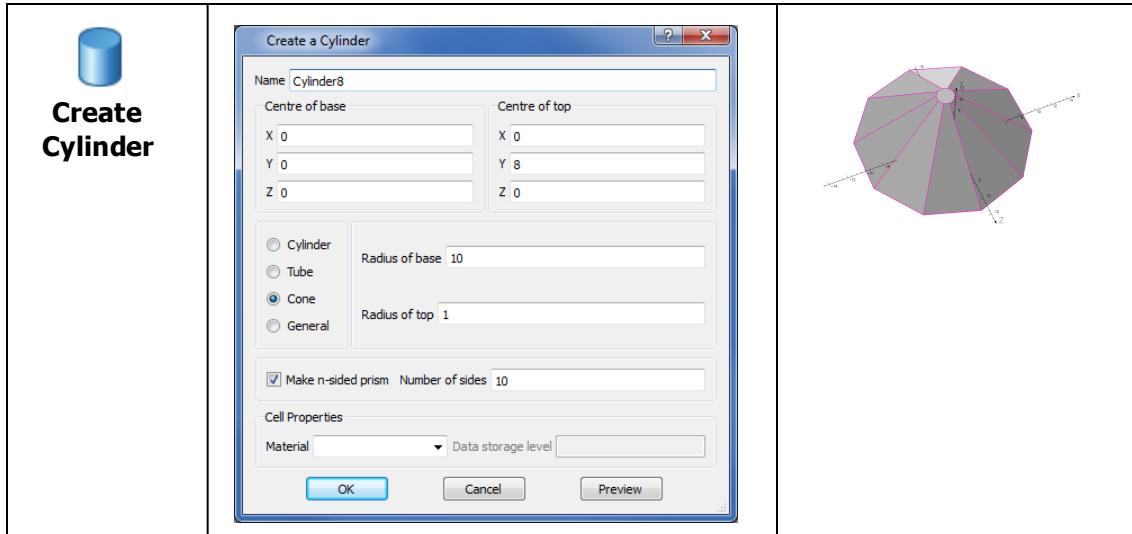


Before moving to the next section, clear all the data using  **Clear All**.

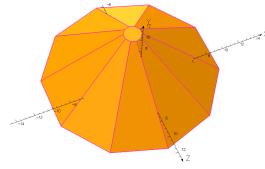
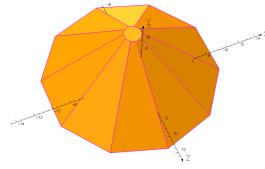
## Rapid Picking of Entities

This example illustrates how to select many entities with a minimum number of mouse clicks. The procedure can be very useful for example in electrostatic models, where all the faces of a body have to be selected in order to apply a label for voltage boundary conditions. Another application might be to select all edges of a body so that a **Blend** or **Chamfer** can be applied.

To show how this can be done, the example creates a 10 sided pyramid.



This pyramid has many faces, and selecting them one by one needs 12 mouse double-clicks. This can be reduced to 1 mouse double-click and a simple operation by the following procedure.

 <b>Pick Bodies</b>	Pick the pyramid as a body.	
 <b>Pick Faces</b>	Press the  <b>Pick Faces</b> button. Note the status bar: the pyramid is still selected as a body. <div style="background-color: #f0f0f0; padding: 2px; border: 1px solid #ccc; display: inline-block;">           Component   1(1)   1   12   30   20   0   Global coordinate system         </div>	

**Change Type  
of Picked  
Entities**

Press the **Change Type of Picked Entities** button. As a result of this operation all faces on the selected body are selected instead. This can be seen on the status bar which now shows 12 picked faces.

Component | 1 | 1 | 12(12) | 30 | 20 | 0 | Global coordinate system |

## Operations on Picked Entities

---

After entities have been picked, operations can be performed or properties changed. This can be done in two ways.

### Tabbed Interface

Buttons in the tabs allow the user to change the geometry by combining, copying or transforming. There are also options to export entities to CAD files. These operations are described in the following sections.

In addition to geometric operations there are functions which change or list the properties of entities: material labels, mesh sizes, etc.

### Context Menus

There is a second way of picking an entity which pops up a "context menu". To use context menus, instead of double-clicking with the left mouse button, use a single click with the right mouse button.

For each type entity there is a separate pop-up menu with context dependent entries. The context menus contain the items from the **Operations** and **Properties** menus which are relevant to the entities which have been picked. If several entities are needed for an operation or property change, the context menu can still be used, if the last entity is picked using the right mouse button, the other entities having been picked with double-click on the left mouse button.

Context menus exist for bodies, cells, faces, edges, vertices, local coordinate systems and conductors. There is also a context menu when no entity has been picked which can be used to change:

- the parts of the model being displayed (**Selection**);
- the size, angles and centre of the picture (**Set View**);
- **Picking** options.

Here are some of the context menus.

Body	Cell	Face	Edge
<ul style="list-style-type: none"><li> Copy and Transform T</li><li> Extract Cells</li><li> Morph</li><li> Check</li><li> Cover</li><li>Rebuild</li><li> Delete Del</li><li> Cell Properties</li><li> List</li><li> Default Select</li><li> Select</li><li> 3d Display</li><li>Common Views</li><li>Picking</li></ul>	<ul style="list-style-type: none"><li> Copy and Transform T</li><li> Extract Cells</li><li> Delete Del</li><li> Cell Properties</li><li> List</li><li> Default Select</li><li> Select</li><li> 3d Display</li><li>Common Views</li><li>Picking</li></ul>	<ul style="list-style-type: none"><li> Copy and Transform T</li><li> Sweep Face</li><li> Offset</li><li> Shell</li><li>Move WCS</li><li> Delete Del</li><li> Face Properties</li><li> List</li><li> Default Select</li><li> Select</li><li> 3d Display</li><li>Common Views</li><li>Picking</li></ul>	<ul style="list-style-type: none"><li> Copy and Transform T</li><li> Blend/Chamfer</li><li> Cover</li><li>Move WCS</li><li> Delete Del</li><li> Edge Properties</li><li> List</li><li> Default Select</li><li> Select</li><li> 3d Display</li><li>Common Views</li><li>Picking</li></ul>

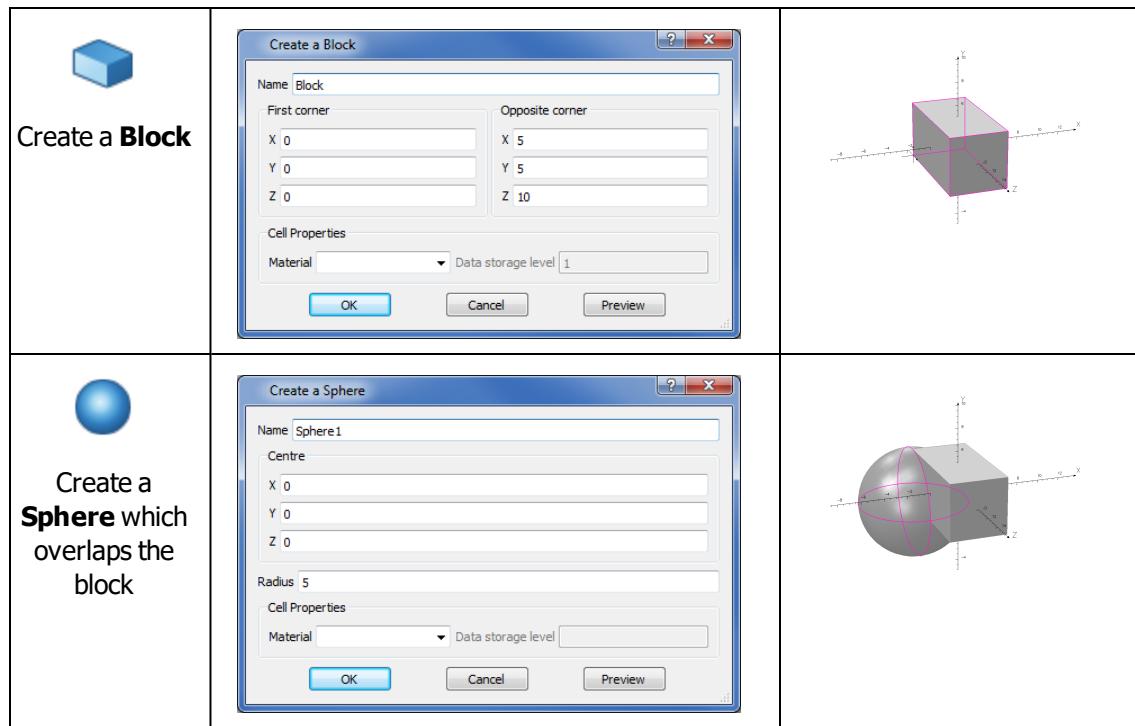
## Boolean Operations

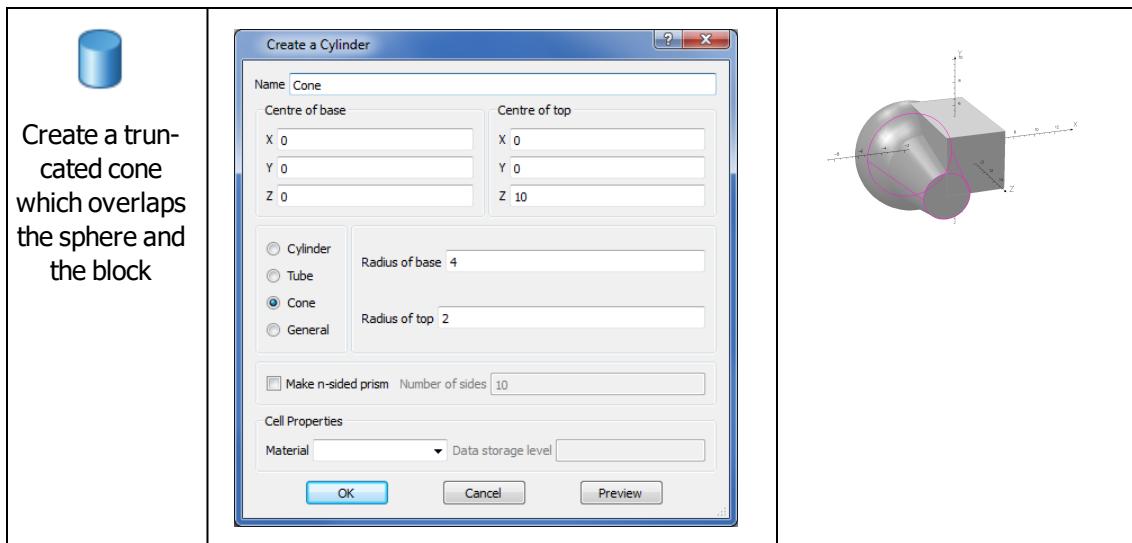
The five basic Boolean operations are:

- Union,
- Intersection,
- Subtraction,
- Trim Overlap,
- Cutaway Overlap.

The Union, Subtraction, Trim Overlap and Cutaway Overlap operations can be applied with or without regularization, which is the process to remove overlapping or internal cells, faces, edges and vertices from the resultant body.

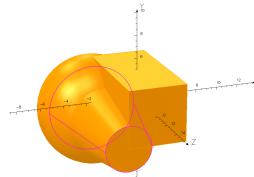
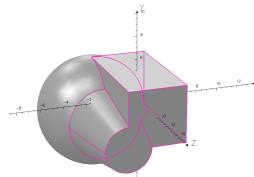
In order to explain these operations a set of examples follow that go through most of the options available to the user. For these examples three bodies are used to demonstrate the principles. The three bodies are created in the following way:





## Union with Regularization

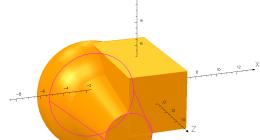
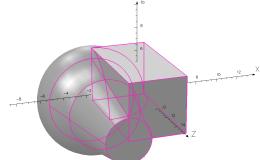
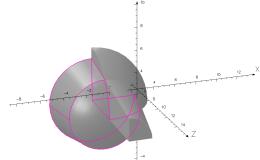
This operation allows several bodies to be combined to form a single body consisting of a single cell.

 <b>Pick Bodies</b>	<p>Using the mouse pick each of the three bodies by double-clicking the left mouse button as each body is highlighted.</p>	
<p>Use the right mouse button to view context menu and select:  <b>Combine Bodies &gt; Union, with Regularization</b></p>		
	<p>The three bodies are now combined to form a single body which consists of a single cell. The status bar at the bottom of the Modeller window now shows that the model consists of 1 body, 1 cell, 9 faces, 19 edges and 12 vertices (none of which are picked - which would be indicated by additional values in brackets).</p>	 <div style="border: 1px solid black; padding: 2px; margin-top: 10px;"> Component   1   1   9   19   12   0   Global coordinate system </div> <p>Moving the cursor over each field explains which entity they refer to.</p>

The  **Undo** button can be used to return the solid model back to the point before the **Union** was applied, or the primitives can be re-created to allow the next example to be started.

## Union without Regularization

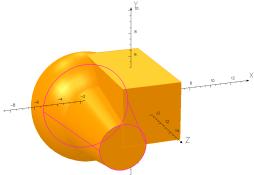
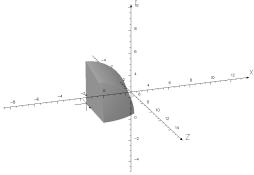
**Union, without Regularization** allows two or more bodies to be joined together such that the original cells are kept and any intersections of the original cells will form new cells. The new cells created can then have separate properties assigned to them.

 <b>Pick Bodies</b>	Using the mouse pick each of the three bodies by double-clicking the left mouse button as each body is high-highlighted.	
Use the right mouse button to view context menu and select: <b>Combine Bodies &gt; Union, without Regularization</b>		
	The three bodies are now combined to form a single body. However this now contains 7 cells, resulting from the overlap of the original 3 cells. This can be seen by looking at the status bar at the bottom of the Modeller window which now shows 1 body, 7 cells, 27 faces, 37 edges and 18 vertices.	
Component   1   7   27   37   18   0   Global coordinate system		
 <b>Hide Cells</b>	The new cells can be seen by hiding the parts of the original cells that formed the Block, Sphere and Cone. Double-clicking the left mouse button as each of these cells is high-highlighted will hide these cells until the <b>Unhide entities</b> button  is pushed. In this way it is possible to view each of the new cells that was formed from an intersection of the original cells.	

The  **Undo** button can be used to return the solid model back to the point before the **Union** was applied, or the primitives can be re-created to allow the next example to be completed.

## Intersection with Regularization

**Intersection with regularization** results in a new body being created that is formed from the intersection of the starting bodies.

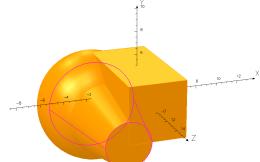
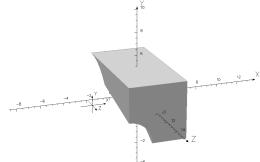
 <b>Pick Bodies</b>	<p>Using the mouse pick each of the three bodies by double-clicking the left mouse button as each body is high-highlighted.</p>	
<p>Use the right mouse button to view context menu and select:  <b>Combine Bodies &gt; Intersection with regularization</b></p>		
	<p>The resultant single body is formed from the intersection of the original three bodies. The parts of the original bodies that were not intersecting are removed. Again the status bar shows that there is now only a single body which has one cell.</p>	

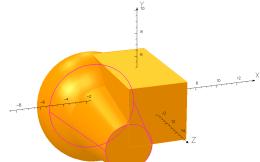
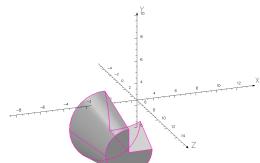


The **Undo** button can be used to return the solid model back to the point before the **Intersection** was applied, or the primitives can be re-created to allow the next example to be completed.

## Subtraction with Regularization

The **Subtraction** operation gives different results depending on the order in which the bodies are picked. The first body picked will have the intersections of the other picked bodies removed from it and the other picked bodies will be removed.

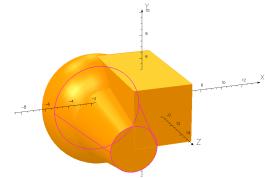
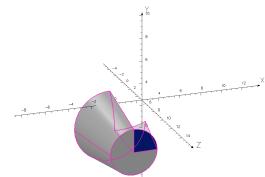
 <b>Pick Bodies</b>	<p>Using the mouse pick each of the three bodies by double-clicking the left mouse button as each body is high-highlighted. Start by picking the Block first and then the Cone and the Sphere.</p>	
<p>Use the right mouse button to view context menu and select:  <b>Combine Bodies &gt; Subtraction, with regularization</b></p>		
	<p>The body that results from this operation is formed from subtracting the Cone and Sphere from the Block.</p>	

 <b>Undo</b> until nothing is picked   and then <b>Pick Bodies</b> again	<p>Using the mouse pick each of the three bodies by double-clicking the left mouse button as each body is high-highlighted. Start by picking the Cone first and then the Block and the Sphere.</p>	
<p>Use the right mouse button to view context menu and select:  <b>Combine Bodies &gt; Subtraction, with regularization</b></p>		
	<p>By picking the bodies in a different order a very different resultant body will be formed. The subtraction operation subtracts the second and third body from the first.</p>	

The **Undo** button  can be used to return the solid model back to the point before the **Subtraction** was applied, or the primitives can be re-created to allow the next example to be completed.

## Subtraction without Regularization

The **Subtraction** operation can also be applied without regularization. The resultant body will then consist of a subtraction of the second and subsequent bodies from the first one picked plus any faces that lie on the surface of the removed parts. This allows the user to leave behind faces that can then have properties set on them.

 <b>Pick Bodies</b>	<p>Using the mouse pick two of the three bodies by double-clicking the left mouse button as each body is high-lighted. Start by picking the Cone first and then the Block and the Sphere.</p>	
<p>Use the right mouse button to view context menu and select:  <b>Combine Bodies &gt; Subtraction, without regularization</b></p>		
	<p>The subtraction operation subtracts the second and third body from the first, but as the bodies are not regularized, the face that lay on the surface of the block on the end of the cone has been left. This face can now be used to apply local mesh refinement etc. as it can have properties assigned to it.</p>	

Before moving to the next section, clear all the data using:



**Clear All**

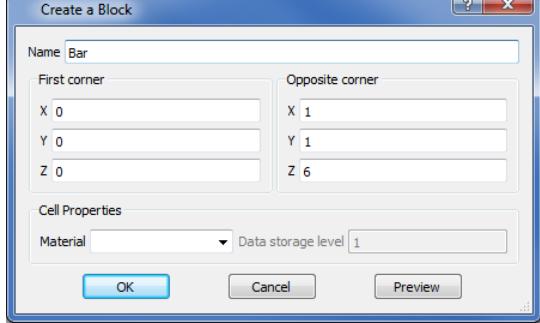
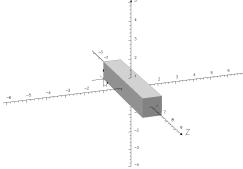
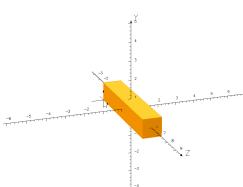
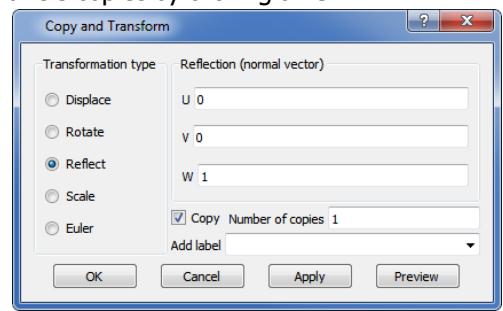
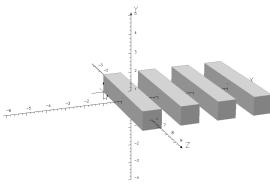
## Trim Overlap

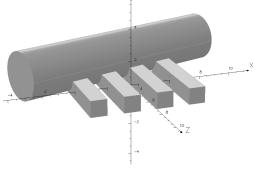
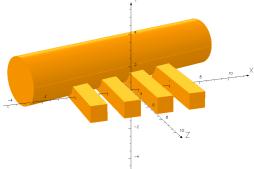
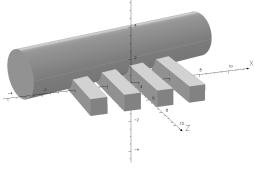
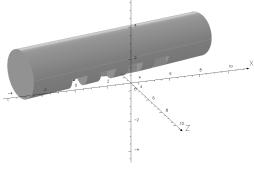
**Trim Overlap** allows the user to remove parts of bodies that overlap and still leave the original bodies. In effect the command issues a **Subtraction** but takes a copy of the original subtracting bodies first and replaces these once the **Subtraction** is complete.

The **Trim Overlap** operation is useful when adding volumes around regions of interest. A new body can be created and picked first, followed by the other bodies that overlap, then when the **Trim Overlap** command is issued the new volume will fill the gaps around the area of interest.

**Trim Overlap** is used to cut several bodies from one body.

The following example uses the **Copy** operation which is described in more detail in [Making Replications of a Body \[page 115\]](#).

 <b>Create a Block</b>	 <p><b>Name:</b> Bar  <b>First corner:</b> X: 0, Y: 0, Z: 0  <b>Opposite corner:</b> X: 1, Y: 1, Z: 6  <b>Cell Properties:</b>  <b>Material:</b> [dropdown]  <b>Data storage level:</b> 1  <input type="button" value="OK"/> <input type="button" value="Cancel"/> <input type="button" value="Preview"/></p>	
 <b>Pick Bodies</b>	<p>Move the mouse over the body to high-light it.      Pick it with the right mouse button, and from view context menu select: <b>Copy and Transform</b></p>	
	<p>Displace by 2 length units in U direction and make 3 copies by clicking on <b>OK</b>.</p>  <p><b>Transformation type:</b> Reflection (normal vector)  <input checked="" type="radio"/> <b>Displace</b>: U: 0, V: 0, W: 1  <input type="radio"/> <b>Rotate</b>  <input type="radio"/> <b>Reflect</b>  <input type="radio"/> <b>Scale</b>  <input type="radio"/> <b>Euler</b>  <input checked="" type="checkbox"/> <b>Copy</b>: Number of copies: 1  <input type="button" value="OK"/> <input type="button" value="Cancel"/> <input type="button" value="Apply"/> <input type="button" value="Preview"/></p>	

 <b>Create a Cylinder</b>	<p><b>Create a Cylinder</b></p> <p>Name: Cylinder3</p> <table border="1"> <tr> <td>Centre of base</td> <td>Centre of top</td> </tr> <tr> <td>X: 10</td> <td>X: -3</td> </tr> <tr> <td>Y: 1.5</td> <td>Y: 1.5</td> </tr> <tr> <td>Z: -2</td> <td>Z: 2</td> </tr> </table> <p><input checked="" type="radio"/> Cylinder  <input type="radio"/> Tube  <input type="radio"/> Cone  <input type="radio"/> General</p> <p><input type="checkbox"/> Make n-sided prism Number of sides: 10</p> <p>Cell Properties  Material: [dropdown] Data storage level: [dropdown]</p> <p>OK Cancel Preview</p>	Centre of base	Centre of top	X: 10	X: -3	Y: 1.5	Y: 1.5	Z: -2	Z: 2	
Centre of base	Centre of top									
X: 10	X: -3									
Y: 1.5	Y: 1.5									
Z: -2	Z: 2									
 <b>Pick Bodies</b>	<p>Using double-click with the left mouse button, pick the cylinder first, then each of the four bars. The order (cylinder first) is important, because the first one picked is the body which will be changed by the <b>Trim</b> operation.</p>									
<p>Use the right mouse button to view the context menu and select:  <b>Combine Bodies &gt; Trim overlap, with regularization</b> <sup>a</sup></p>										
	<p>The resulting geometry still contains 5 bodies, but the overlap between the cylinder and the 4 bars has now been removed from the cylinder. The picture does not look much different yet, but ...</p>									
 <b>Hide Bodies</b>	<p>... by hiding the 4 bars one by one, the result of the operation can be seen more clearly. The 4 bars have been cut out of the cylinder. Try unhiding the bars and then hide the cylinder: the 4 bars are intact.</p>									

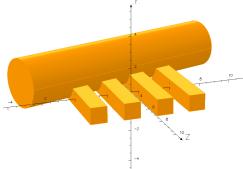
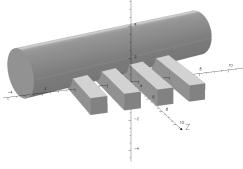
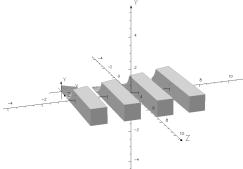
<sup>a</sup> Note that the choice of with or without regularization does not affect this example.

## Cutaway Overlap

**Cutaway Overlap** is very similar to the previously discussed **Trim Overlap** operation. The difference is that the cylinder is now used as a cutting tool to cut the overlapping parts from the 4 rectangular bars.

**Cutaway Overlap** is used to cut one body from several bodies.

Again the cylinder has to be picked first, followed by the other bodies that overlap, then the **Cutaway Overlap** command can be issued. Start with the previous model of 4 bars and a cylinder.

 <b>Pick Bodies</b>	<p>Using double-click with the left mouse button, pick the cylinder first, then each of the four bars. The order (cylinder first) is important, because the second and subsequent bodies picked will be changed by the <b>Cutaway</b> operation.</p>	
<p>Use the right mouse button to view the context menu and select:  <b>Combine Bodies &gt; Cutaway overlap, with Regularization</b><sup>a</sup></p>		
	<p>The resulting geometry still contains 5 bodies, but the overlap between the cylinder and the 4 bars has now been removed from the bars. The picture does not look much different yet, but ...</p>	
 <b>Hide bodies</b>	<p>... by hiding the cylinder, the result of the operation can be seen more clearly. The cylinder has been cut out of the bars.      Try unhiding the cylinder and then hide the bars: the cylinder is intact.</p>	

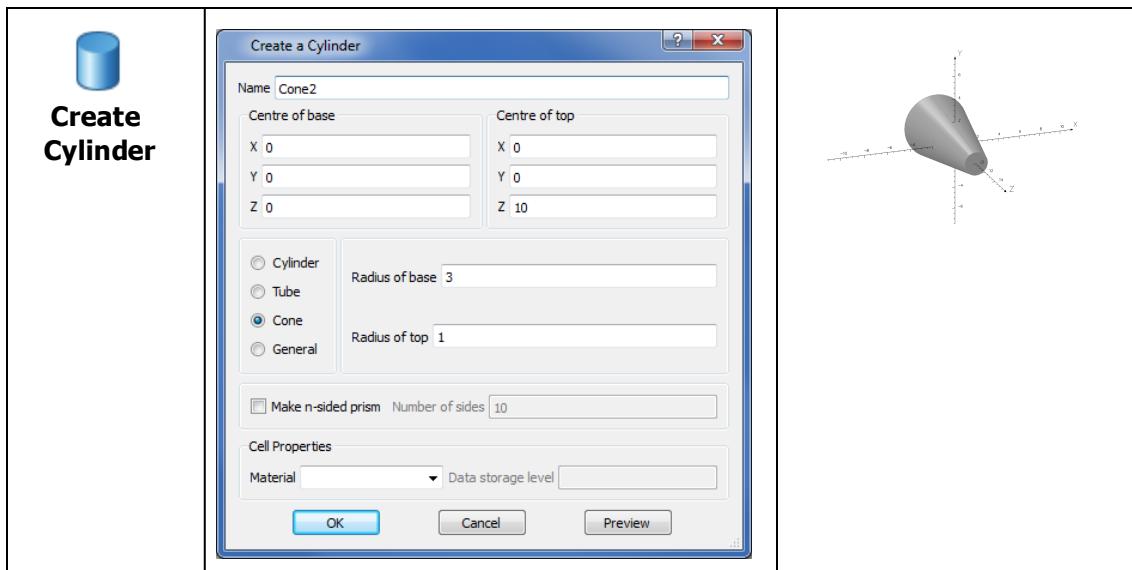
<sup>a</sup> Note that the choice of with or without regularization does not affect this example.

## Transforming and Copying Objects

Bodies can be moved in a number of different ways. For example the **Transform** operation can be used to change the position or shape of a body. The **Copy** operation allows multiple instances of bodies, cells, faces and edges to be created.

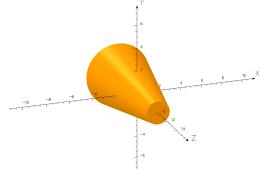
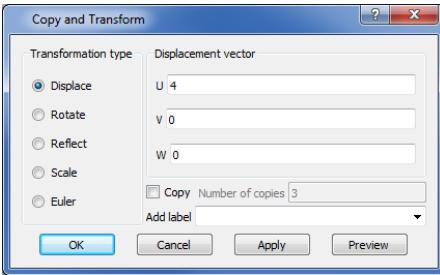
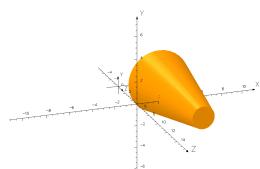
The **Copy** and **Transform** operations use coordinate axes and directions called U, V and W. These correspond to the local X, Y and Z axes of the working coordinate system (see [Local Coordinate Systems \[page 143\]](#)).

In this example, a simple cone is created. The **Transform** and **Copy** operations are then demonstrated.



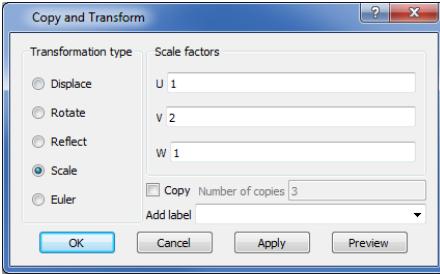
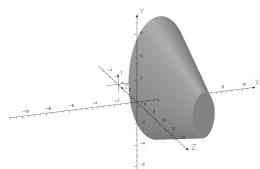
### Displacement

The cone will now be moved to another position.

 <b>Pick Bodies</b>	<p>Pick the cone with the right mouse button, and select <b>Transform</b> from the context menu.</p>	
	<p>Specify a displacement by 4 units in the U direction.</p>  <p>Use <b>Apply</b>, to perform the displacement operation and keep the object picked.</p>	

## Changing the Shape of a Body

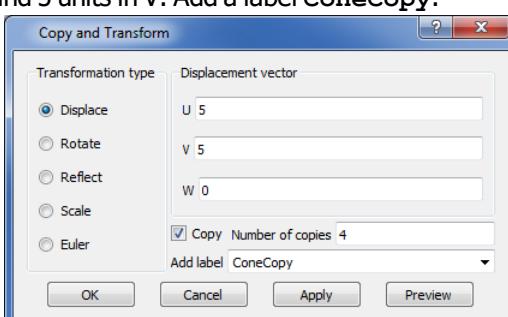
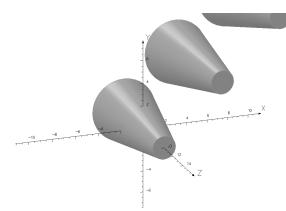
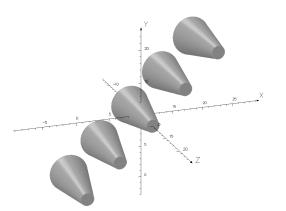
Under the **Transform** options, the shape of the body may also be altered using anisotropic scaling.

	<p>Scale the body by a factor of 2 in the V direction, leaving U and W unchanged.</p>  <p>Use <b>OK</b> this time, to unpick the object after making the transformation.</p>	
--	---	---

Select the **Undo** button  twice to return the picked cone to its original position. Note that two separate transformations (**Displace** and **Scale**) have been used, so two applications of **Undo** are required.

## Making Replications of a Body

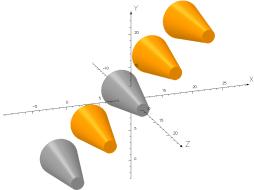
Many complex geometric models consist of repeated copies of a simple set of objects. A single instance of the set can be defined and multiple copies made.

	<p>Pick the cone with the right mouse button and then select: <b>Copy</b> from the context menu.</p> <p>Make 4 new copies separated by 5 units in U and 5 units in V. Add a label <b>ConeCopy</b>.</p>  <p>Use <b>OK</b> to replicate the object and unpick the original cone.</p>	
 Initialize the view	<p>All the copies can now be seen.</p>	

## Names, Unique Names and Labels

It is important to note that the model now contains 5 bodies. All are called **cone2** but each one has a unique name. The unique names can be seen in the **Pick by Property** dialog. The unique names are generated automatically for each entity. They are needed for the REPLAY functionality. It is not recommended to use unique names for picking in scripts - pick by names, labels or coordinates instead.

The example above shows how to add the label **ConeCopy** to the cones. The original cone gets the label **ConeCopy**, the other 4 copies will be labelled **ConeCopy1**, **ConeCopy2**, **ConeCopy3** and **ConeCopy4**.

 <b>Pick Entities by Property</b>	<p>Expand the branches of the tree by clicking on the + next to <b>Labels</b>.</p> <p><b>Pick Items</b></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Property</th> <th style="width: 70%;">Colour</th> </tr> </thead> <tbody> <tr> <td>+ Coordinate System Name</td> <td></td> </tr> <tr> <td>+ Element Shape Preference</td> <td></td> </tr> <tr> <td>+ Element type</td> <td></td> </tr> <tr> <td>+ Information</td> <td></td> </tr> <tr> <td>+ Labels</td> <td></td> </tr> <tr> <td>   - ConeCopy</td> <td style="background-color: #800080;"></td> </tr> <tr> <td>   - ConeCopy1</td> <td style="background-color: #008000;"></td> </tr> <tr> <td>   - ConeCopy2</td> <td style="background-color: #00A0A0;"></td> </tr> <tr> <td>   - ConeCopy3</td> <td style="background-color: #00A0A0;"></td> </tr> <tr> <td>   - ConeCopy4</td> <td style="background-color: #808000;"></td> </tr> <tr> <td>+ Layering Method, Backward</td> <td></td> </tr> <tr> <td>+ Layering Method, Forward</td> <td></td> </tr> <tr> <td>+ Material</td> <td></td> </tr> <tr> <td>+ Model part</td> <td></td> </tr> <tr> <td>+ Name</td> <td></td> </tr> <tr> <td>+ Name (unique)</td> <td></td> </tr> </tbody> </table> <p>Add      Toggle      Remove      Close</p> <p>Click on <b>Toggle</b> to pick different labels and see the result. Finally, <b>Close</b> the dialog.</p>	Property	Colour	+ Coordinate System Name		+ Element Shape Preference		+ Element type		+ Information		+ Labels		- ConeCopy		- ConeCopy1		- ConeCopy2		- ConeCopy3		- ConeCopy4		+ Layering Method, Backward		+ Layering Method, Forward		+ Material		+ Model part		+ Name		+ Name (unique)		
Property	Colour																																			
+ Coordinate System Name																																				
+ Element Shape Preference																																				
+ Element type																																				
+ Information																																				
+ Labels																																				
- ConeCopy																																				
- ConeCopy1																																				
- ConeCopy2																																				
- ConeCopy3																																				
- ConeCopy4																																				
+ Layering Method, Backward																																				
+ Layering Method, Forward																																				
+ Material																																				
+ Model part																																				
+ Name																																				
+ Name (unique)																																				

## Copying Cells, Faces and Edges

It is possible to copy cells, faces and edges with the following effects.

- If 2 overlapping bodies are **Combined** with a **Union without Regularization**, the overlapping cell will be retained. It can be picked and copied to a new location to create a new body.
- Copying a face will create a sheet face.
- Copying an edge will create a wire edge.

## Sweeping Faces

Sweeping a face provides another way of creating geometry. There are 4 types of sweep operation:

- **Distance** sweeps in a direction normal to the face;
- **Vector** sweeps along a vector specified by displacements in U, V and W;
- **Rotate** sweeps by an angle around a specified axis;
- **Along a path** uses a wire edge to specify the sweep direction and extent.

With each type of sweep operation, other options can be specified:

- The sides of the extrusion can be kept parallel or they can be allowed to diverge or converge by setting a draft angle.
- The original face can either be left in place or removed. Leaving the original face in place means that the extruded volume produces a new cell within the body; removing it means that the existing cell is extended.
- The final face can be forced to be parallel to the original face or can be orthogonal to the sweep path.
- The swept body can be twisted around the sweep path.

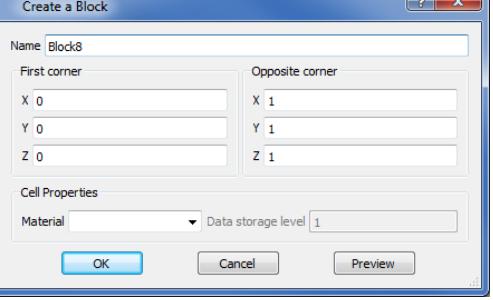
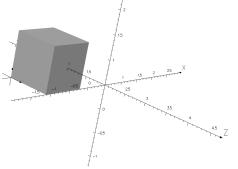
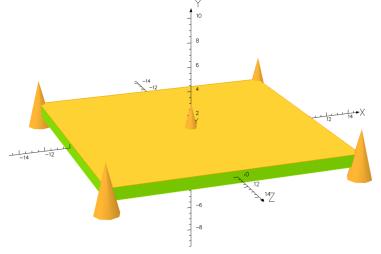
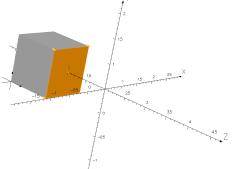
Sweep operations use coordinate axes and directions called U, V and W. These correspond to the local X, Y and Z axes of the working coordinate system (see [Local Coordinate Systems \[page 143\]](#)).

The following examples illustrate some of these capabilities.

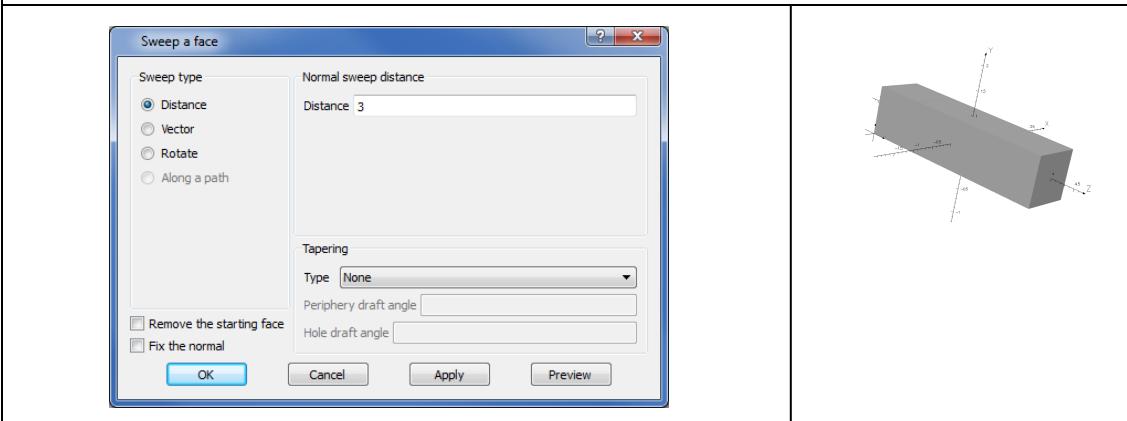
### Sweep a Face through a Distance

Sweeping a face through a distance allows the starting face to be extruded in a linear direction that is normal to the face. This option is only available for planar faces.

Before sweeping, it is a good idea to check the direction of the normal to the face. The outward normal should point in the sweep direction. The **Vectors** dialog can be used to select display of **Face normal** vectors on picked faces.

 <b>Create Block</b>	 <p>Start with a simple block.</p>	
 <b>Vectors</b>	<p>Select <b>Face normal (picked faces only)</b> so that the normal direction will be displayed using vectors when the face is picked for sweeping.</p> 	
 <b>Pick Faces</b>	<p>Ensure that the <b>Pick Entity</b> button has been selected and then pick the front face (at z=1) using the right mouse button.</p>	

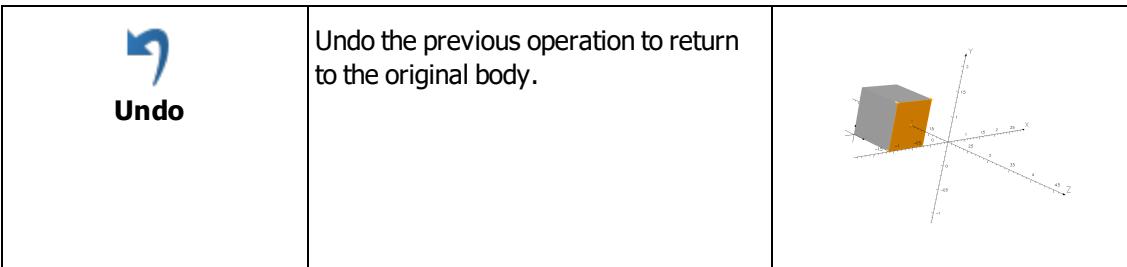
Select **Sweep Face** from the context menu and enter a **Distance**.



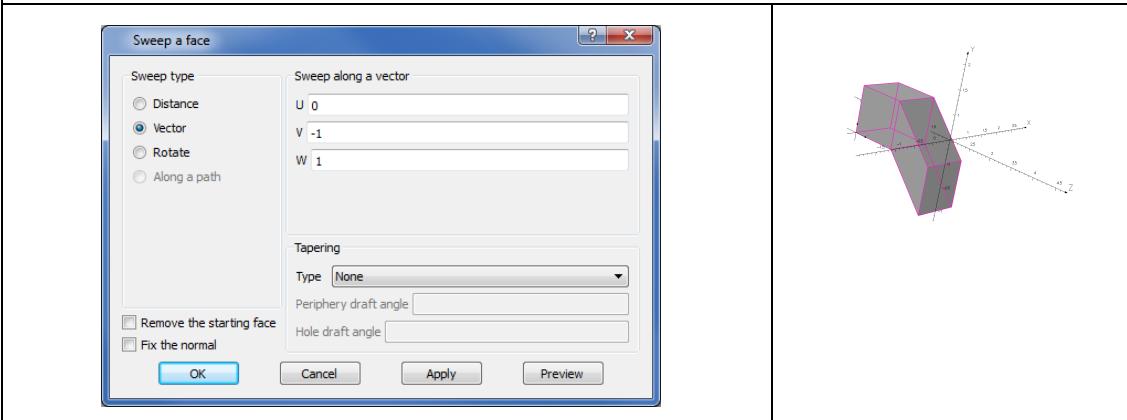
By not removing the starting face, the body is now made up of two cells which could then have different material labels or other properties applied to them.

## Sweep a Face along a Vector

Sweeping a face along a vector also produces a linear extrusion but this time there is more control over the extrusion direction.



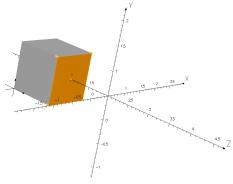
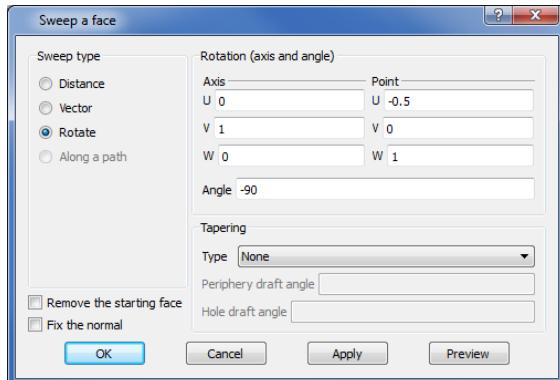
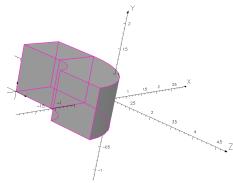
Select **Sweep Face** from the context menu and specify the vector in terms of U,V, and W.



The face has now been swept along the defined vector from the position of the starting face to produce the new shape. Again as the starting face has been left the body now consists of two cells.

## Sweep a Face by Rotation

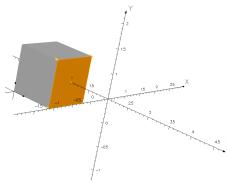
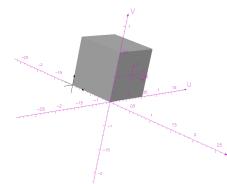
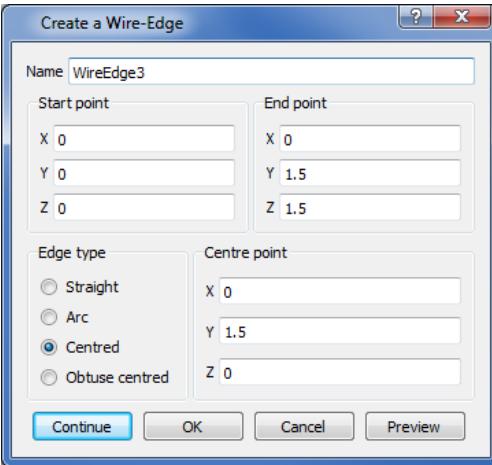
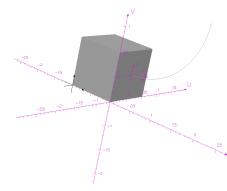
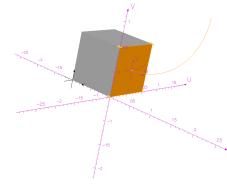
Sweeping a face by rotation is a powerful command that allows complex shapes to be produced.

 <b>Undo</b>	Undo the previous operation to return to the original body.	
<p>Select <b>Sweep Face</b> from the context menu and specify the orientation of the axis as a UVW vector and a point on the axis as UVW coordinates.</p>		
		

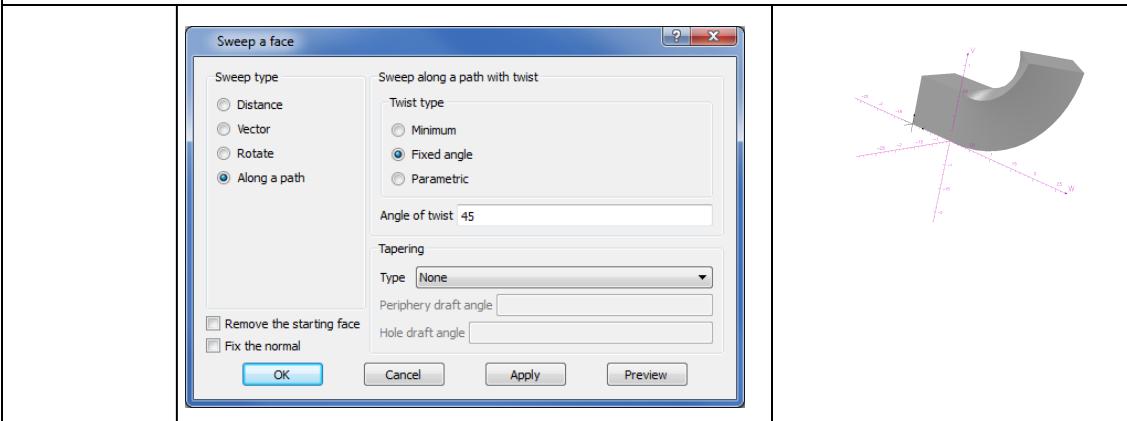
The face is swept through an angle of -90 degrees around a line parallel to the V-axis through a point that lies at -0.5, 0, 1.

## Sweep a Face along a Path

Sweep along a path with twist is a very useful command that allows very complex shapes to be produced. In this example a twisted sweep is performed along a curved edge which starts in the middle of the front face. Moving the working coordinate system to the centre of the face makes this easier.

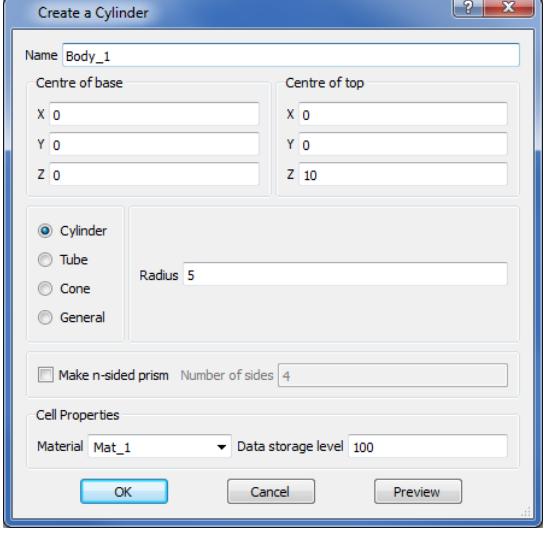
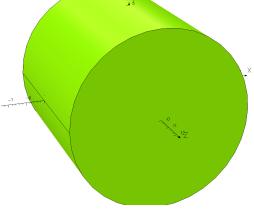
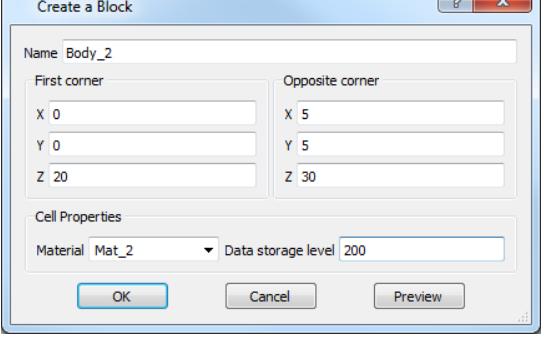
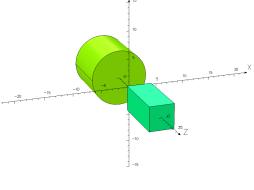
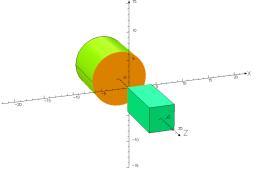
 <b>Undo</b>	<p>Undo the previous operation to return to the original body.</p>	
 <b>Coordinate Systems &gt; Move WCS</b>	<p>This moves the working coordinate system to the centre of the picked face.</p>	
 <b>Wire-Edge</b>	<p>Define a <b>Centred</b> arc by its end points and centre of curvature.</p> <p>The dialog box shows the following settings:  Name: WireEdge3  Start point: X: 0, Y: 0, Z: 0  End point: X: 0, Y: 1.5, Z: 1.5  Edge type: Centred (radio button selected)  Centre point: X: 0, Y: 1.5, Z: 0  Buttons: Continue, OK, Cancel, Preview</p> <p>Press <b>OK</b> to close the dialog</p>	
 <b>Pick Edges</b>	<p>Pick the front face (at z=1) again and then change picking to edges. Move the mouse over the wire-edge until it high-lights, then bring up the context menu with the right mouse button. Select <b>Sweep Face</b> from the context menu.</p>	

In the **Sweep Face** dialog, select the option to sweep **Along a path**. The face will be swept along the edge, with a fixed twist angle of 45 degrees.

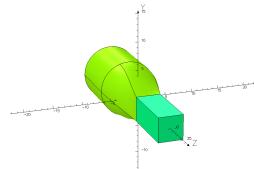
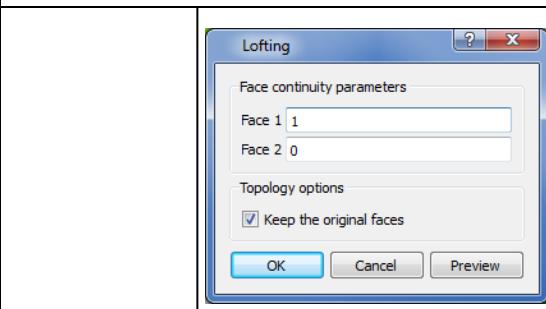


## Lofting Between Surfaces

Lofting is a modelling technique that creates a new volume between two picked faces.

 <b>Create Cylinder</b>	 <p><b>Create a Cylinder</b></p> <p>Name: Body_1</p> <p>Centre of base:</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 0</li> </ul> <p>Centre of top:</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 10</li> </ul> <p><input checked="" type="radio"/> Cylinder  <input type="radio"/> Tube  <input type="radio"/> Cone  <input type="radio"/> General</p> <p>Radius: 5</p> <p><input type="checkbox"/> Make n-sided prism Number of sides: 4</p> <p>Cell Properties:</p> <p>Material: Mat_1 Data storage level: 100</p> <p>OK Cancel Preview</p>	
 <b>Create Block</b>   <b>Initialize the View</b>	<p>Make a block with the <b>Block</b> command, extending from Z=20 to Z=30. Then press the <b>Initialize View</b> button.</p>  <p><b>Create a Block</b></p> <p>Name: Body_2</p> <p>First corner:</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 20</li> </ul> <p>Opposite corner:</p> <ul style="list-style-type: none"> <li>X: 5</li> <li>Y: 5</li> <li>Z: 30</li> </ul> <p>Cell Properties:</p> <p>Material: Mat_2 Data storage level: 200</p> <p>OK Cancel Preview</p>	
 <b>Pick Faces and Pick Entity</b>	<p>Pick the end face of the cylinder at Z=10, then pick the end face of the block at Z=20.</p>	

Use the right mouse button to view context menu and select:



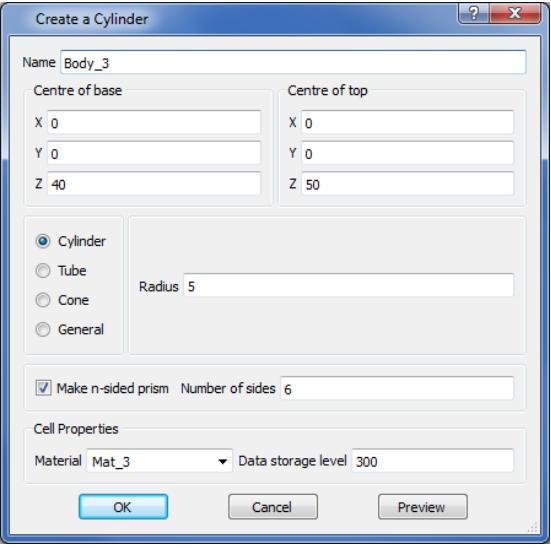
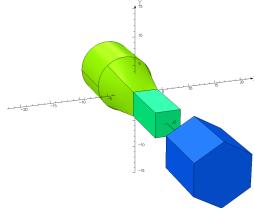
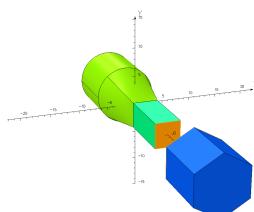
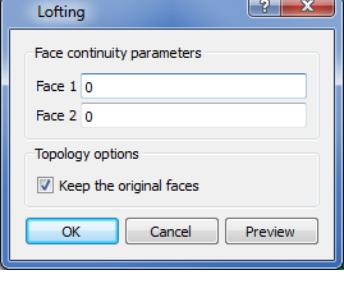
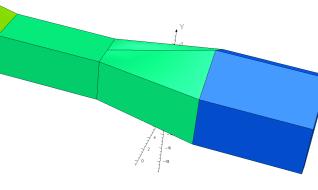
A different face tangent parameter can be used for each face. The parameters do not have to be integers.

With the option to keep the original faces set (**KEEP=YES**) , the material properties of the new cell in the middle is determined by the order of picking the faces. Although the data storage level of the block is higher in this example, the cell properties of the cylinder are passed on to the new cell.

The shape of the new lofted cell does not depend only on the 2 lofted faces, but also on the faces to which the lofted faces are joined. This can be demonstrated when lofting to a sheet face, and the face tangent parameter for the sheet face is set to either 0 or 2 - it makes a big difference to the new cell.

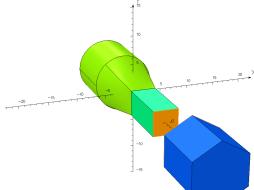
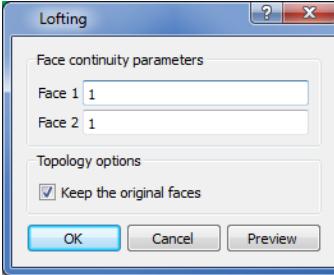
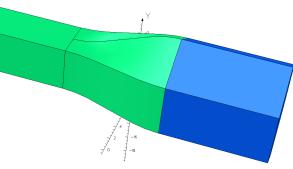
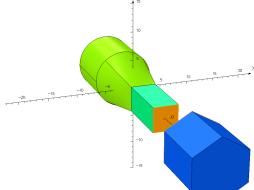
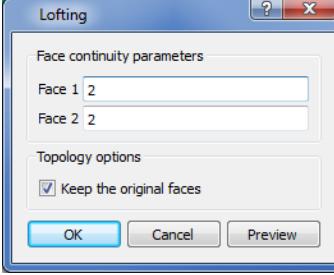
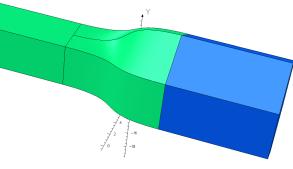
## Face Tangent Parameters

Different face tangent parameters can be tried to understand the effects on the newly created volume. **FACE1TANGENT** sets the smoothness for the face selected first and **FACE2TANGENT** for the face selected second.

 <b>Create Cylinder</b>	<p>Make a hexagon with the <b>Cylinder</b> command, extending from Z=40 to Z=50.</p> 	
 <b>Pick Faces and Pick Entity</b>	<p>Pick the end face of the block at Z=30, then pick the end face of the hexagon at Z=40.</p>	
<p>Use the right mouse button to view context menu and select:</p>		
	<p>With a face tangent of 0, the 2 faces are connected with straight edges.</p> 	

Note that with a face tangent of 0, the new edges look like straight edges. Internally they are however represented as "curved edges", which can lead to surprising effects when creating regular volumes that qualify for hexahedral meshing.

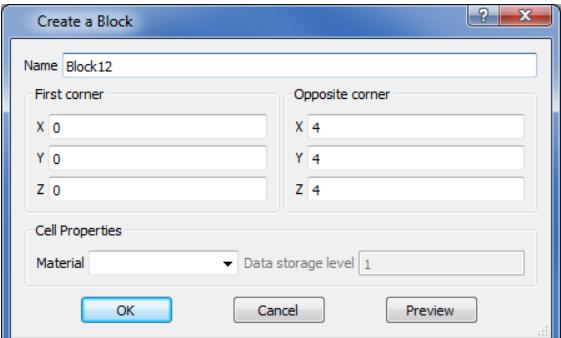
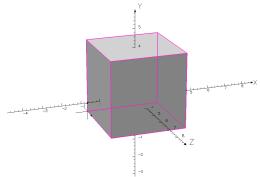
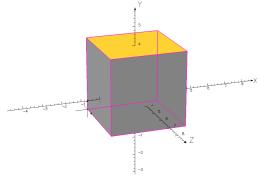
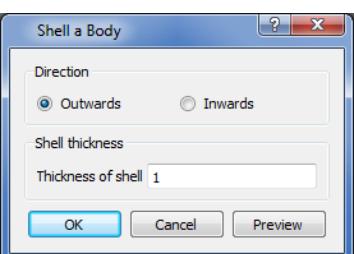
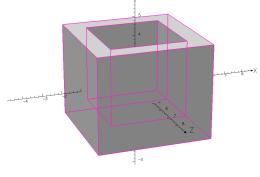
To conclude this section, face tangents of 1 and of 2 are applied.

 <b>Undo</b>	Undo until the 2 faces at Z=30 and Z=40 are picked.	
Use the right mouse button to view context menu and select:  <b>Loft</b>		
		
 <b>Undo</b>	<b>Undo</b> the Loft operation. (Rather than applying the Loft with the <b>OK</b> button, the <b>Preview</b> button could have been used to try the effects of different face parameters.)	
Use the right mouse button to view context menu and select:  <b>Loft</b>		
		

## Shell and Offset

### Shell

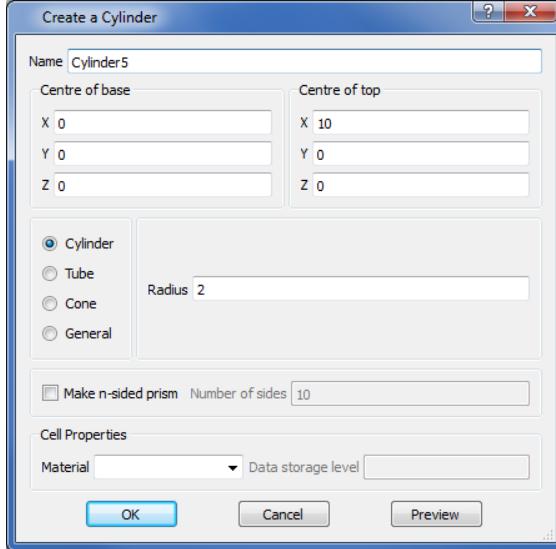
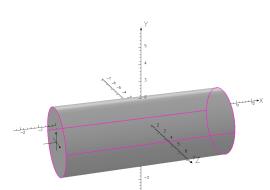
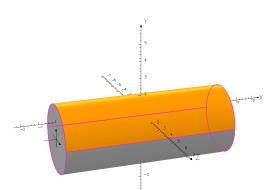
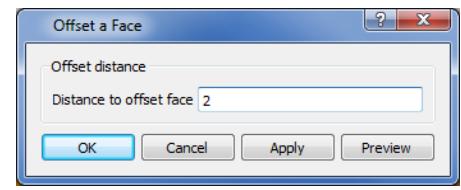
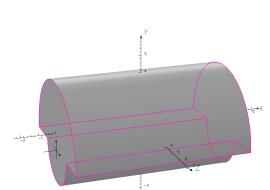
The **Shell** operation removes selected faces of a body. The remaining faces are then swept either toward the centre of the original body, or outwards by a defined distance.

 <b>Create Block</b>		
 <b>Pick Faces</b>  <b>Pick Entity</b>	Pick one of the faces of the block using the right mouse button.	
Select:  <b>Shell</b> from the context menu.		
		

### Offset

The **Offset** operation extends a face to create a body with a new thickness. It behaves in a similar way to the Face sweeping operation, except that **Offset** can also be applied to curved faces.

The face selected can either be a face of an existing body, or can be a zero-thickness face for which a thickness is to be assigned.

 <b>Create Cylinder</b>		
 <b>Pick Faces</b>  <b>Pick Entity</b>	<p>Pick one of the curved faces of the cylinder with the right mouse button.</p>	
<p>Select:  <b>Offset</b> from the context menu.</p>		
		

## Blend and Chamfer

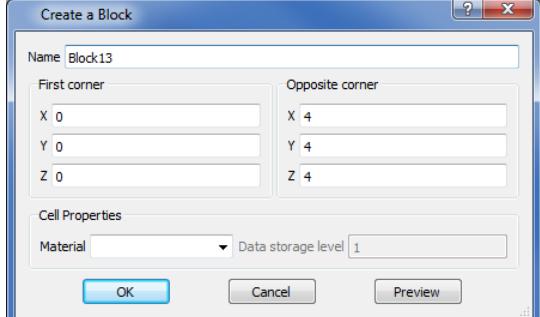
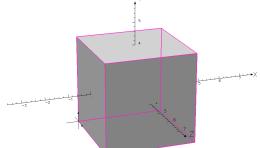
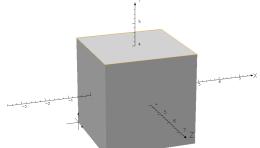
**Blend** and **Chamfer** are operations which can be applied to edges of a volume.

- The **Blend** operation smooths an edge using a radius.
- The **Chamfer** operation cuts off an edge by using a plane. The **Chamfer** operation can also be applied to a vertex.

If the operations are applied to an exterior edge, material will be cut away; if there is an interior edge, material will be added. This gives a total of 4 combinations, all of which will be demonstrated in the following examples.

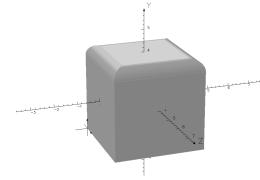
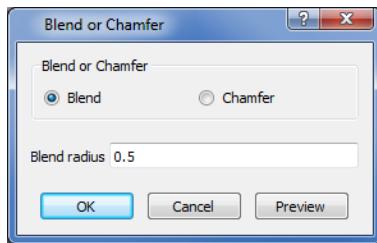
There are restrictions on which edges and vertices can be blended or chamfered. See "Non-Manifold Edges" on page 132.

### Blending Exterior Edges

 <b>Create Block</b>	<p>Create a Block with the opposite corners at (0, 0, 0) and (4, 4, 4).</p> 	
 <b>Pick Edges and Pick Entity</b>	<p>Pick the 4 edges on top of the block (<math>y=4</math>). The colour of the edges will change as they are picked.</p>	

Use the right mouse button to view context menu and select:  **Blend or Chamfer**

Create a **Blend** with radius 0 . 5.

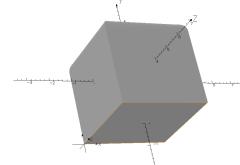


## Chamfering Exterior Edges

In the next operation the bottom of the block will be chamfered. The **Second chamfer distance** is chosen to be the same as the first, so that a 45 degree section will be cut off the block. Specifying different values would result in an asymmetric chamfer.

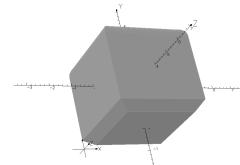
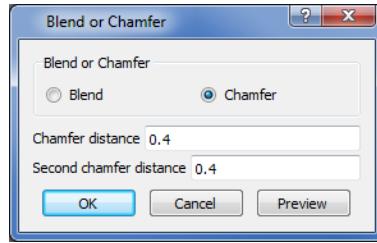
-  **Pick Entity**
-  **Pick Edges**

Rotate the block using the mouse, and select the 4 edges at the bottom of the block. The colour of the edges will change as they are picked.



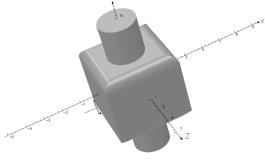
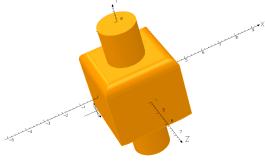
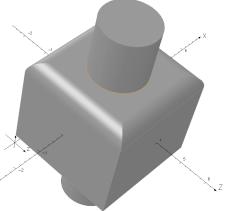
Use the right mouse button to view the context menu and select: **Blend or Chamfer**.

Create a **Chamfer** with distances of 0 . 4.



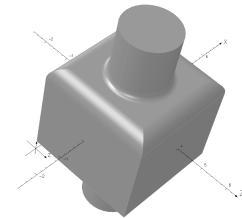
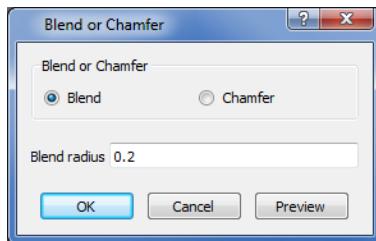
## Blending Interior Edges

A cylinder will now be defined to cut through the block. This will form interior edges which will be used for **Blend** and **Chamfer**.

 <b>Create Cylinder</b>	<p><b>Create a Cylinder</b></p> <p>Name: Cylinder7</p> <p>Centre of base:</p> <ul style="list-style-type: none"> <li>X: 2</li> <li>Y: -2</li> <li>Z: 2</li> </ul> <p>Centre of top:</p> <ul style="list-style-type: none"> <li>X: 2</li> <li>Y: 6</li> <li>Z: 2</li> </ul> <p><input checked="" type="radio"/> Cylinder  <input type="radio"/> Tube  <input type="radio"/> Cone  <input type="radio"/> General</p> <p><input type="checkbox"/> Make n-sided prism Number of sides: 10</p> <p>Cell Properties:</p> <p>Material: [dropdown] Data storage level: [dropdown]</p> <p>OK Cancel Preview</p>	
 <b>Pick Entity</b>   <b>Pick Bodies</b>	<p>Pick the cylinder and the block using the mouse. The sequence is not important as the next operation will be a <b>Union, with Regularization</b>. Note that it is necessary to choose <b>Union, with Regularization</b> so that only a single cell is created, and to ensure that no edges are non-manifold.</p>	
<p>Use the right mouse button to view context menu and select:  <b>Combine Bodies &gt; Union, with regularization</b>.</p>		
 <b>Pick Edges</b>	<p>Pick the edge on top of the cube, where the cylinder meets the cube, as indicated on the right picture.</p> <p>It does not matter whether a 180 degrees section or both 180 degrees sections are selected; the program will always perform the operation on the whole circle.</p>	

Use the right mouse button to view context menu and select:  **Blend or Chamfer**.

Create a **Blend** with radius 0 . 2.



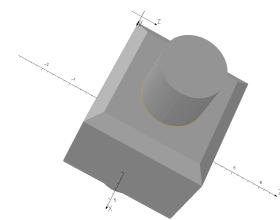
## Chamfering Interior Edges

The last operation will be a **Chamfer** of the bottom junction. Use the mouse to rotate the body and pick the edge where the cylinder meets the cube.



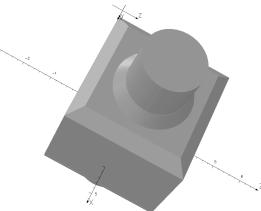
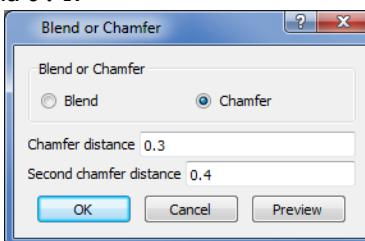
**Pick Edges**

Pick the edge at the bottom of the cube, where the cylinder meets the cube, as indicated on the right picture.



Use the right mouse button to view context menu and select: **Blend or Chamfer**.

Create an asymmetric **Chamfer** with distances 0 . 3 and 0 . 4.



## Non-Manifold Edges

Edges which can be blended or chamfered must belong to only one body and only one cell and can only be shared by two faces. Edges which do not satisfy these conditions are called "non-manifold". Similarly, vertices to be chamfered must belong to only one body and only one cell.

This can be demonstrated by using **Undo**  to return to the Boolean **Union** in **Blending Interior Edges** and changing to a **Union, without regularization**. This will create 4 cells in the body and the edge at the junction of the cylinder and block will now belong to four faces. When the chamfer or blend is applied the user will receive the following error message:

Error reported applying chamfers: non-manifold edge or vertex in or near blend.

## Morphing Operations

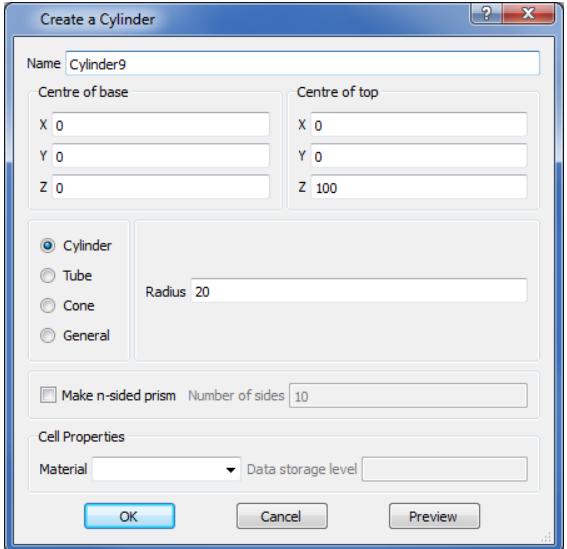
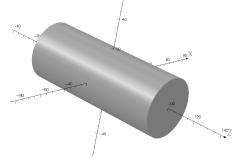
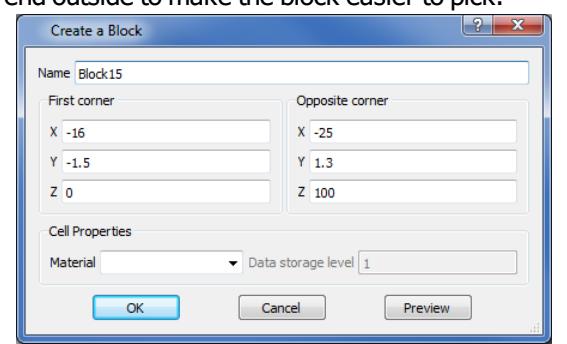
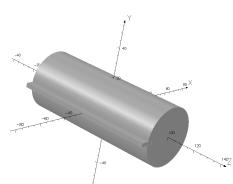
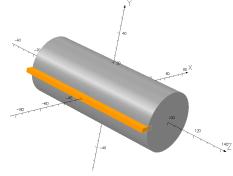
---

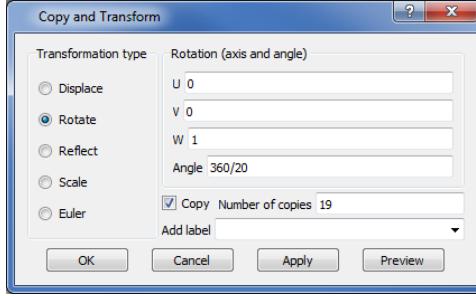
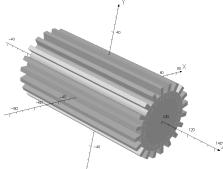
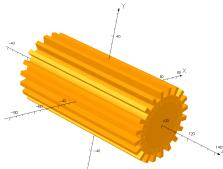
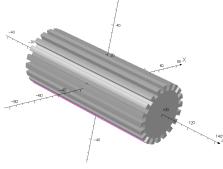
Morphing allows a variety of geometric modifications to be made to selected bodies:

- [TWIST Command \[page 136\]](#),
- [STRETCH Command \[page 138\]](#),
- [BEND Command \[page 139\]](#) and
- [MORPH Command \[page 140\]](#) (new coordinates expressed as algebraic functions of the old).

The model used to illustrate the morphing operations is a slotted cylinder, chosen because the slots make it easy to see the effects of twisting and stretching.

Create a cylinder and a slot in the cylinder as follows.

 <b>Create Cylinder</b>	<p>Define a cylinder with length 100, radius 20:</p>  <p><b>Create a Cylinder</b></p> <p>Name: Cylinder9</p> <p>Centre of base</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 0</li> </ul> <p>Centre of top</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 100</li> </ul> <p><input checked="" type="radio"/> Cylinder</p> <p><input type="radio"/> Tube</p> <p><input type="radio"/> Cone</p> <p><input type="radio"/> General</p> <p>Radius: 20</p> <p><input type="checkbox"/> Make n-sided prism Number of sides: 10</p> <p>Cell Properties</p> <p>Material: [dropdown] Data storage level: [dropdown]</p> <p>OK Cancel Preview</p>	
 <b>Create Block</b>	<p>Define a block which will be used to cut a slot. The x-coordinates start inside the cylinder and end outside to make the block easier to pick.</p>  <p><b>Create a Block</b></p> <p>Name: Block15</p> <p>First corner</p> <ul style="list-style-type: none"> <li>X: -16</li> <li>Y: -1.5</li> <li>Z: 0</li> </ul> <p>Opposite corner</p> <ul style="list-style-type: none"> <li>X: -25</li> <li>Y: 1.3</li> <li>Z: 100</li> </ul> <p>Cell Properties</p> <p>Material: [dropdown] Data storage level: 1</p> <p>OK Cancel Preview</p>	
 <b>Pick Entity</b>  <b>Pick Bodies</b>	<p>Pick the block to make copies: use the right mouse button to view context menu and select: <b>Copy</b>.</p>	

	<p>Make 19 copies (20 blocks in total). Rotational angle is <math>360/20</math>.</p>  <p>Select <b>OK</b>.</p>	
 <b>Pick Bodies</b>	<p>Subtract the blocks from the cylinder to create a slotted body.</p> <p>Pick the cylinder first, and then select all the blocks using the <b>Pick All Filter Type Entities</b> button, .</p>	
<p>Use the right mouse button to view context menu and select: <b>Combine Bodies &gt; Subtraction, with regularization</b>.</p>		

## TWIST Command

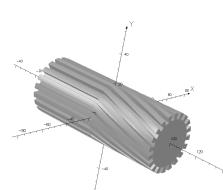
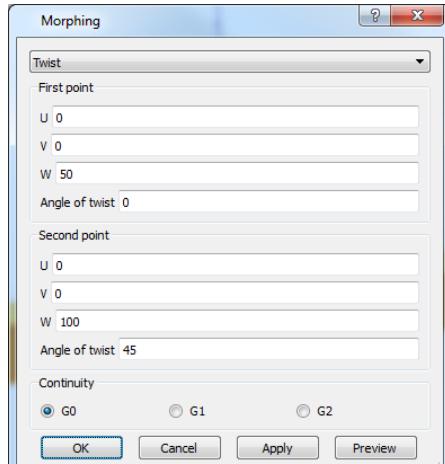


The **TWIST** command ( **Morph**) transforms the coordinates of a selected body by twisting it around a vector between two points. The angle of twist at each point can be given and different levels of surface continuity can be specified as **G0**, **G1**, or **G2** (representing zero, first and second order distortion of the twist surface respectively).

The **Twist** option will be used to twist the cylinder. The cylinder will be twisted by an angle of 45 degrees, starting half way along its length.

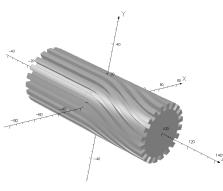
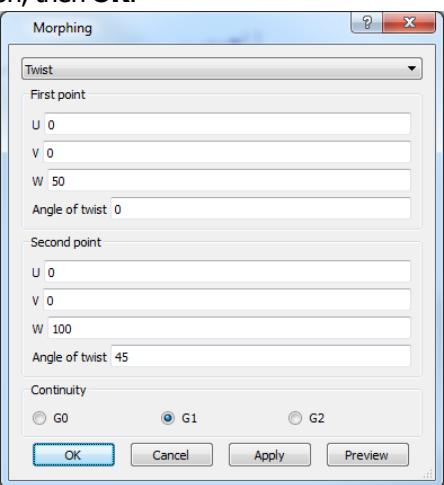
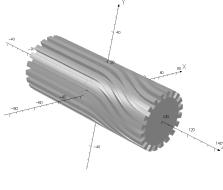
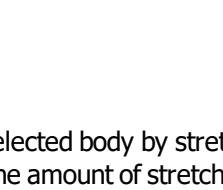
Use the right mouse button to pick the cylinder and select  **Morph** from the context menu.

Select the Twist option. Start twisting at z=50, with angle 45 degrees. Set **G0** for order of continuity. Select **Preview**.



The effect of different orders of continuity can be seen by repeating with different parameters. First

 **Undo** the **G0** twist.

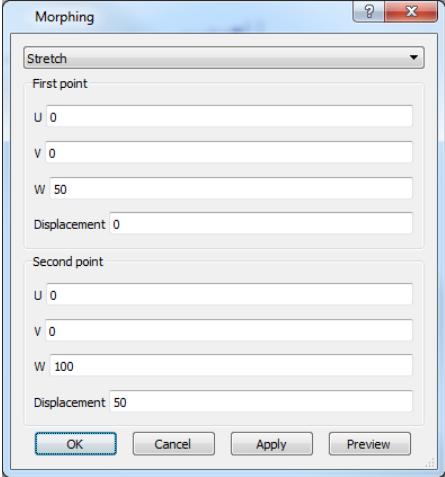
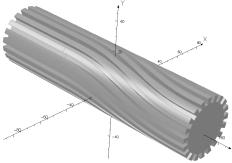
	<p>Use the right mouse button to view context menu and select:  <b>Morph</b></p>	
	<p>Ensure the Twist option is still set. Select <b>G1</b> option, then <b>OK</b>.</p> 	
	<p>Undo again, then use the right mouse button to view context menu and select: <b>Morph</b>. This time select <b>G2</b> option, then <b>OK</b>.</p>	

## STRETCH Command



The **STRETCH** command ( **Morph**) transforms the coordinates of a selected body by stretching one point relative to another point (which need not be on the body itself). The amount of stretch is given as a distance along the vector from point one to point two.

This example doubles the length of the twisted section of the slotted cylinder, created above.

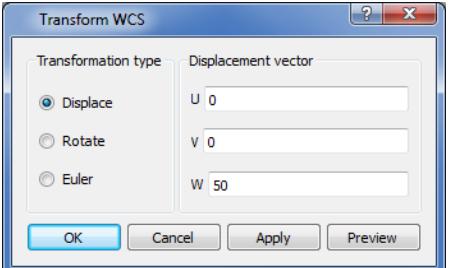
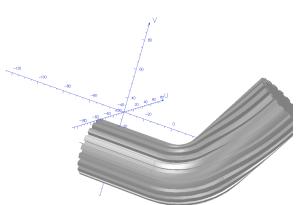
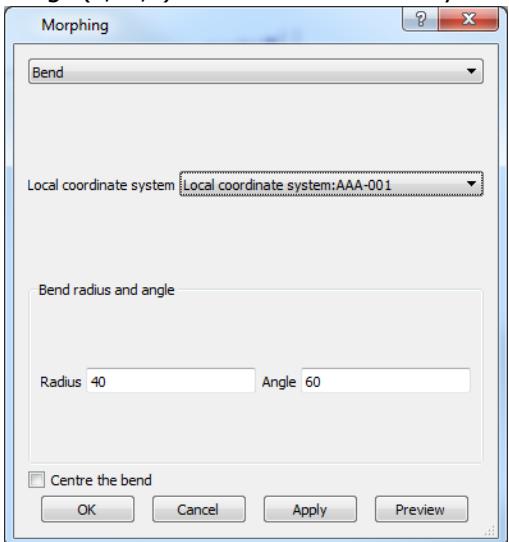
 <b>Pick Bodies</b>	<p>Move the mouse over the twisted body defined above.</p>	
<p>Use the right mouse button to view context menu and select:</p>		<b>Morph.</b>
	<p>Select the Stretch option. Fix the start of the twisted section (displacement of 0) and stretch the end of the cylinder by 50 units.</p>  <p>Select <b>OK</b>.</p>	

## BEND Command

The **BEND** command (**Morph > Bend**) bends the selected body around a line parallel to the U-axis of a specified coordinate system at local V=radius. (For more information, see [Local Coordinate Systems \[page 143\]](#).) With an uncentered bend, the part of the body in positive local z is moved through the specified angle; if the bend is **Centred**, both ends of the body are moved.

The following example bends the twisted part of the slotted cylinder.

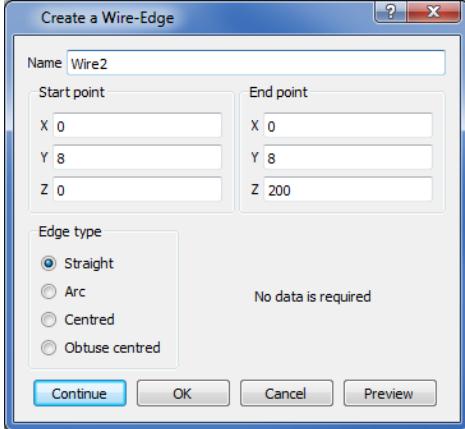
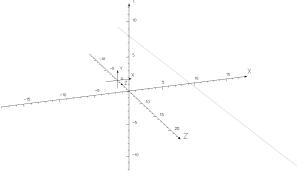
First a local coordinate system is created with its origin at global z=50.

 <b>Coordinate Systems &gt; Transform WCS</b>	<p>Create a new coordinate system, displaced from the global system by 50 units in W.</p> 	
 <b>Pick Bodies</b>	<p>Move the mouse over the twisted body defined above.</p>	
	<p>Use the right mouse button to view context menu and select:</p>  <b>Morph.</b>	
	<p>Select the Bend option. Bend by 60 degrees around a line parallel to the U axis which passes through (0,40,0) in the local coordinate system.</p> 	

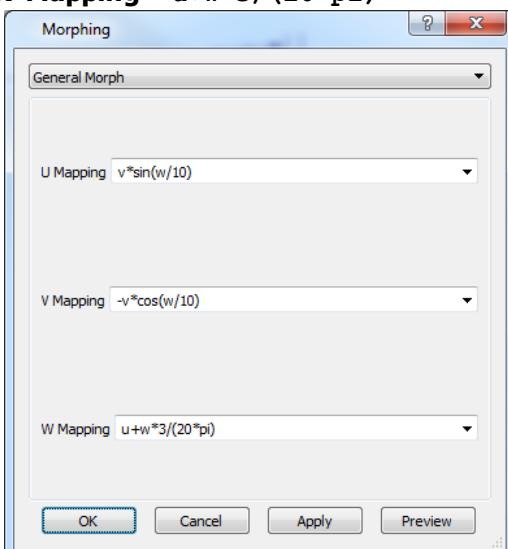
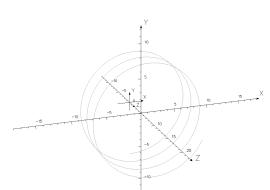
## MORPH Command

The **MORPH** command (**Operations > Morphing > General Morph**) transforms the coordinates of the selected bodies using expressions for the new coordinates in terms of the old. An example of the **MORPH** command is given below, showing the creation of spiral wire.

Reset the Modeller to its initial state using  **Clear All**.

 <b>Create Wire-Edge</b>	<p>Create a wire, length 200, offset in the y-direction to y=8 and close the dialog with OK.</p>  <p>The dialog box shows the following settings:  <b>Name:</b> Wire2  <b>Start point:</b> X: 0, Y: 8, Z: 0  <b>End point:</b> X: 0, Y: 8, Z: 200  <b>Edge type:</b> Straight (radio button selected)  <input type="radio"/> Arc  <input type="radio"/> Centred  <input type="radio"/> Obtuse centred  <b>Buttons:</b> Continue, OK, Cancel, Preview</p>	
 <b>3d Display</b>	<p>The wire created is aligned with the Z-axis, which makes it hard to view. Change the centre of picture coordinates to (0;0;5) and the rotation around axes to (20;20;0).</p>	
 <b>Pick Entity, Pick Bodies</b>	<p>High-light the wire, and then bend it into a spiral using the following command:</p>	

Use the right mouse button to view context menu and select:  **Morph**

	<p>Select the General Morph option. Set mappings: <b>U Mapping</b> = <math>v * \sin(w/10)</math> <b>V Mapping</b> = <math>-v * \cos(w/10)</math> <b>W Mapping</b> = <math>u + w^3 / (20 * \pi)</math></p> <p></p>	
--	---	---

The general morph operation shown above can also be applied to bodies with a volume, for example to a long cylinder. If a wrong morph operation has been applied, the resulting body might look OK on the screen, but it could be turned "inside-out", that means it is defined from infinity to the faces we see on the screen.

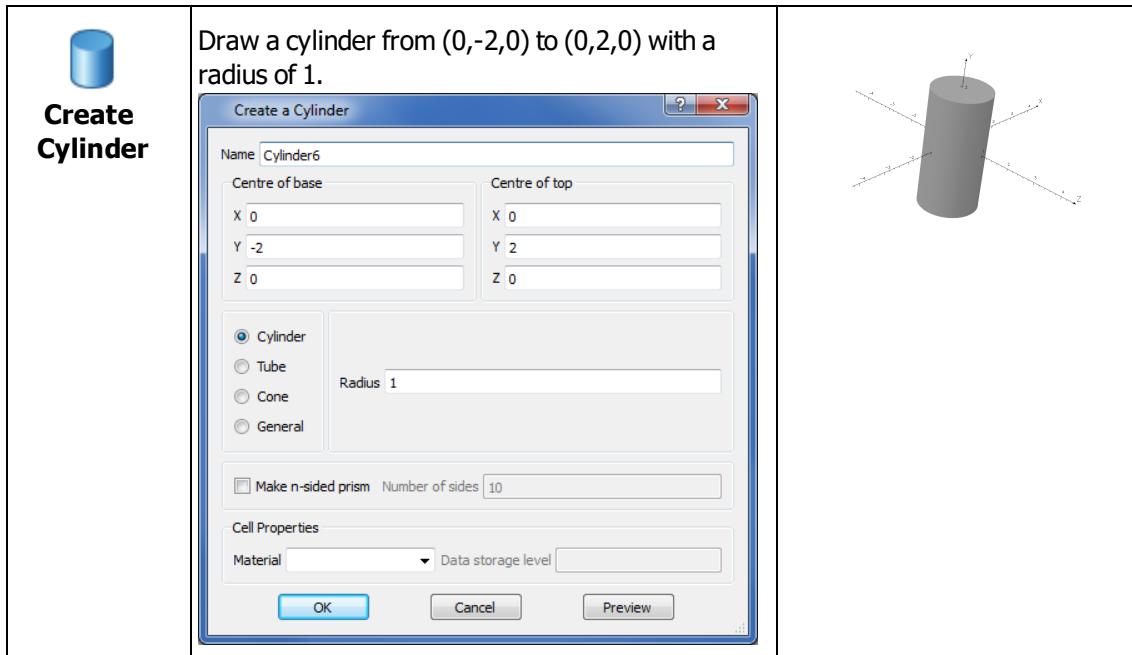
## Local Coordinate Systems

Local coordinate systems can be used to ease the definition of some models. For instance, should the definition of the same object be needed repeatedly, but at different positions, the local coordinate system could be changed in each instance, instead of the parameters of the object.

This is illustrated by the definition of several identical cylinders in different local coordinate systems.

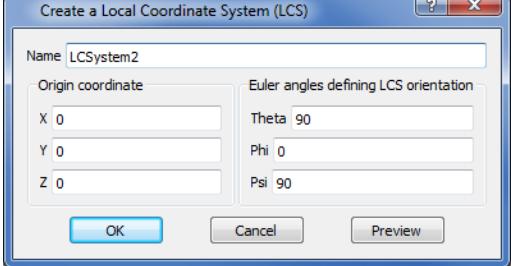
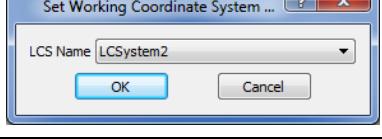
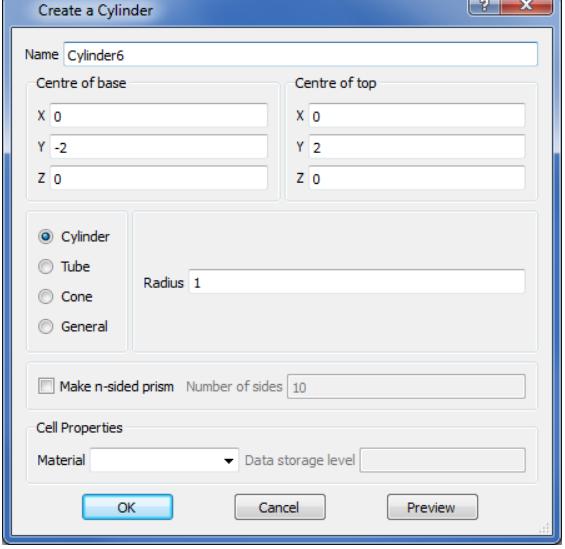
### Global Coordinate System

The first cylinder can be defined on the global coordinate system as follows:



### Rotated Coordinate System

A dimensionally identical cylinder but with its axis parallel to the Z-axis, can be defined by first creating a local coordinate system and subsequently defining the same cylinder on this new local system.

 <b>Create a Local Coordinate System (LCS)</b>	<p>Set origin to (0,0,0), and Euler angles<sup>a</sup> to (90,0,90)</p> 	
 <b>Coordinate Systems &gt; Set to Named LCS</b>	<p>The previously defined system is available to select:</p> 	
 <b>Create Cylinder</b>	<p>Repeat the create cylinder with previous dimensions.</p> 	

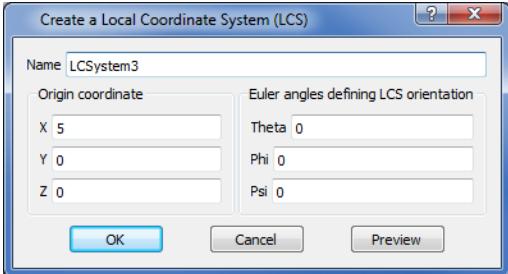
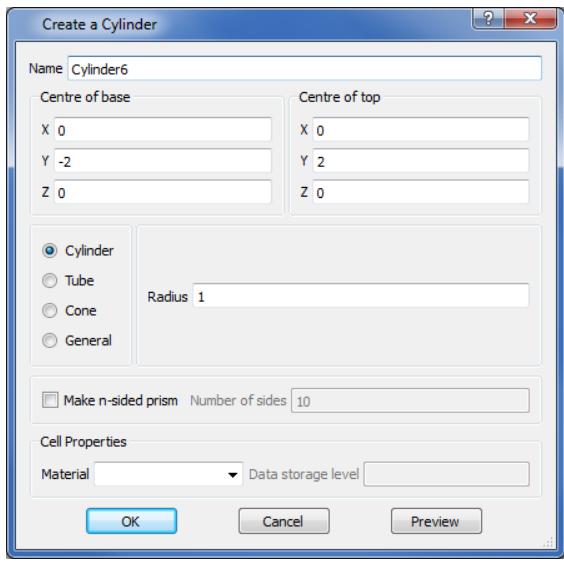
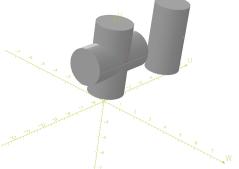
## Displaced Coordinate System

A cylinder can also be defined to be displaced in any direction. The following settings define a cylinder displaced in the positive U direction. Note the this is the U direction of the working coordinate

---

<sup>a</sup>The definition of Euler angles is explained in the ***Opera-3d Reference Manual***.

system, which corresponds to the Y direction of the global coordinate system.

<b>Local Coordinate System (LCS)</b>	Set origin to (5,0,0) and Euler angles to (0,0,0) 	
 <b>Coordinate Systems &gt; Set to Named LCS</b> This time, choose <b>LCSYSTEM3</b>		
 <b>Create Cylinder</b>		

To reset the Global Coordinate System:

Use the right mouse button to view context menu and select: **Use Global Coordinate System**.

## Moving the WCS

Existing entities can also be used to set a new working coordinate system (WCS). This technique has already been used other examples:

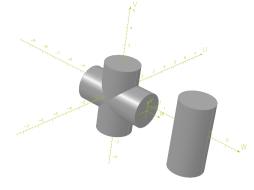
- [Sketching Primitives \[page 79\]](#)
- [Sweep a Face along a Path \[page 120\]](#)

The position of the origin and the orientation of the new axes depends on the entities picked and the order in which they are picked.

Picked Entities	New Origin	New Orientation
Vertex	At the vertex.	Unchanged
Edge	At the centre of the edge.	U parallel to the edge.
Face	At the centre of the face.	W normal to the face.
Face followed by edge	At centre of the edge.	U parallel to the edge. W normal to the face.
Edge followed by face	At the centre of the face.	U parallel to the edge. W normal to the face.
Face, edge and vertex in any order	At the vertex.	U parallel to the edge. W normal to the face.

N.B. Only planar faces can be used.

The working system can be moved to coincide with the last cylinder drawn above.

 <b>Pick Faces</b>	Move the mouse over the top face of the first cylinder (its outline will be highlighted).	
	Use the right mouse button to view context menu and select: <b>Move WCS</b> . Note that the axes colour, labels and values change.	

## Mesh Control

Mesh control is an important feature of the Modeller, as it allows the user to refine a Finite Element Mesh in regions where fields are rapidly changing or where greater accuracy is required.

The model of a simple C-cored electromagnet will be used to demonstrate some mesh control features. [Figure 3.2](#) illustrates the model, which consists of 5 cells, namely the two **iron** cells, the **airgap** cell (air between the pole pieces), the **airin** cell (air space inside the C-core) and finally the **background** (surrounding) air (not shown).

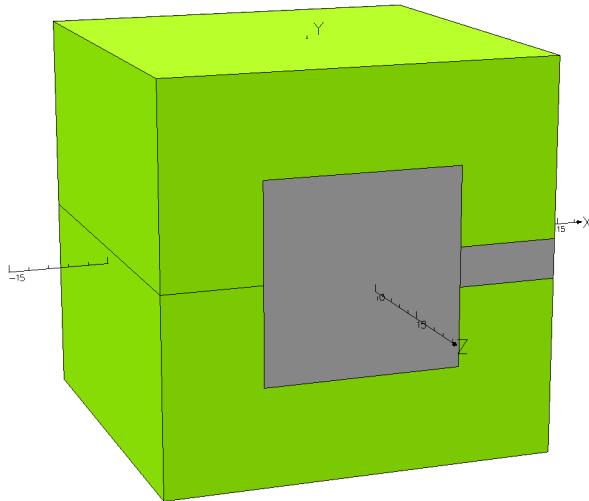
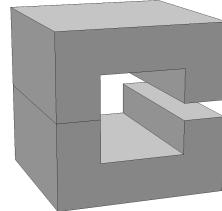


Figure 3.2 A C-core electromagnet (surrounding air omitted)

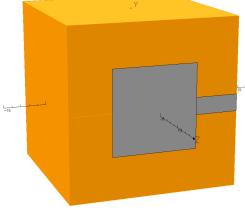
### Building the model

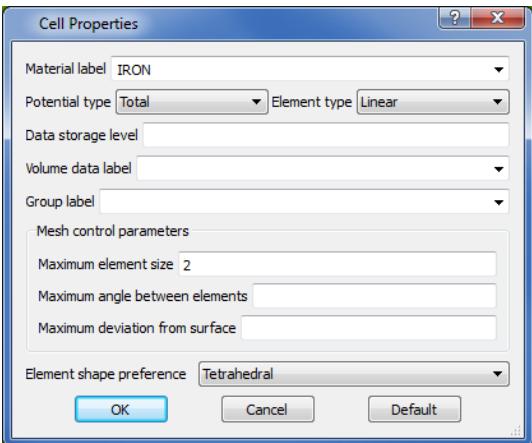
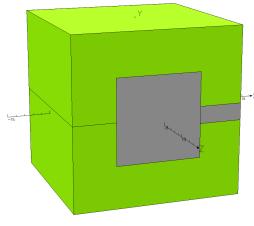
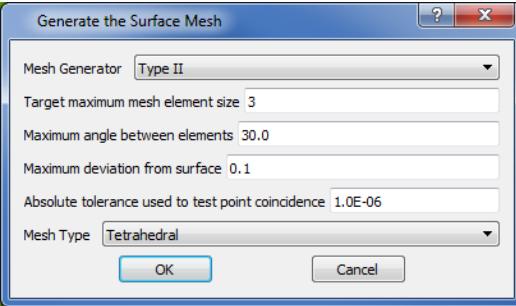
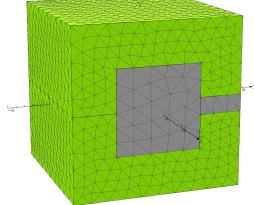
The iron part of the model was defined in an earlier section ([2D Sketching with Wire-Edges \[page 89\]](#)). It can be loaded into the Modeller using:

 <b>Open</b> the data file	Selecting the file <b>C_Core.opc</b> .	
--	--	---

Two cells of air are added so that a different mesh size can be set for each part of the air space.

	Create 2 blocks with the following data:			
Name	First Corner	Opposite Corner	Material	
airin	-5,-5,-10	5,5,10	air	
airgap	5,-1,-10	10,1,10	air	

	Pick the two outer cells to set their properties.	
Right mouse to view context menu, select: <b>Cell Properties</b>		

	<p>Complete the parameter box as follows, setting the <b>Material name</b> and <b>Maximum element size</b> to limit the element size in these cells.</p> 	
 <b>Create Model Body</b>   <b>Generate Surface Mesh</b>	<p>Follow this sequence to create the surface mesh. Enter a target maximum mesh size of 3. This applies to all cells which do not have their own maximum element size set. Select a tetrahedral mesh.</p> 	

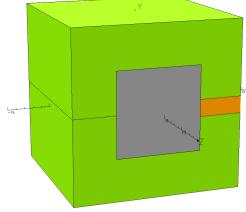
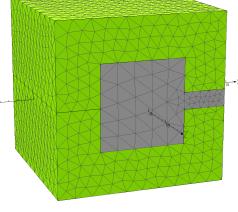
Two other options in the **Generate the Surface Mesh** dialog apply to curved surfaces only and specify the criteria which are used to reduce the element size. The mesh size is reduced,

- if the angle between the normals to two adjacent facets is greater than 30 degrees, or
- if the distance between the centroid of the (planar) element and the real curved surface is greater than a specified value, in the example above 0.1 length units.

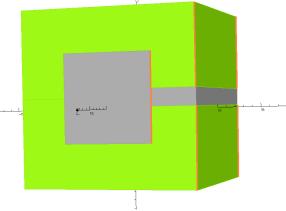
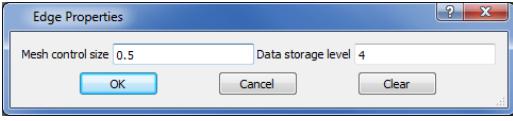
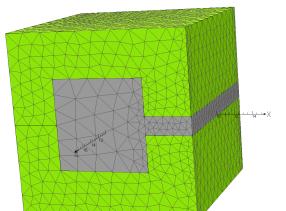
## Modifying the mesh

The mesh needs to be fine in the iron (especially near the air gap) and in the air gap itself, where an accurate field solution is required. In order to refine the mesh in the entire air gap region the user would have to follow the sequence below.

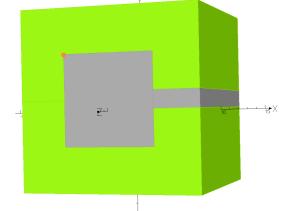
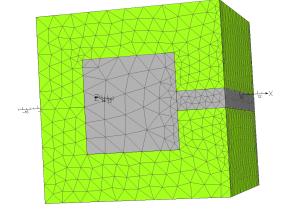
Ensure the **Pick Cells**  toolbutton is enabled. To make changes to the model, it is recommended that the model body is deleted (note this does not delete the component parts of the model). It is possible to make changes to cell, face etc. properties in **Model** mode but the changes are lost if the user returns to **Component** mode. The program warns the user if the changes have been made in **Model** mode.

 <b>Delete Model Body</b>	High-light the cell which makes up the air gap.	
Right mouse to view context menu, select: <b>Cell Properties</b>		
	Set <b>Maximum element size</b> to 1, and select <b>OK</b> .	
	Similarly for the cell in the centre that makes up the central air gap, modify the cell properties: <b>Maximum element size</b> to 2.	
 <b>Create Model Body</b>  <b>Generate Surface Mesh</b>		

To further refine the mesh, some edges will be selected, and a finer mesh defined.

 <b>Delete Model Body</b>		
 <b>Pick Edges</b>	<p>Pick the edges shown in the figure on the right (and the equivalent edges on the face hidden from view).</p>	
Right mouse to view context menu, select: <b>Edge Properties</b>		
	<p>Set <b>Mesh control size</b> to 0 . 5, <b>Data storage level</b> to 4</p>  <p>Creating the mesh as before leads to the mesh shown on the right.</p>	

Mesh control can also be used on faces and vertices. The former is useful if the mesh over the whole face needs to be refined. To refine around a specific vertex, the following sequence can be used:

Right mouse to view context menu, select: <b>Vertex Properties</b>		
 <b>Select Pick Vertex icon</b>	<p>Pick the vertex at (-5, 5, 10) as shown in the figure on the right.</p>	
Right mouse to view context menu, select: <b>Vertex Properties</b>		
	<p>Set mesh size to 0.5, data level 4:</p>  <p>Creating the mesh as before leads to the mesh shown on the right.</p>	

Note that the data storage level helps to avoid ambiguities in the properties of edges, faces etc. when the data is merged to form a model body. Data set at a higher storage level will always be kept.

### Save model

This model is used in some subsequent application notes, so it might be useful to save it for later use.

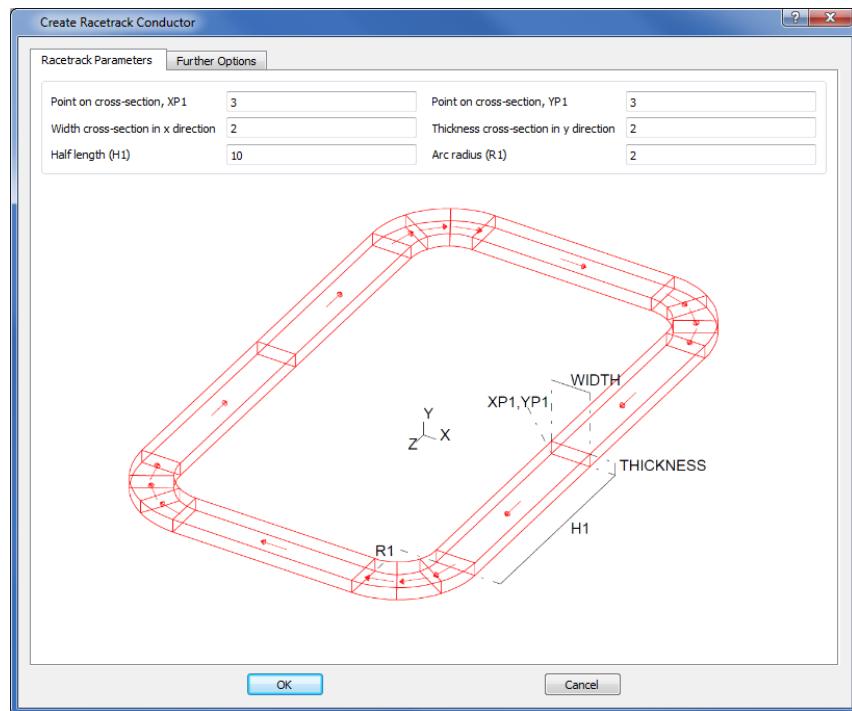
## Defining and Editing Conductors

The Modeller has a range of pre-defined (parameterized) conductor geometries, such as solenoids and racetracks, though simple (generic) shapes are also available and can be used to construct any conductor shape which is not already offered in parametric form.

### Create a new conductor

As an example, the racetrack conductor for the C-cored electromagnet introduced earlier can be

defined by selecting the **Racetrack** toolbutton:  and specifying the settings in the two tabs of the dialog box. This will produce the racetrack conductor of Figure 3.3.



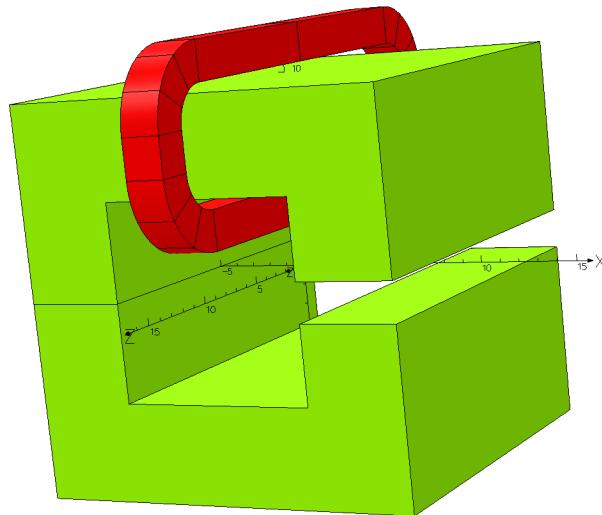
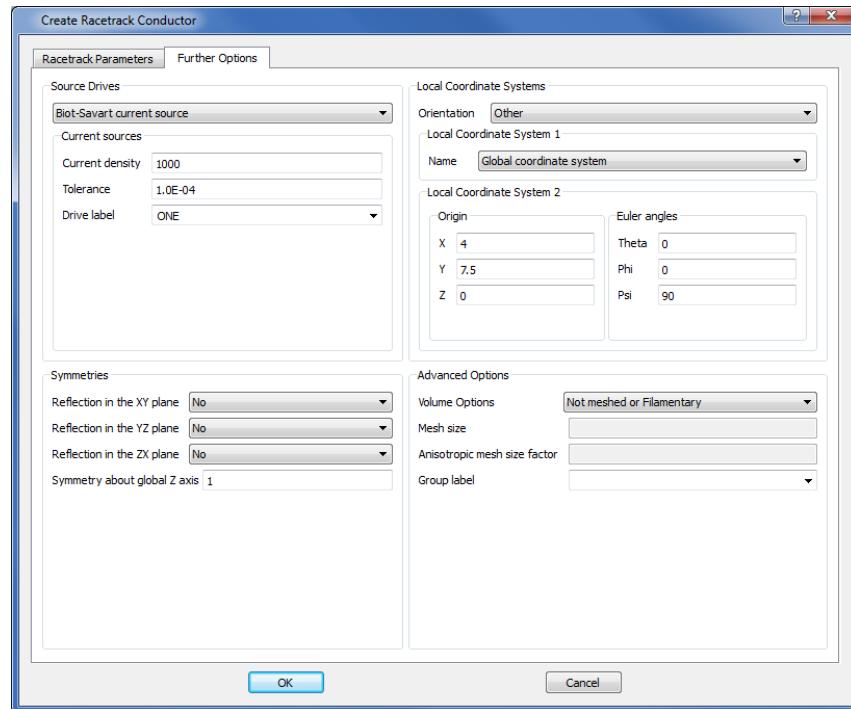


Figure 3.3 The racetrack conductor

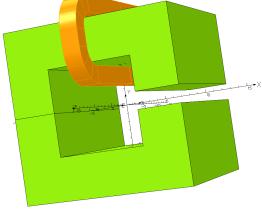
## Save the model

At this point it is recommended to save the C-core and the conductor in a file for use in a later example, using  **Save as New Model Data** and give a suitable file name.

## Modify the conductor

In order to modify an existing conductor it is necessary to carry out the following steps.

Ensure the **Pick Entity**  toolbutton is enabled.

 <b>Pick Conductors</b>	High-light the conductor. In the figure on the right, the air regions were first hidden, but this is not necessary.	
Right mouse to view context menu, select: <b>Modify Conductors &gt; Racetrack</b>		
	Select <b>Modify conductors</b> menu, which gives access to dialogs which are similar to the dialogs used to create the racetrack.	

The **Modify conductors** dialog has an additional Apply button. This button can be useful to see immediately the effects of certain parameters, for example when a conductor has to be fitted to a given geometry.

## Different Types of Conductors

In Opera we can have 4 different types of conductors.

- Biot-Savart conductors are used for pre-defined current sources not connected to circuits.
- Filamentary and the 2 types of meshed conductors are used to model conductors connected to circuits.

The following 4 sections discuss the conductor types in more detail.

### Biot-Savart conductors

Biot-Savart are not part of the Finite Element mesh, and have to be embedded in **REDUCED POTENTIAL**.

They are current driven; they have predefined current density values which can be:

- scaled by different scaling factors in statics analysis;
- the amplitude of the current density in steady-state ac analysis;
- scaled by a function of time in transient analysis.

The value of current can be used to define the current density using the expression **current/AREA**.

If only a part of the Finite Element mesh is built (using symmetry), the Biot-Savart conductors will still have to be defined as a complete set of coils.

They can be used in all magnetostatic and electromagnetic solvers.

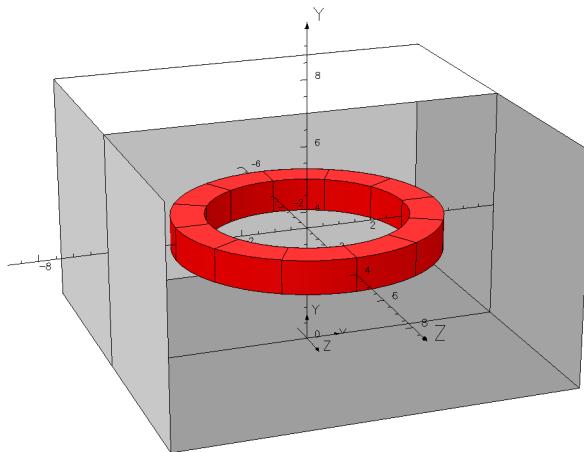


Figure 3.4 A conductor used as Biot-Savart conductor

## Filamentary conductors

Filamentary conductors are part of the mesh, and have to be in **TOTAL POTENTIAL**. They are connected to circuits, and as such can be driven either by a voltage or a current.

They can only be used in edge variable solvers like **Transient / Harmonic Electromagnetic**, **Transient Motion** or **Demagnetization**.

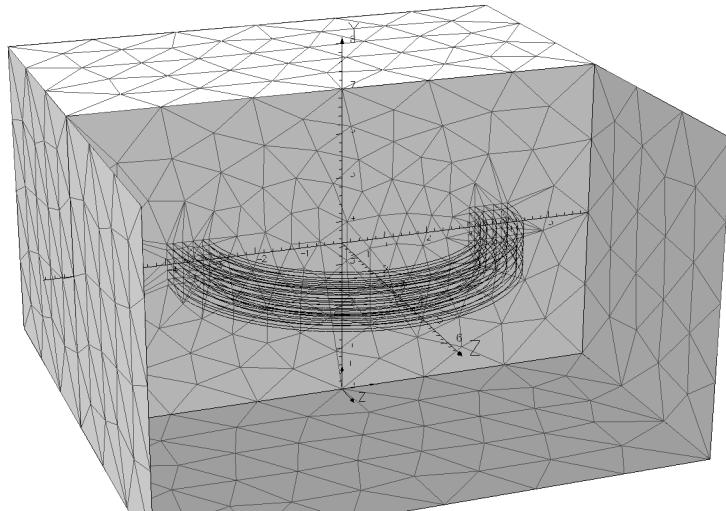
If the geometry of a model shows symmetry and only a part of it has to be meshed, also that part of the conductors has to be defined. The symmetry factor for the conductors is then given in the setup of the circuits; in most cases the option to **Use model symmetry** can be used.

For a filamentary conductor, the parameter **Resistance per unit length (RPUL)** of the wire and the number of **TURNs** must be set in the circuit data.

Filamentary conductors are modelled using a number of representative filaments which are distributed through the cross-section. A single filament will be placed at the centre of the conductor; setting the number of filaments to  $n$  will create a  $n \times n$  array of filaments evenly spaced in the cross-section. The number of filaments will not normally be the same as the number of turns.

Filaments can only be included in tetrahedral elements. The cells surrounding a filamentary conductor must also be meshed with tetrahedra to avoid pyramid elements which could exist if the neighbouring cells had prism or hexahedral elements (see [Meshing in the Modeller \[page 541\]](#)).

[Figure 3.5](#) shows an example with 6 by 6 representative filaments.



*Figure 3.5 Representative filaments are part of the Finite Element mesh. In this picture the conductor shape is hidden so that the filaments become visible.*

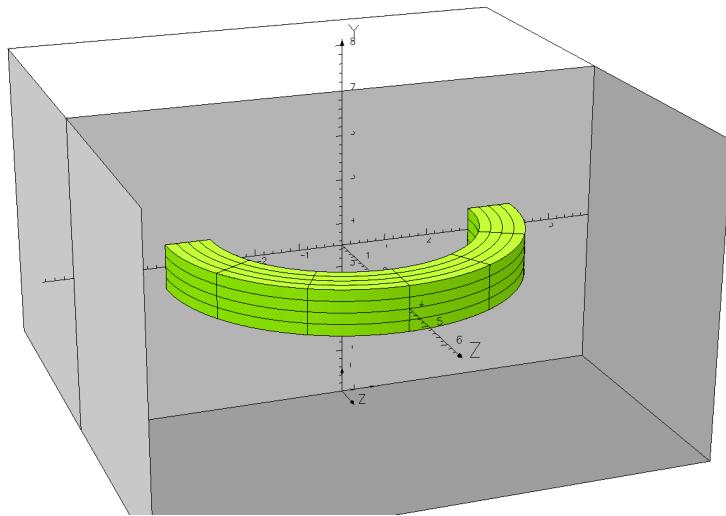
Note that filaments must not lie on boundary surfaces of the finite element mesh. If a conductor is positioned so that it is half inside and half outside the mesh, an even number of filaments should be used so that there are no filaments in the boundary surface.

## Meshed conductors

The mesh inside meshed conductors is formed from cells which are created during the **Create Model Body** stage. This process can be rather expensive in terms of CPU-time for complex conductor shapes (helical and constant perimeter ends and 20-node bricks).

The potential is set to **TOTAL POTENTIAL** automatically.

The meshed conductors are also connected to circuits, and as such can be driven either by a voltage or a current. They can be used in edge variable solvers like **Transient / Harmonic Electromagnetic**, **Transient Motion** or **Demagnetization** as well as **QUENCH**. Figure 3.6 shows the solenoid conductor set to this option.



*Figure 3.6 A meshed conductor after "Create Model Body"*

Regarding model symmetry, the same rules apply as for filamentary conductors: only the parts of the conductors that are inside the finite element mesh have to be defined; and the symmetry factor is given in the setup of the circuits.

A meshed conductor consists of cells which are given a material name which is the same as the circuit element name. The material properties of the cells can be defined (electrical conductivity of wire & area of wire cross section) in the **MATERIALS** command (parameters **SIGWIRE** & **AREAWIRE**) or the conductor can be defined in terms of its circuit parameters in the Circuit Editor.

### Regular meshed conductors

Regular meshed conductors are similar to the meshed conductors described above except they are converted into relatively few larger cells during the stage of creating the Model Body, which are then meshed with hexahedral elements.

As before, the potential is set to **TOTAL POTENTIAL** automatically and the meshed conductors have to be connected to circuits. [Figure 3.7](#) shows the solenoid again; this time as a regular meshed conductor.

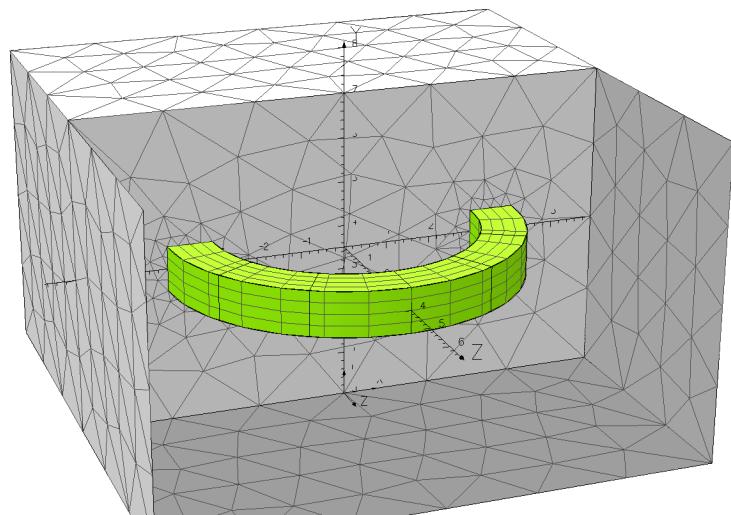


Figure 3.7 A regular meshed conductor. The mesh inside the conductor is visible after "Create Surface Mesh".

## Uses of Volume Data

Volume data is used for setting the following attributes:

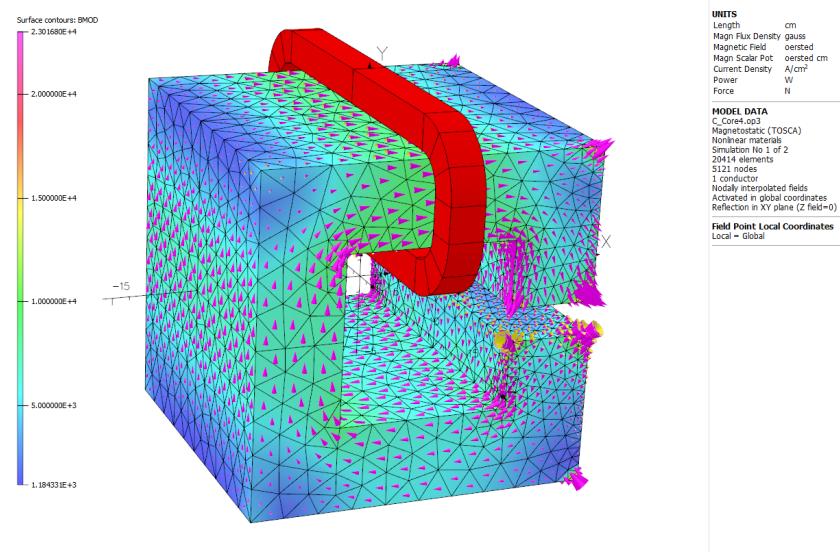
- Permanent magnet and anisotropic material orientation
- Current density assignment (for **Harmonic** and **Transient Electromagnetic** models)
- Velocity (for **Fixed Velocity** models)
- Charge density (for **Electrostatic** models)
- Thermal heat sources

In this section, volume data will be used to assign anisotropic properties to the iron of the C-core magnet that was previously discussed.



In Component Mode, go to Model Symmetry , set the shape of the background to **Block** with region scaling factors of (10,10,10), and a reflection in the XY plane defined as **Tangential magnetic**. Alternatively, the data file (**C\_Core4.opc**) is provided in the examples folder.

**Figure 3.8** illustrates the flux density distribution in the C-core electromagnet. The model was solved with nonlinear magnetic properties for the iron taken from the file **tenten.bh** (supplied with Opera in the sub-folder **%VFDIR%\bh**). The surface mesh element size was set to 15.



**Figure 3.8** Flux density Distribution in the electromagnet  
(values are mesh dependent)

**Figure 3.9** illustrates the flux density distribution along a line in the iron, just inside the racetrack coil (z-directed line from z=-10 to 10 at x=0 and y=7.5). It can be observed that, due to the end winding,

the flux density is higher near the edges of the iron, though the increase in flux density from the centre to the edge is smooth. This is also a natural effect, and is due to the fact that end winding induced field can travel axially into the iron which is not laminated. Note that the values are approximate, and are mesh dependent at this level of discretization.

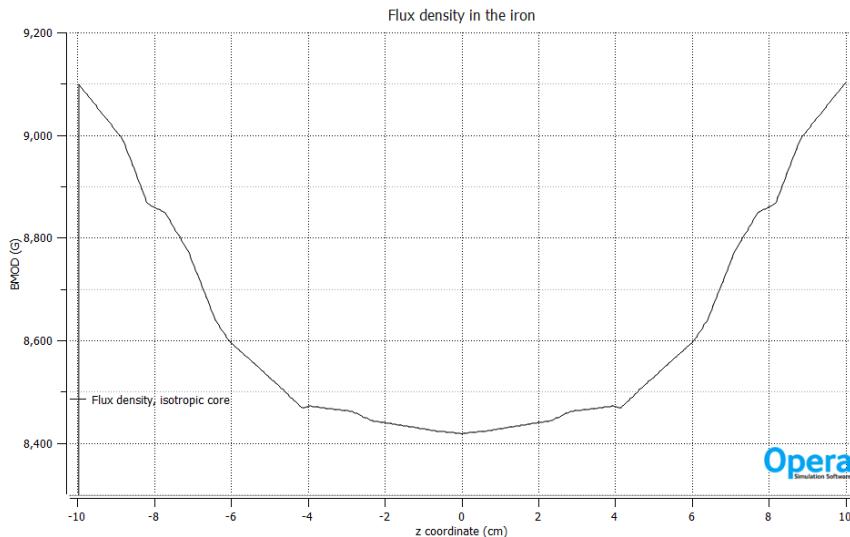
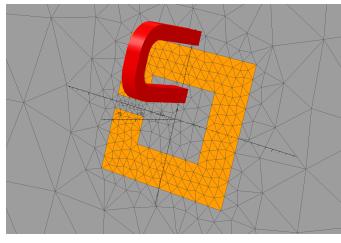
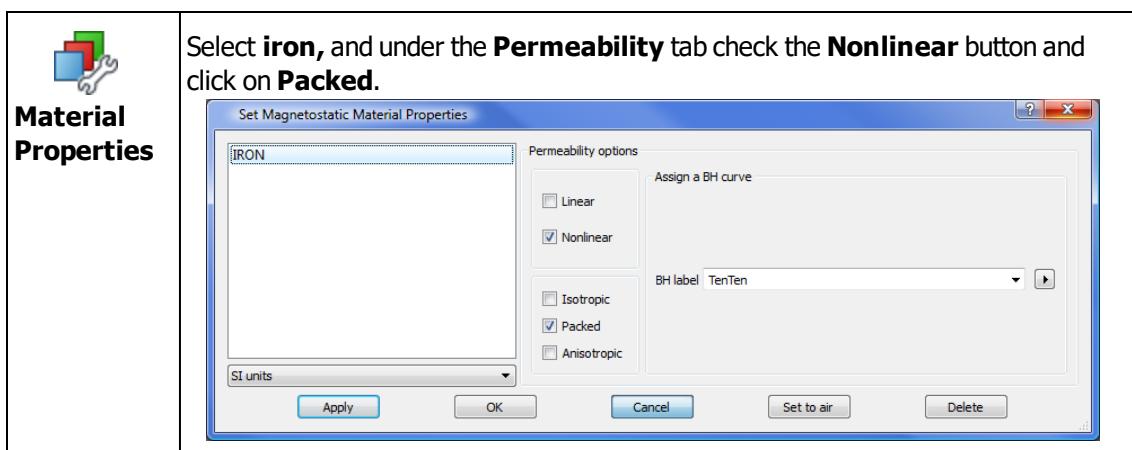
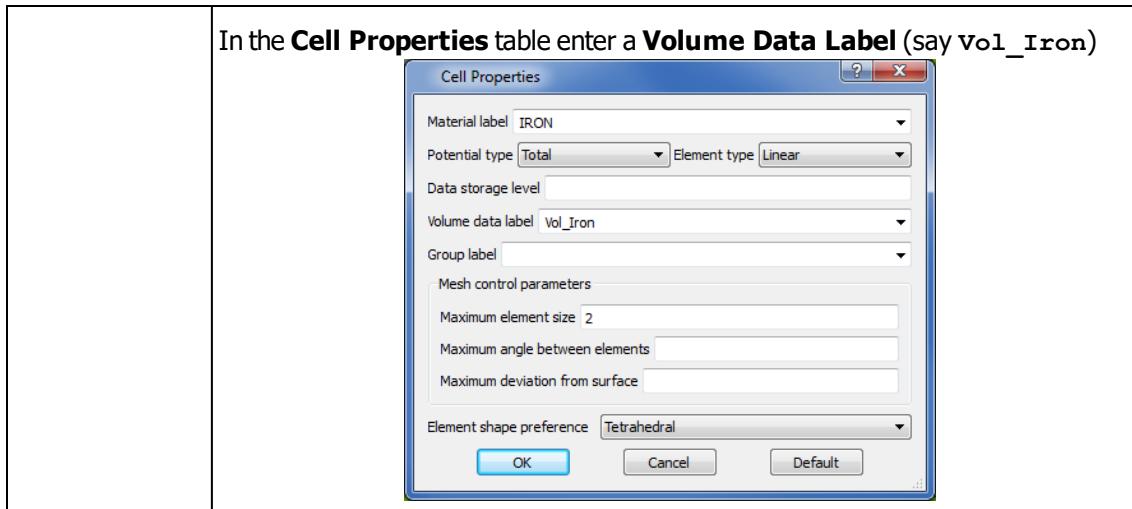


Figure 3.9 Flux density in the iron

## Including laminations

In order to solve the same model but now with laminated material anisotropy, the user should carry out the following steps (note that the steps would be similar for any model).

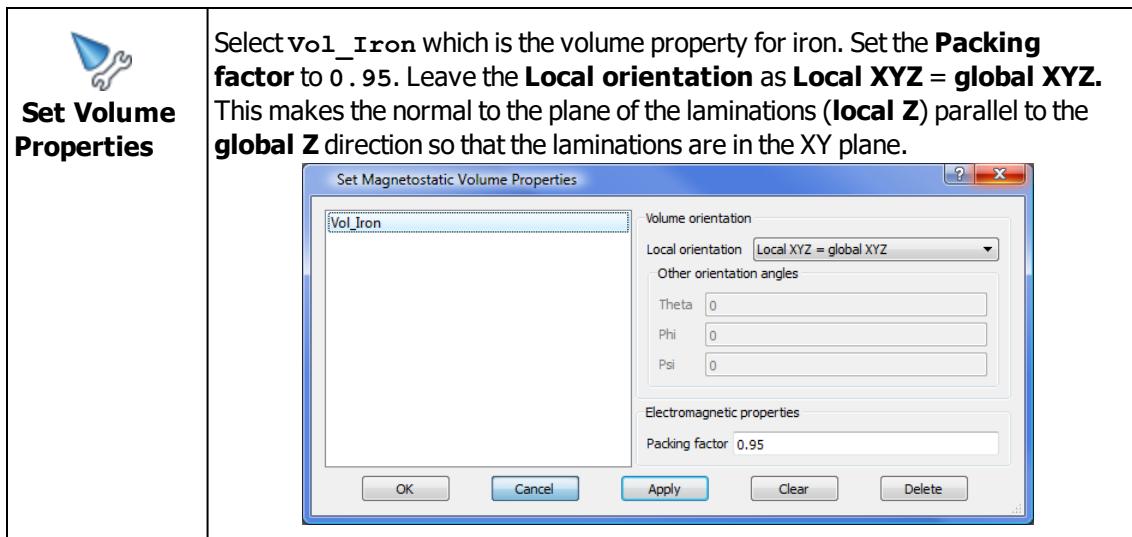
 <b>Pick Entity</b>  <b>Pick Cells</b>	Select the two cells which make up the iron.	
Right mouse to view context menu, select: <b>Cell Properties</b>		



The BH label **TenTen** corresponds to the BH data assigned using

**BH Data** (see [Defining Material Properties \[page 55\]](#)).

To set the volume properties:



The data is now ready for mesh generation, database creation and analysis using the Magnetostatic Solver, followed by post-processing.

**Figure 3.10** shows the line graph of flux density in the anisotropic iron (just inside the coil), created by the Post-Processor. Comparing this with **Figure 3.9**, it is clear that the end effects are now more pronounced at the edges of the iron because the **B**-field can no longer penetrate in the axial direction. In physical terms this implies that the final few laminations of the core are operating at higher flux densities.

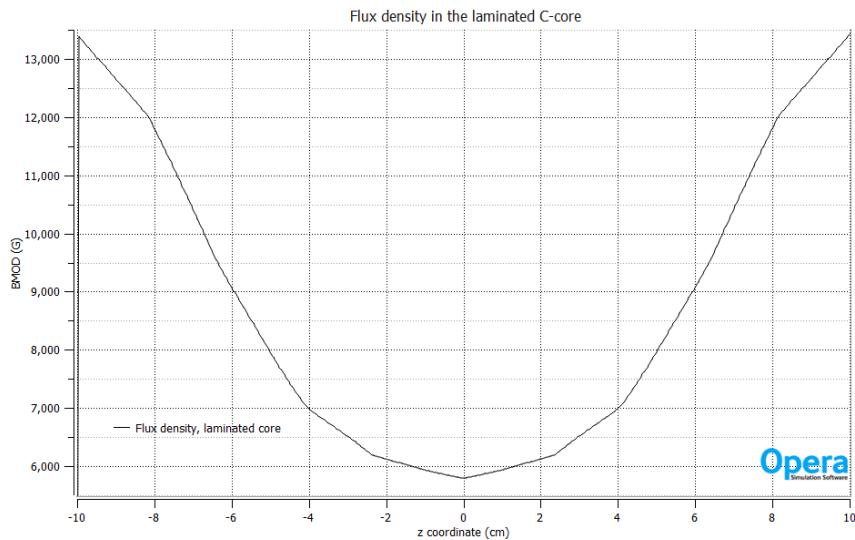


Figure 3.10 Flux density in the laminated C-core

## Analysis Options and Database Creation

Before the Modeller can write out an ***op3*** database file, the following steps must be completed:

- create the model body (see [Creating the Model Body \[page 52\]](#));
- set the analysis-specific data;
- create the surface and volume meshes (see [Mesh Control \[page 147\]](#)).

These steps can be performed one by one, or in one go by providing all necessary data in one large dialog box.

The steps are illustrated in the worked examples. Some more examples of analysis-specific data and database creation are given here. In this section the use of the **Harmonic Electromagnetic** solver for eddy current analysis is shown, but very similar menu items and dialogs are used for other analysis types.

### AC Eddy Current Analysis with Harmonic Electromagnetic solver

The C-core electromagnet, shown in [Figure 3.2, on page 147](#), can be driven at multiple frequencies. In order to create a single database, containing solutions for multiple frequencies, load the data file (***C\_Core4.opc***) provided in the examples folder. Then change the analysis type to **Harmonic Electromagnetic**.

The configuration pane of the analysis also allows a uniform external field to be imposed. A drive

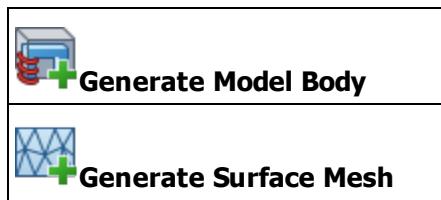


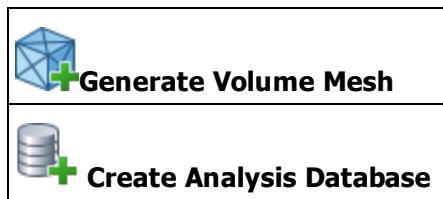
label can be given so that the phase angle of the AC external field can be specified under **Drives**.

### Database Creation

An Opera-3d database (***op3***) file contains the complete specification of a model for analysis. It is created by the Modeller; the analysis programs add solution vectors; and the Post-Processor reads it to display the results.

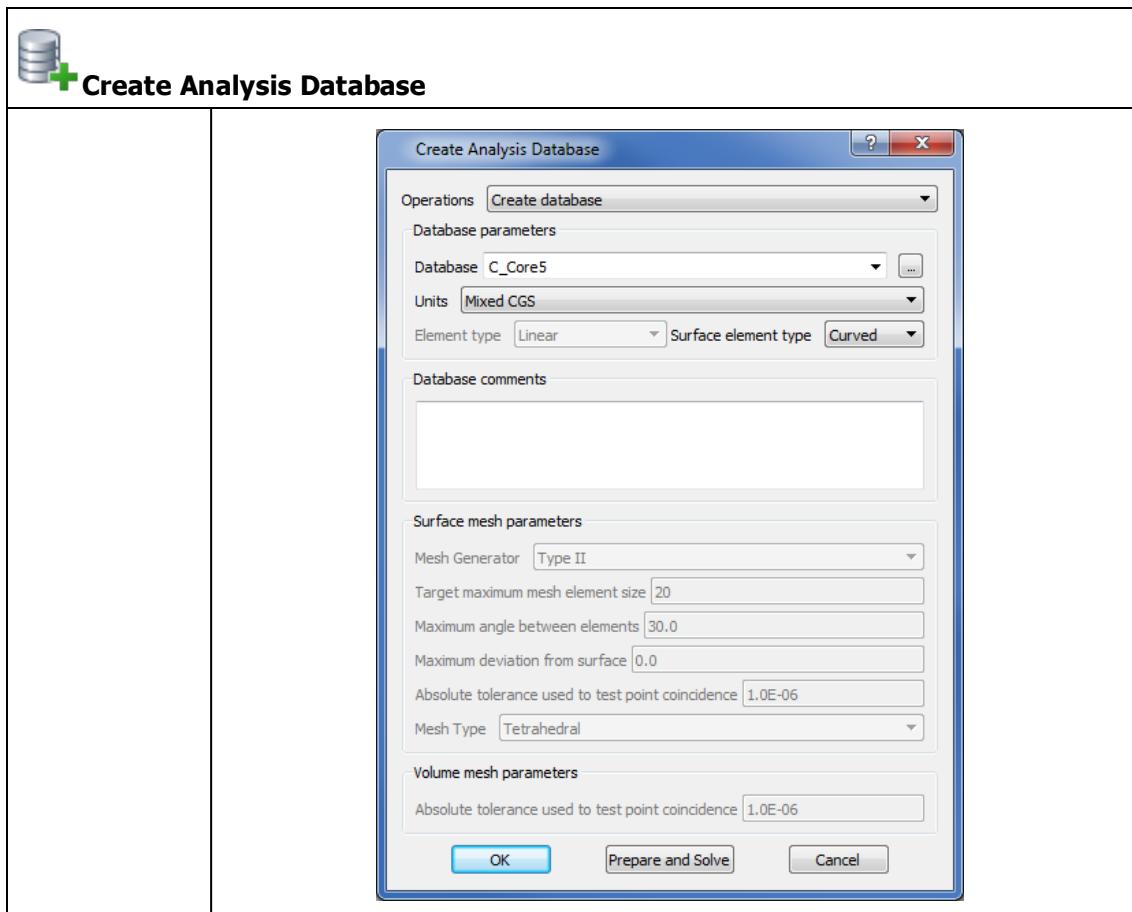
When building a new model from scratch, it is recommended to perform the steps explained above manually, so it would be:





Depending on the complexity of a model, each of these steps can take a while to complete. Therefore it is essential that suitable parameters are used for each step.

When it has been established that all these operations will complete without error, the 4 steps can be performed in one operation, as in the dialog below. It is recommended for use after minor modifications to a model.



The following data can be supplied:

- **Operations.** Depending on the actual status (surface mesh or volume mesh present), this menu can offer different options:  
**Generate surface and volume meshes and create database**  
**Generate volume mesh and create database**

**Create database**, (if both surface mesh and volume mesh are present).

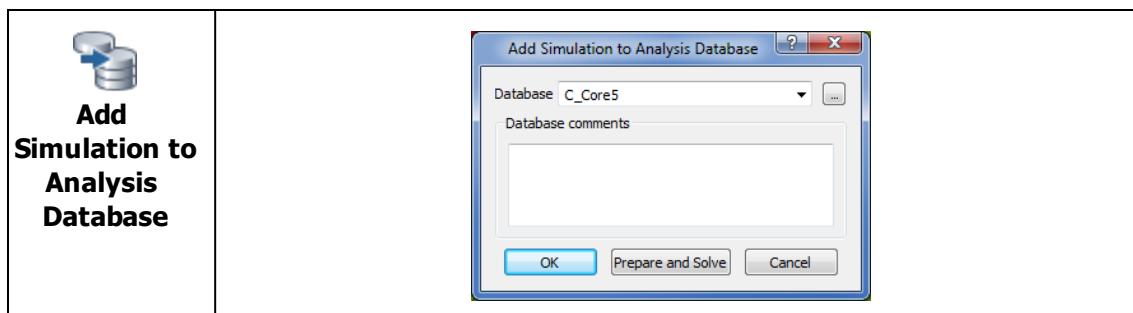
According to the current choice, parts of the dialog, which are not relevant, will be greyed out.

- **Units.** Up to this point, all the geometric data has been expressed as numbers. Here the units are defined by unit sets: **Mixed GCS**, **True CGS**, **SI**, **SI with mm**, **SI with microns** and **SI with inches**. The details of each set can be found using the **What's this** help or typing **F1** while the dialog is visible.
- **Element and Surface Element Type.** For edge element simulations, the **Element Type** can only be linear. Therefore this option is greyed out for a **Transient** or **Harmonic Electromagnetic** simulation. For more information, please see the online help (on Windows systems via a right mouse-click on the relevant option).  
For nodal solvers like **Magnetostatic** or **Charged Particle**, by default linear and quadratic elements are used according to the cell and face data, with quadratic elements touching curved surfaces.  
The options in this dialog can be used to override the element types on the geometric entities with one type of element.
- **Simulation Title.** A multi-line title of the simulation is recorded in the database so that it can be displayed later in the Post-Processor.

## Add to existing Database

Once a database has been created, either including 1 "case", or as in the example above, 3 "cases" with the frequencies 50, 100 and 200 Hz, it can be submitted to be solved.

However it is also possible to add a 4th and further cases to the OP3 database, before or after it has been solved, using the following menu route:



This option is only available after a first database has been written out. Note that by using this option, the user has to make sure that the volume mesh has not changed. This can be achieved by either saving the volume mesh to an **opcdb** file, or by leaving the Modeller window with the volume mesh open.

## Transient Eddy Current Analysis with Transient Electromagnetic solver

The same model can also be used to examine the transient behaviour of the electromagnet. Opera-3d can be used to estimate how the flux builds up in the C-core when a transient is applied to the exciting coil.

### Output times

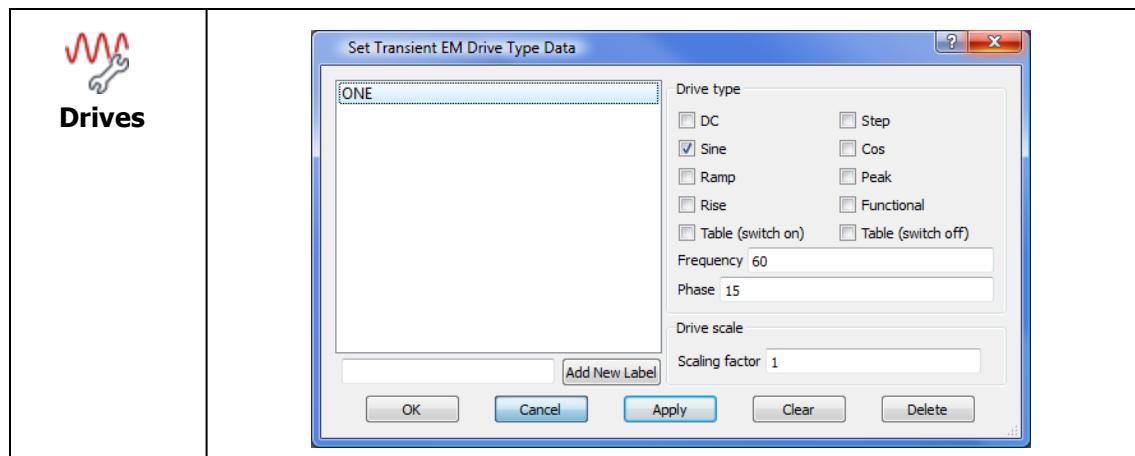
In order to set up the transient solution replace the **Harmonic Electromagnetic** analysis with a **Transient Electromagnetic** one in the **Analysis Settings** window. In the configuration pane of the analysis, add the additional **Output points** **5E-04** and **1E-03**.

### Transient drive

The drive information is used by the Opera-3d Transient solvers (**Transient Electromagnetic**, **Transient Motion** and **Demagnetization**) to define the type of drive to be applied to each drive label. Among others, the following functions are available:

- Step
- Ramp
- Sine
- Cosine
- User-defined Table

The last option can be used to define the shape of the function to be applied to the exciting coil, if none of the pre-defined functions available are appropriate. Drive information can be defined via



Drive labels, each corresponding to one (or a set of) conductor(s), a uniform external field or a boundary condition, must be individually selected and assigned the appropriate properties. Note that

different entries are relevant to the different drives which are available. In this instance, a sinusoidal source has been selected and therefore the **Frequency** and **Phase** entries must be filled in.

## Parameterizing Models and Rebuilding

An existing model can be parameterized allowing different versions of the model to be created quickly and easily. To show the features working, the starting point will be the C-core example created in [2D Sketching with Wire-Edges \[page 89\]](#). The model is shown in [Figure 3.11](#).

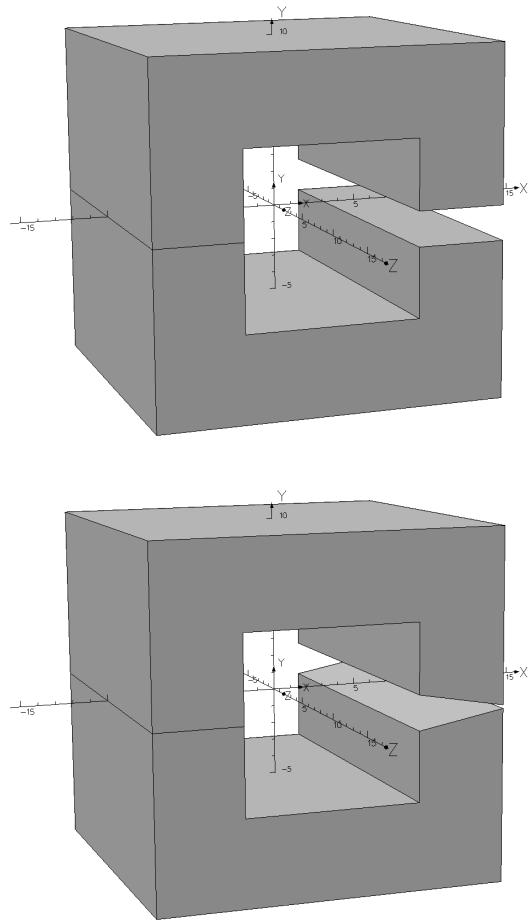
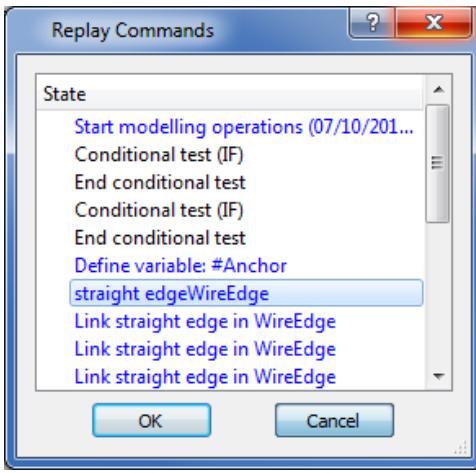
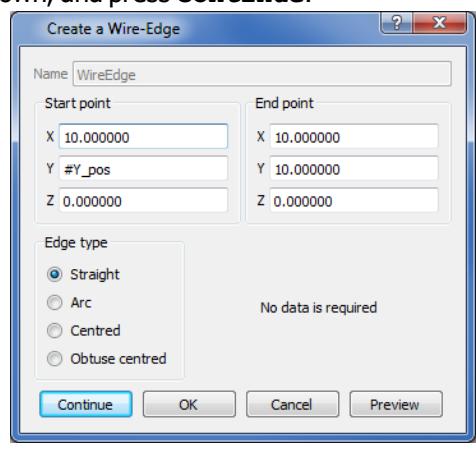
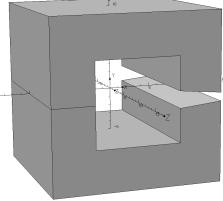


Figure 3.11 The C-Core example before and after changing a coordinate

### Replay commands

As a model is created, a history of the commands, which were used to build the model, is stored. It is possible to replay these commands from any point in the history stream. For example, we can go back to the command that created the first point of the wire-edge, and make a change to its coordinate, then replay all the subsequent commands to create a new version of the model.

 <b>File Open</b>	Open the file <b>C_Core.opc</b> which was created in one of the previous sections.
<b> Replay Commands</b>	<p>High-light the <b>straight edge</b> entry as shown, and select <b>OK</b>.</p> 
	<p>The <b>Create a wire edge</b> dialog appears. Change the y-coordinate of the <b>Start Point</b> from 1 to a new user variable called <b>#y_pos</b> as shown, and press <b>Continue</b>.</p> 

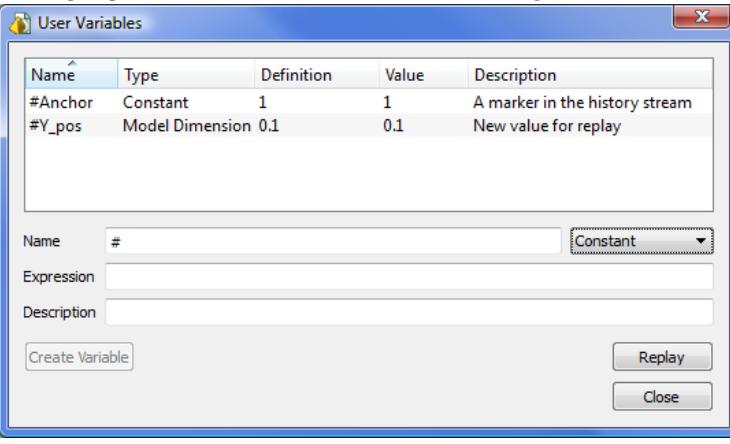
	In the <b>Define Unknown Variable</b> dialog, enter a value of $0.5$ for the Y-position. Optionally give a description. Press <b>OK</b> to close the dialog.
	After a few seconds the geometry has been rebuilt with the new dimensions.  

## Model dimensions

The user defined variable called `#y_pos`, which has been defined above, will now be modified to show how new models can be created simply by changing the value of a parameter. Any number of such parameters can be defined, but for simplicity only one is used here.

Note that the user variable named `#y_pos` is a variable of type **Model dimension** (other types of variable are **Constants** and **Parameters**, and are described in the *Reference Manual*).

To make changes to the model, it is now only necessary to modify the value of the variable `#y_pos`, as follows.

 <b>User Variables</b>	<p>Under <b>Model Dimension</b>, double-click on the Definition for the variable <code>#y_pos</code>. Change the expression to <math>0.1</math> and press return. The column labelled <b>Value</b> will now show the new value of <math>0.1</math> as well. Select the <b>Replay</b> button to re-run the commands using the new value.</p>  <p>Select <b>Close</b> to close this dialog.</p>
--	---

The new model is displayed, with the Y-position of an edge of the air gap now at the new position, see [Figure 3.11](#), bottom.

## Parts Libraries and CAD Files

---

### CAD Files

The geometry of a model or parts of a model can be saved in several different types of file:

- Opera Modeller (\*.opc) files are based on the ACIS **sat** format but also contain information about conductors, meshing, analysis and command history, in fact, all the information necessary to recreate the data at the time the file was written.
- Opera Modeller binary (\*.opcb) files contain the same information as **opc** files but also contain the finite element mesh.
- CATIA, IGES, Pro-E, SAT and STEP files provide an interface between the Modeller and other CAD systems. They contain the geometric model, not including the conductors. (N.B. Some of these formats require additional licensing.)

### Reading CAD files

There are 3 ways of reading geometric data from file into the Modeller.

- Open an **opc** or **opcb** file:  , this replaces any data in the Modeller with the data in the **opc** file, including conductors, material properties, analysis data and command history. If the file is type **opcb**, the finite element mesh will also be loaded.
- Open a CAD file:  , then use the drop-down file filter to choose the type of CAD file to be loaded (CATIA, IGES, Pro-E, SAT or STEP) from the list.<sup>1</sup> Once selected, the available files will be displayed. This replaces any geometry data in the Modeller with the geometry in the CAD file, and resets the Analysis Type and all analysis settings to default values.
- Insert parts from a CAD file: **Insert Other** . This adds to the current model data from Opera, CATIA, IGES, Pro-E, SAT or STEP files.
- Insert parts from an **op2** file: **Insert Opera-2d**. This adds sheet faces from the non-air regions in an Opera-2d model. The sheet faces can be extruded to form volumes.

### Writing CAD files

There are 4 ways of saving Modeller data to a file.

- Save an **opc** file: **Save as New Model Data**. This writes the geometry, conductors, material properties, analysis data and command history to file.

---

<sup>1</sup>Sometimes, parts imported or inserted from CAD files have very small features which can cause problems in the mesh generator. These can often be repaired by adjusting tolerances (**Tolerances**) or by picking bodies and using **Operations -> Check**. For more information see the **CHECK** and **PRECISIONDATA** commands in the **Opera-3d Reference Manual**.

- Save an ***opc*** file: **Save Model with Mesh**. This writes the geometry, conductors, material properties, analysis data, command history and finite element mesh to file.
- Save to current file: **Save**,  . This overwrites the current ***opc*** or ***opc*** file with the latest data.
- Export parts to a CAD file: **Export Picked**. This creates a CATIA, IGES, SAT or STEP file with the bodies that have been picked.

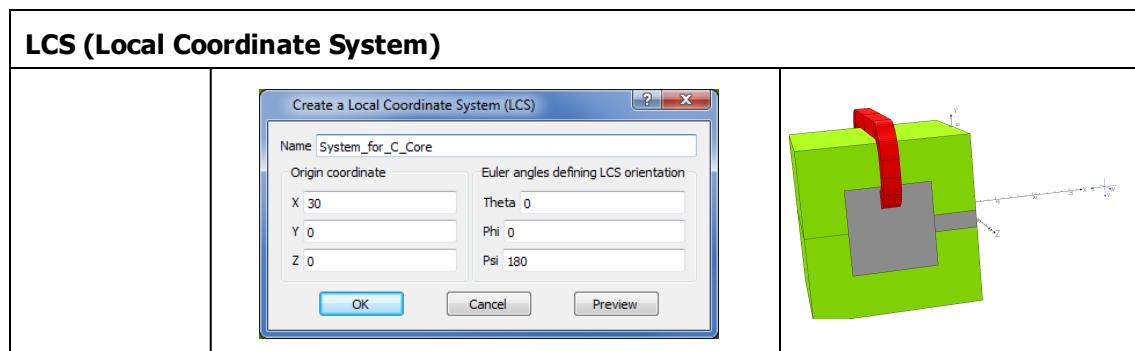
## Building a Model from Parts

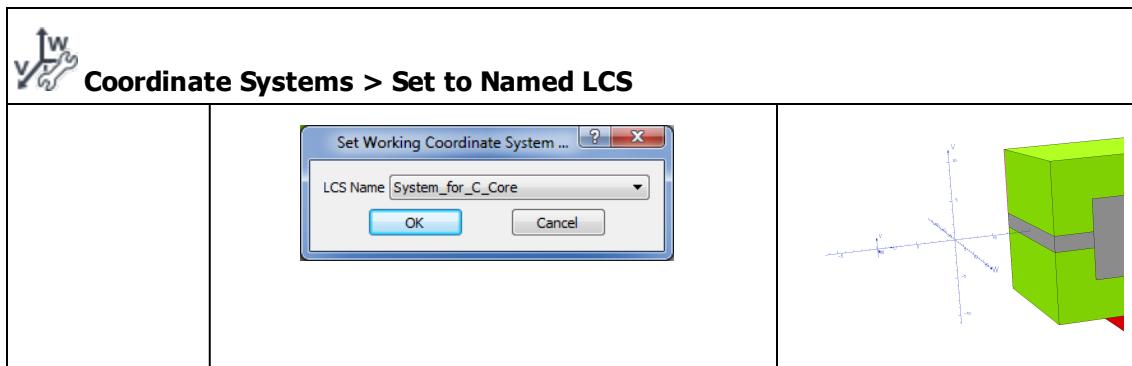
The Modeller allows users to build a model from several parts, which can be created and saved separately. This example uses the C-core magnet, which has been created in the section about **Mesh Control [page 147]** and saved in the section about **Defining and Editing Conductors [page 153]**. To illustrate how a model can be created from parts, this model will be loaded in twice.

Any type of file can be used to build up a model from parts.

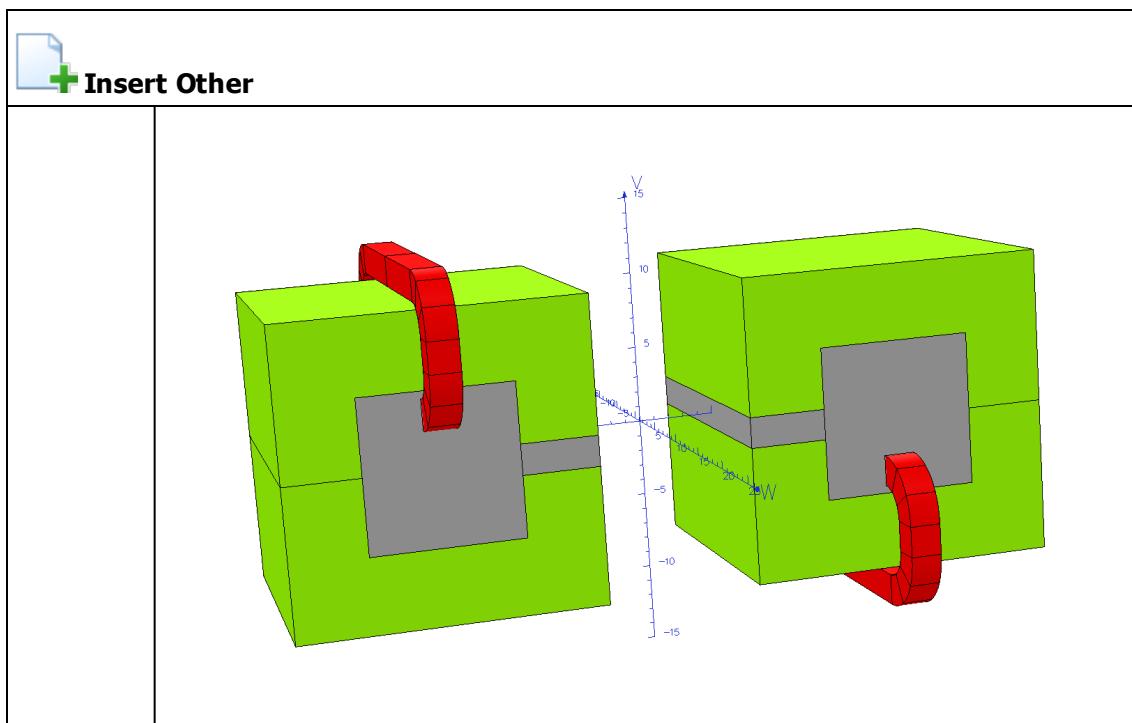
Before starting this example, the Modeller should be reset to its initial state using  **Clear All**. Load the C-Core example from one of the previous sections into the Modeller, using the **File Open** icon:  and selecting the file from the appropriate directory. Alternatively, a data file (**C\_Core4.opc**) is provided in the examples folder. Use  **Delete Model Body**.

Following this, the same C-core will be loaded a second time into the Modeller. A new local coordinate system will be used to displace the second copy and to rotate it by 180 degrees at the same time.





The C-core is again loaded into the Modeller, using:



Based on this procedure, very complex models can be built up using parts from a user defined library.

# **Chapter 4**

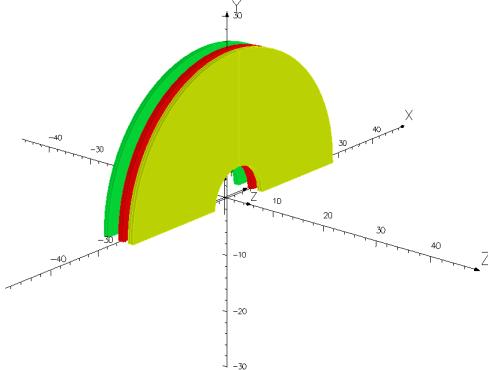
## **Plate Capacitor Electrostatic Example**

This example uses the **Electrostatic** solver to analyse a plate capacitor. The model is built in the Modeller and then analysed before the results are examined in the Post-Processor.

The first section involves defining the model of the plate capacitor and the surrounding air. Voltages are applied to the plates to simulate a charged capacitor. The database is created and solved with the **Electrostatic** solver.

In the second section the solved database is loaded into the Post-Processor, the electric fields are displayed and the capacitance of the complete system is calculated.

Figure 4.1 shows the plate model with different colours representing the different voltage boundary conditions that have been defined on the surface of the plates.



*Figure 4.1 Plate Capacitor*

## Building and Running the Model

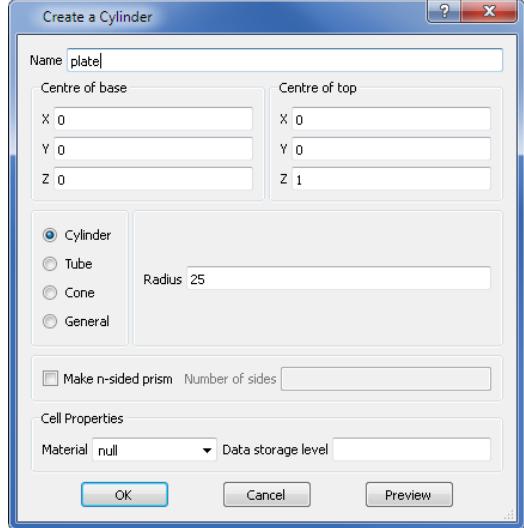
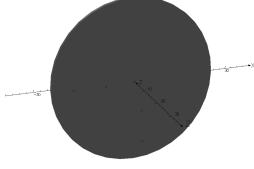
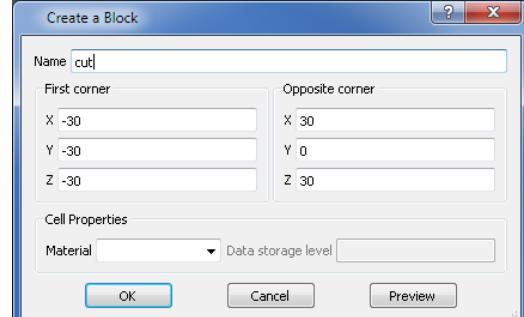
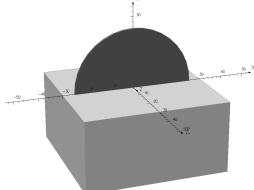
---

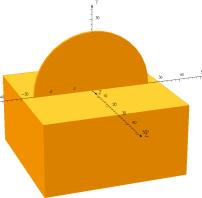
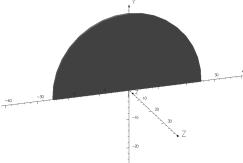
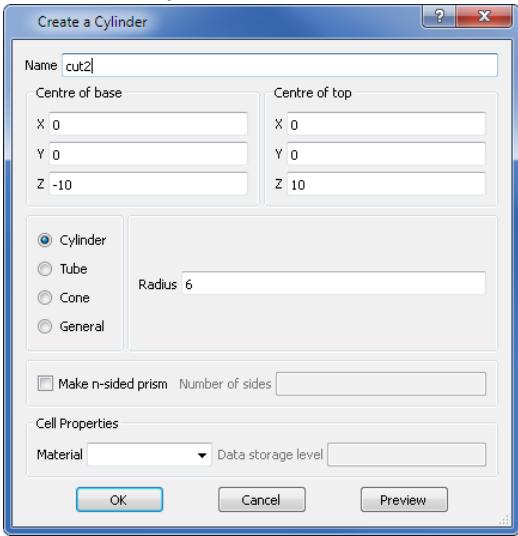
### Creating the Geometry

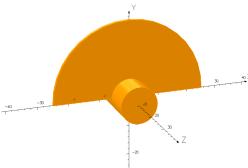
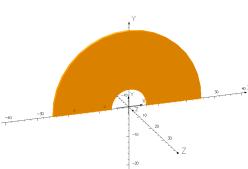
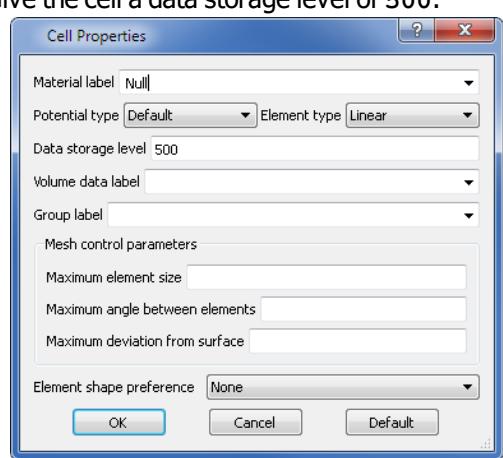
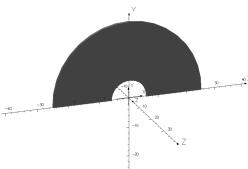
The geometry is created from performing boolean operations on a set of primitives to create the required shape and then adding in an air region to encompass the model.

#### Building the first plate

The first plate will be built from cutting away from a short, flat cylinder. This will then be copied to create the other plates.

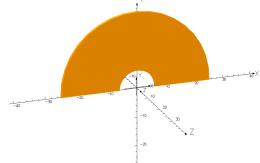
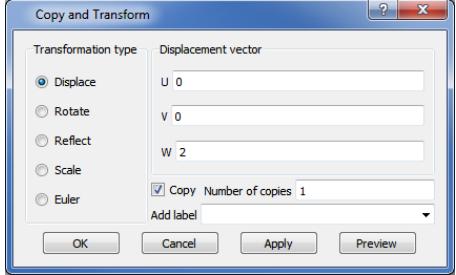
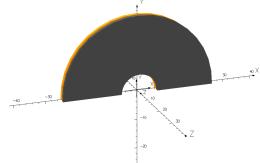
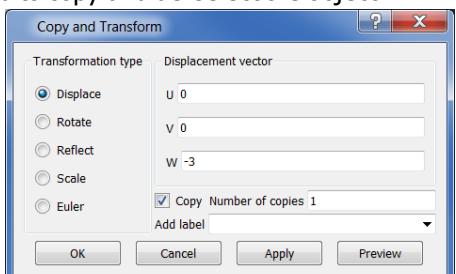
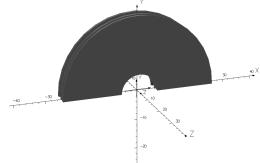
 <b>Create Cylinder</b>	<p>First create the initial disk, base <math>(0, 0, 0)</math> and top <math>(0, 0, 1)</math>, radius 25.</p> <p>The material is set to <b>null</b> which means that the inside of the disk will not be included in the mesh. The field inside the plate is zero and the voltage on the surface is defined by the boundary conditions which will be applied.</p> 	
 <b>Create Block</b>	<p>Then create a block <math>(-30, -30, -30)</math> to <math>(30, 0, 30)</math> which will be used to cut away half of the disk.</p> 	

 <b>Pick entity</b>  <b>Pick Bodies</b>	<p>Pick the disk by double-clicking the left mouse button once the object has been high-lighted. Then pick the block by clicking the right mouse button. Note that it is essential to pick the disk first and the block second in order to ensure that the block is removed from the disk and not the reverse.</p>	
<p>Right mouse to view context menu, select: <b>Combine Bodies -&gt; Subtraction, with regularization</b></p>		
	<p>The overlap and the block are both removed from the model to leave the half disk.</p>	
 <b>Create Cylinder</b>	<p>The inner part of the disk will also be removed. This is done by first creating a cylinder from <math>(0, 0, -10)</math> to <math>(0, 0, 10)</math>, radius 6 which will be used to cut away from the centre of the disk.</p>	

 <b>Pick Bodies</b>	<p>By picking the disk first and then the part to be subtracted, the following <b>Subtraction</b> will result in the disk with a hole.</p>	
<p>Right mouse to view context menu, select: <b>Combine Bodies -&gt; Subtraction, with regularization</b></p>		
 <b>Pick Cells</b>	<p>High-light the half disk as a cell so that its properties can be set.</p>	
<p>Right mouse to view context menu, select: <b>Cell Properties</b></p>		
	<p>Give the cell a data storage level of 500.</p> 	

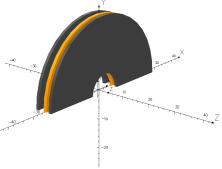
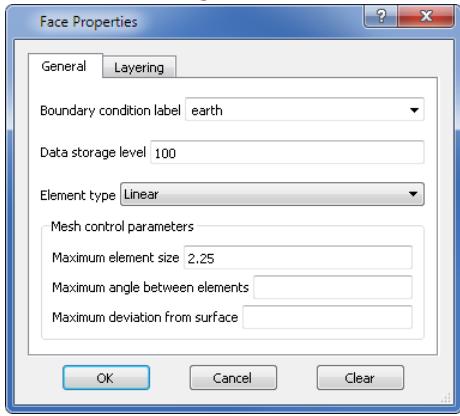
## Creating the other plates

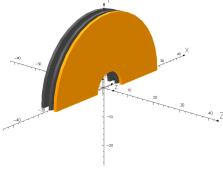
Once one plate has been created and its properties set up, it can be used as a template to create a whole series of other plates. In this example two more plates will be created, with different spacings.

 <b>Pick Bodies</b>	<p>High-light the disk that has already been created.</p>	
<p>Right mouse to view context menu, select: <b>Copy and Transform</b></p>		
	<p>The second plate is created by a displacement of +2 in the Z direction. Using <b>Apply</b> keeps the first disk selected so that it can be copied again.</p> 	
	<p>The third plate is created by a displacement of -3 in the Z direction. This time the <b>OK</b> button is used to copy and de-select the object.</p> 	

## Setting boundary labels

Now that the three plates have been created it is necessary to set up boundary condition labels on the surfaces. These labels will be used later on to assign voltage boundaries to the plates.

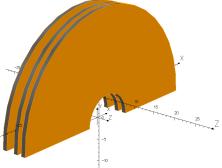
 <b>Pick Bodies</b>	<p>Select the body of the first plate.</p> <p>To select the faces of this body, the filter is changed to faces, and then change the type of picked entity, as follows:</p>	
 <b>Pick Faces</b> then  <b>Change type of picked entity</b>	<p>The display will not change, except in the status bar at the bottom of the window it will indicate that instead of 1 body being selected, there are now 6 of the 18 faces selected.</p>	
<p>Right mouse to view context menu, select: <b>Face Properties</b></p>		
	<p>For the middle plate the faces are given the label <b>earth</b> and a maximum element size <b>2 . 25</b>. An elevated data storage level is also given (<b>100</b>) to ensure that the required face properties take precedence when the plate is later embedded in regions of air.</p> 	

 <b>Pick Bodies</b>	Select the 6 faces that make up the outer surfaces of the second plate. As before, select the body, then change the type of filter as follows:	
 <b>Pick Faces</b> , then  <b>Change type of picked entity</b>	The display will not change, except in the status bar at the bottom of the window it will indicate that instead of 1 body being selected, there are now 6 of the 18 faces selected.	
Right mouse to view context menu, select: <b>Face Properties</b>		

	<p>For the second plate the faces are given the label <code>live1</code>. As before, a maximum element size <code>2.25</code> and elevated data storage level are also given (<code>200</code>).</p>	
	<p>Repeat for the third plate, giving it the label <code>live2</code> and a data storage level of <code>400</code>.</p>	

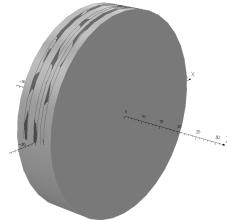
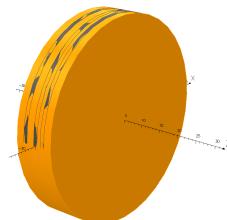
### Layering to improve the mesh

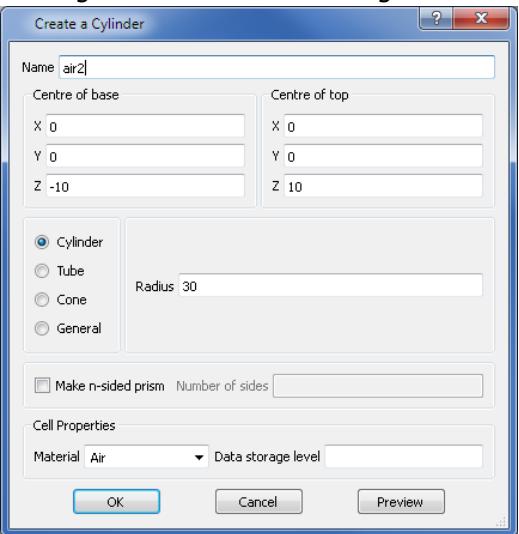
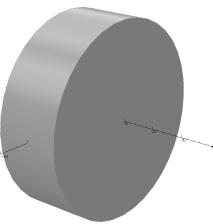
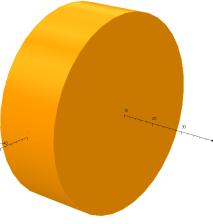
In order to improve the mesh in the air gaps between the plates the Layering feature will be used. This will introduce layers of elements coming away from the plane faces of the plates.

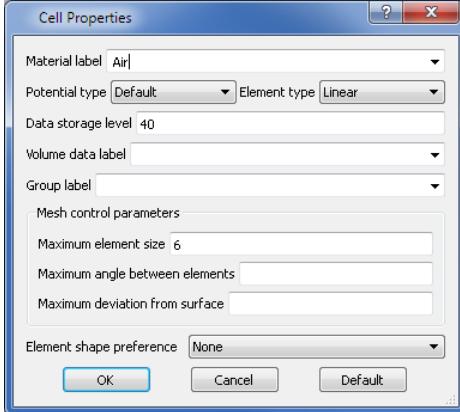
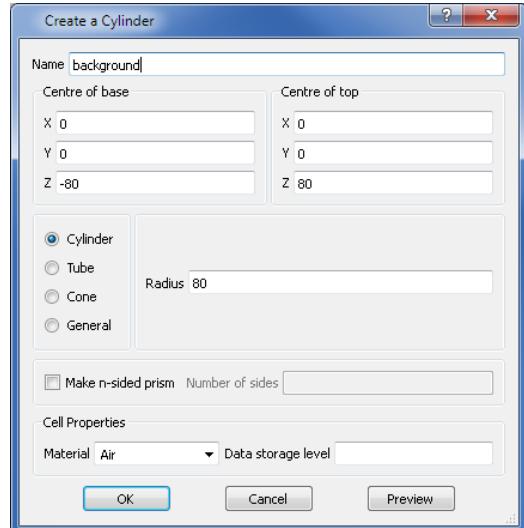
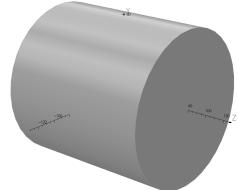
 <b>Pick Faces</b>	<p>The six planar faces of plates need to be picked (not the edge faces)</p>									
<p>Right mouse to view context menu, select: <b>Face Properties</b></p>										
	<p>A single layer of 0 . 5 is set up for each of the selected faces. Backward layering is, in this case, out into the air around the nulled out volumes.</p>  <table border="1"> <tr> <td>Forward layering</td> </tr> <tr> <td>Layering Method: None</td> </tr> <tr> <td>Number of layers: 1</td> </tr> <tr> <td>Layer offset: 0.5</td> </tr> <tr> <td>Backward layering</td> </tr> <tr> <td>Layering Method: Geometry</td> </tr> <tr> <td>Number of layers: 1</td> </tr> <tr> <td>Layer offset: 0.5</td> </tr> </table>	Forward layering	Layering Method: None	Number of layers: 1	Layer offset: 0.5	Backward layering	Layering Method: Geometry	Number of layers: 1	Layer offset: 0.5	
Forward layering										
Layering Method: None										
Number of layers: 1										
Layer offset: 0.5										
Backward layering										
Layering Method: Geometry										
Number of layers: 1										
Layer offset: 0.5										

## Adding background Air Regions

The plate capacitor model needs to be surrounded by air so that a finite element mesh can be created. This will be built in several pieces to simplify the meshing.

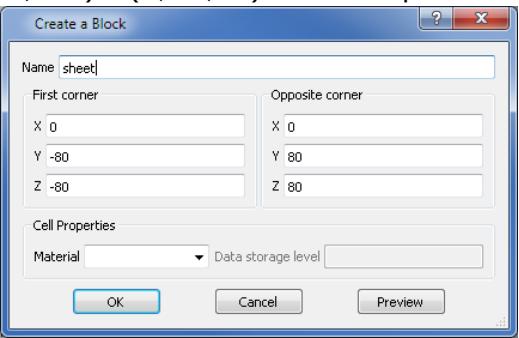
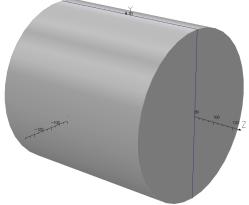
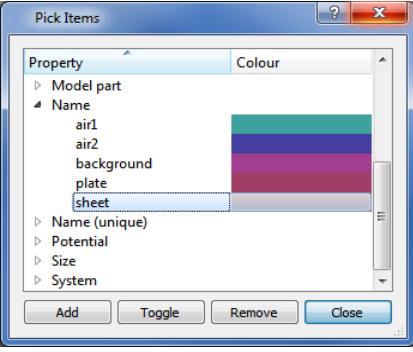
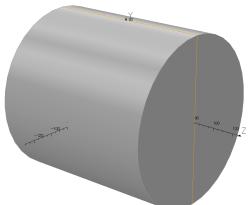
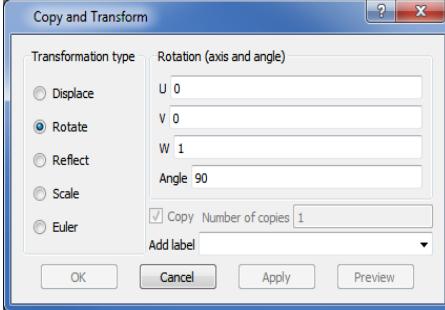
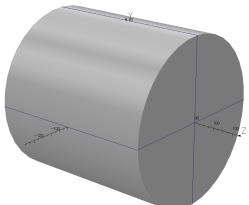
 <b>Create Cylinder</b>	<p>First a cylinder of air is created close in to the plates, from (0 , 0 , -5) to (0 , 0 , 5) radius 25.</p> <p><b>Create a Cylinder</b></p> <p>Name: air1</p> <table border="1"> <tr> <td>Centre of base</td> <td>Centre of top</td> </tr> <tr> <td>X: 0</td> <td>X: 0</td> </tr> <tr> <td>Y: 0</td> <td>Y: 0</td> </tr> <tr> <td>Z: -5</td> <td>Z: 5</td> </tr> </table> <p><input checked="" type="radio"/> Cylinder    <input type="radio"/> Tube    <input type="radio"/> Cone    <input type="radio"/> General</p> <p>Radius: 25</p> <p><input type="checkbox"/> Make n-sided prism    Number of sides: [ ]</p> <p>Cell Properties</p> <p>Material: Air    Data storage level: [ ]</p> <p>OK    Cancel    Preview</p>	Centre of base	Centre of top	X: 0	X: 0	Y: 0	Y: 0	Z: -5	Z: 5	
Centre of base	Centre of top									
X: 0	X: 0									
Y: 0	Y: 0									
Z: -5	Z: 5									
 <b>Pick Cells</b>	<p>High-light the air cell created. This is picked as a cell so that properties can be applied to it.</p>									
<p>Right mouse to view context menu, select: <b>Cell Properties</b></p>										
	<p>Set data storage to 50, and maximum element size to 2 . 25.</p> <p><b>Cell Properties</b></p> <p>Material label: Air</p> <p>Potential type: Default    Element type: Linear</p> <p>Data storage level: 50</p> <p>Volume data label</p> <p>Group label</p> <p>Mesh control parameters</p> <p>Maximum element size: 2.25</p> <p>Maximum angle between elements</p> <p>Maximum deviation from surface</p> <p>Element shape preference: None</p> <p>OK    Cancel    Default</p>									

 <b>Create Cylinder</b>	The second cylinder, named <code>air2</code> , is larger from $(0, 0, -10)$ to $(0, 0, 10)$ radius 30, and used to grade the mesh out to a larger size. 	
 <b>Pick Cells</b>	High-light the new larger cylinder of air.	
Right mouse to view context menu, select: <b>Cell Properties</b>		

	<p>Set the data storage level to 40 and the maximum element size to 6.</p> 	
 <b>Create Cylinder</b>	<p>Finally a <b>background</b> region is placed around the whole model. This is a cylinder of radius 80 which extends from <math>z=-80</math> to <math>z=80</math>.</p> 	

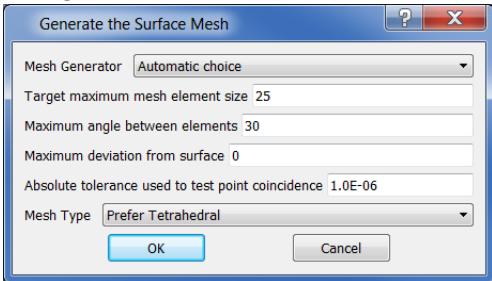
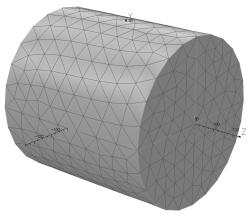
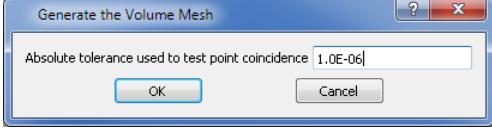
## Model subdivision

The final part of building the model geometry is to use sheet faces to subdivide the model. By cutting the model into simpler cells this makes meshing faster and more reliable.

 <b>Create Block</b>	<p>A sheet face can be inserted by creating a block with zero-length in the X direction, from <math>(0, -80, -80)</math> to <math>(0, 80, 80)</math> in this example.</p> 	
 <b>Pick by Property</b>	<p>As the sheet is obscured, select it by name. High-light the name <b>Sheet</b>, and select the <b>Add</b> button, followed by <b>Close</b>.</p> 	
 <b>Copy and Transform</b>	<p>Select a rotation about the <b>W</b> (Global Z) axis of 90 degrees.</p> 	

## Creating and Meshing the Model

In order to mesh the model it must now be converted to a single Body. Once this has been achieved it can be surface and volume meshed.

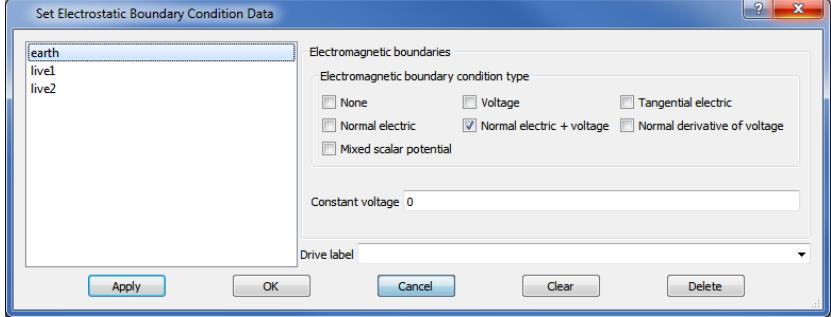
 <b>Create Model Body</b>	This is a single operation that combines all of the bodies into a single body. This is also the stage at which the layering is created.	
 <b>Generate Surface Mesh</b>	A maximum element size of 25 is applied. Smaller values have already been set for the inner air volumes, so this maximum size will only affect the background region. The <b>Mesh Type</b> is left as <b>Prefer Tetrahedral</b> , which, for this example, means that tetrahedral meshing will be used.  	
 <b>Generate Volume Mesh</b>		

## Setting the Analysis Type and Boundary Conditions

The voltage boundary conditions can now be applied to the labels which have been defined on the surfaces of the plates. In order for the correct boundary conditions to be available, the analysis type needs to be set to **Electrostatic**. Use the default settings for this type of analysis.

Next set the boundary conditions.

**Boundary Conditions**

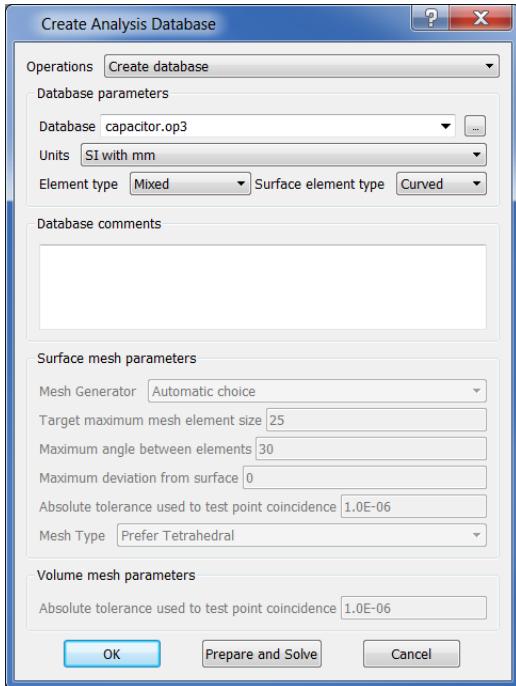


Select the **earth** label and then choose the **Normal electric + voltage** condition and set the **Constant voltage** value to be 0. Rather than pushing the **OK** button, choose the **Apply** button so that the other two boundary conditions can be set as well.

Repeat the above operations for the **live1** and **live2** labels setting each to be +10 volt. Each time a boundary is set push the **Apply** button before exiting the dialogue box using **OK** or **Cancel**.

## Saving the Model File, Creating the Database and Running the Analysis

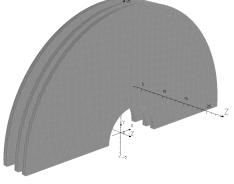
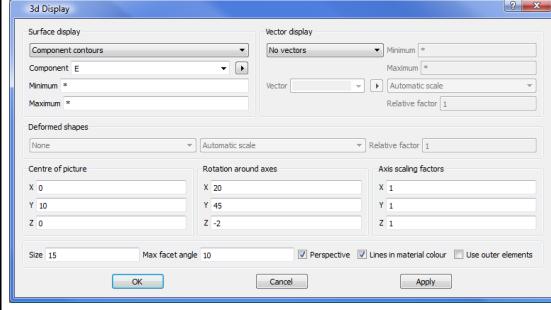
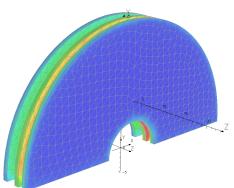
Now that the model is complete, the model geometry can be saved, so that it can be read in again at a later date, and the database can be created ready for solving. Both files are created in the same operation.

 <b>Create Analysis Database</b>	<p>The model and database files are created by specifying the unit system, here selected as <b>SI with mm</b>, and giving the file a name, <b>capacitor</b>, onto which the program will add the correct extensions for the model and database files (<b>opc</b> and <b>op3</b>).</p> 
<p>Press the <b>Prepare and Solve</b> button to create the files and start the analysis.</p>	

## Post-Processing

### Loading and Displaying the Model

Once the analysis is complete select **Post-Process** on the solver window to launch the Post-Processor.

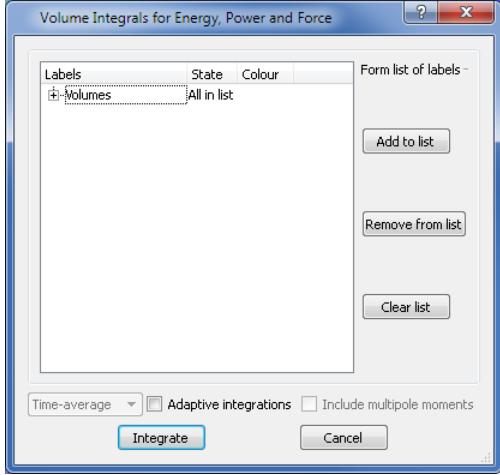
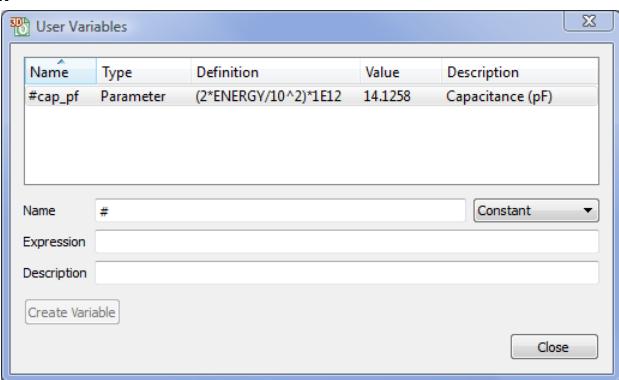
	<p>The default display chooses the parts of the model to be displayed depending on the type of solution that is saved within the database.</p> <p>As this is an electrostatic model, all the voltage boundaries are displayed by default.</p>	
 <b>3d Display</b>	<p>The electric field, <math>\mathbf{E}</math>, can be displayed on the surface of the model by choosing <b>Component contours</b> for the <b>Surface display</b> and entering the desired <b>Component</b>. The electric field strength can also be chosen from the pull right arrow.</p> 	

### Computing the Capacitance

The capacitance,  $C$ , of the system can be computed directly from the stored energy,  $E$  and the known plate voltage,  $V$ .

$$E = \frac{1}{2} CV^2 \quad (4.1)$$

So by computing the stored energy in the system, and from knowing the applied voltages, the capacitance can be found.

 <b>Energy, Power and Force</b>	<p>The default condition is to perform the integral over all volumes within the model. So simply press the <b>Integrate</b> button.</p>  <p>This integral reports back the total stored energy in the model of around 0.7 nJ, which is stored in the system variable, <b>ENERGY</b>.</p>
 <b>User Variables</b>	<p>The capacitance, in pF, is calculated by defining a parameter (<b>#cap_pf</b>) as below.</p> <p>In the <b>Name</b> field, type <b>#cap_pf</b>.  From the pull-down menu, select <b>Parameter</b>.  In the <b>Expression</b> field, type <b>(2*ENERGY/10^2) *1.E12</b>.  In the <b>Description</b> field, type <b>Capacitance (pF)</b>; a <b>Description</b> is optional.  Press <b>Create Variable</b> and <b>#cap_pf</b> will be created and its current value will be displayed.</p> 

## Model Variants

### Varying boundary conditions

So far, in this example, the value for the capacitance has been calculated with both of the live plates having an applied voltage. As the live plates have separate boundary labels it is possible to calculate the contribution from each plate. To do this, two further simulations must be run. Open the existing



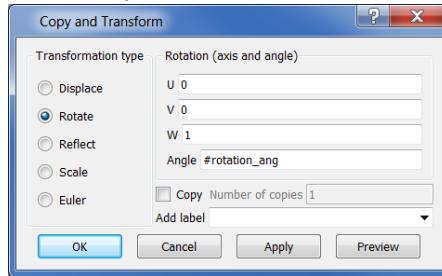
model in the Modeller and select **Boundary Conditions**. Choose boundary condition label **live1** and change the value in the **Constant voltage** field from 10 to 0 volt. Re-run the simulation. Return to the Modeller again and change boundary condition label **live1** back to 10 volt, and set **live2** to 0 volt. Run for a third time. The Post-Processor will then be able to calculate the contribution from each plate. Owing to their different separations, the contribution from each plate will not be the same.

### Rotating vanes

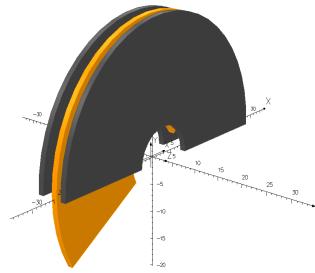
A second instructive variation to the model is to run the analysis for cases where the centre plate is rotated, simulating a variable capacitor. If desired, the rotation angle may be entered as a parameter, to allow easy variation. The existing model may be modified to incorporate this.

**Pick Bodies**

In the Modeler, delete the model body, select **Pick Bodies** and hide the air blocks and cutting planes. Right-click on the centre plate and choose **Transform** followed by **Rotate**. In the dialog, enter the required rotation angle, in degrees, or as a new parameter name.



If a parameter name is used, for example, `#rotation_ang`, the **Unknown User Variable** dialog will appear. Enter the required value and optionally give a description. Press **OK** in this and in the **Transform** dialog.



Select **Unhide Entities** button to display the full geometry again. The new model may be analysed simply by creating the model body, meshing and launching the solver with a new database name.

# **Chapter 5**

## **Radiation Screen Transient EM Example**

### **Introduction**

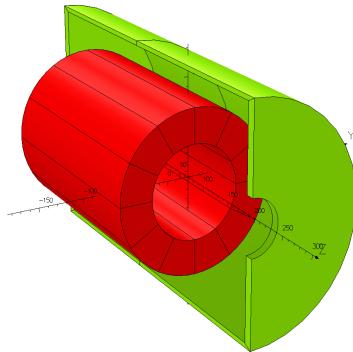
---

This example uses the **Transient Electromagnetic** solver to analyse eddy currents and forces on a thermal radiation shield around a solenoid conductor. The model is built in the Modeller and then analysed in Transient EM before the results are examined in the Post-Processor.

The first section involves defining the model of the radiation shield and the surrounding air. A drive is applied to the solenoid which ramps the source from approximately  $200 \text{ A mm}^{-2}$  down to zero.

In the Post-Processor the forces between the coil and the shield are computed using three different methods, to compare the relative merits of these methods.

Figure 5.1 shows one half of the thermal radiation shield around the solenoid coil.



*Figure 5.1 Radiation Shield Model*

## Building and Running the Model

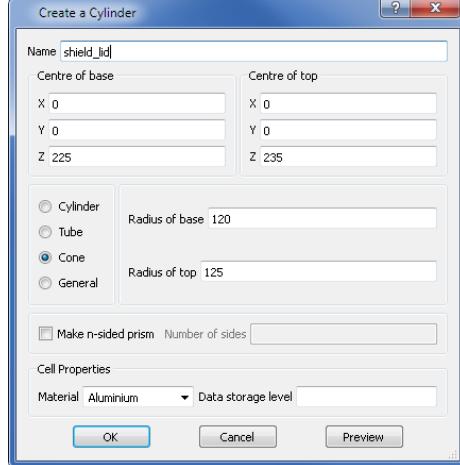
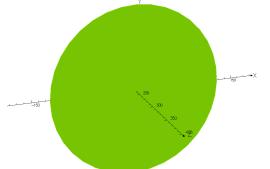
### Creating the Geometry

The model consists of a 90° sector, exploiting the symmetry inherent in the device.

The only material being modelled is the radiation shield. The coil is modelled as a Biot-Savart conductor, which does not form part of the mesh. The only other components are cylinders of air around the coil and the shield which define the space in which the fields are to be computed.

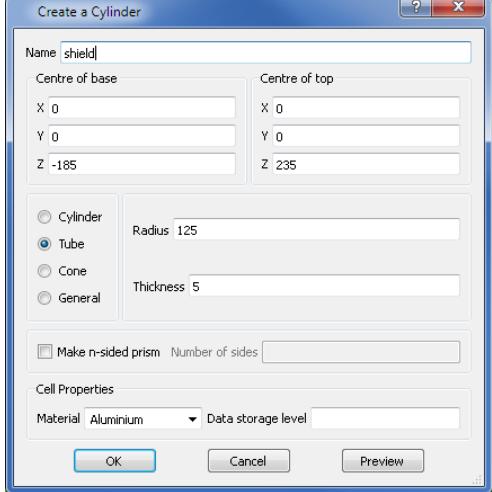
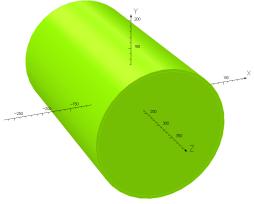
#### Building the aluminium shield

The shield is built up from three sections. A tube of uniform cross-section forms the length of the shield; the base and lid are short cones. Defining the geometry in this way allows hexahedral meshing to be used, which can give significant benefit in the required simulation effort. For more information on the various mesh options available in Opera-3d, please see [Meshing in the Modeller \[page 541\]](#). The base and lid could be built as tubes to provide the central hole in the shield. However, holes will be created later using air cylinders. This allows more flexibility over the properties of the air that will fill the holes.

 <b>Create Cylinder</b>	<p>First create the lid of the shield. Select the <b>Cone</b> radio button and enter <b><math>z=225</math></b> and <b><math>z=235</math></b> for the base and top centres, with a base radius of <b>120</b> and a top radius of <b>125</b>; the material is <b>Aluminium</b>.</p>  <p>The dialog box shows the following settings:</p> <ul style="list-style-type: none"> <li>Name: shield_lid</li> <li>Centre of base: X: 0, Y: 0, Z: 225</li> <li>Centre of top: X: 0, Y: 0, Z: 235</li> <li>Shape: Cone (selected)</li> <li>Radius of base: 120</li> <li>Radius of top: 125</li> <li>Cell Properties: Material: Aluminium</li> </ul>	
---	--	---

The base of the shield is a similar cone. Please construct it in the same way, using the parameters  **$z=-185$**  to  **$z=-175$**  with a base radius of **125** and a top radius of **120**; again the material is **Aluminium**.

The remaining part of the shield is a simple cylindrical tube:

 <b>Create Cylinder</b>	<p>Select the <b>Tube</b> radio button. The base and top of the tube are at <math>z=-185</math> and <math>z=235</math>; the (outer) radius is 125, with a thickness of 5. The material is <b>Aluminium</b>.</p> 	
---	--	---

Once the bodies required to make the shield have been created, they may be selected and combined to form an entity that is suitable for hexahedral meshing.

When selecting objects for manipulation there are several methods that can be used:

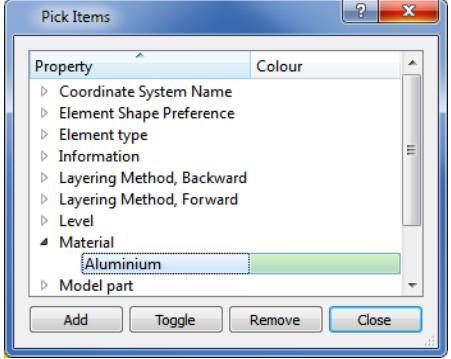
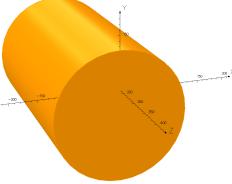
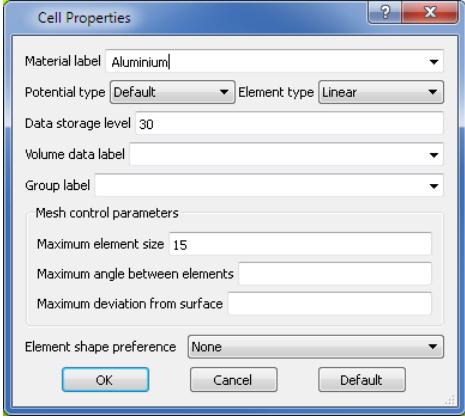
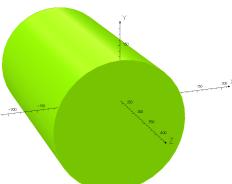
- picking graphically with the mouse by double-clicking on an object,
- right-clicking on an object,
- using the  **Pick All Filter Type Entities** option or
- using the  **Pick by Property** option .

In this instance, the first option is used.



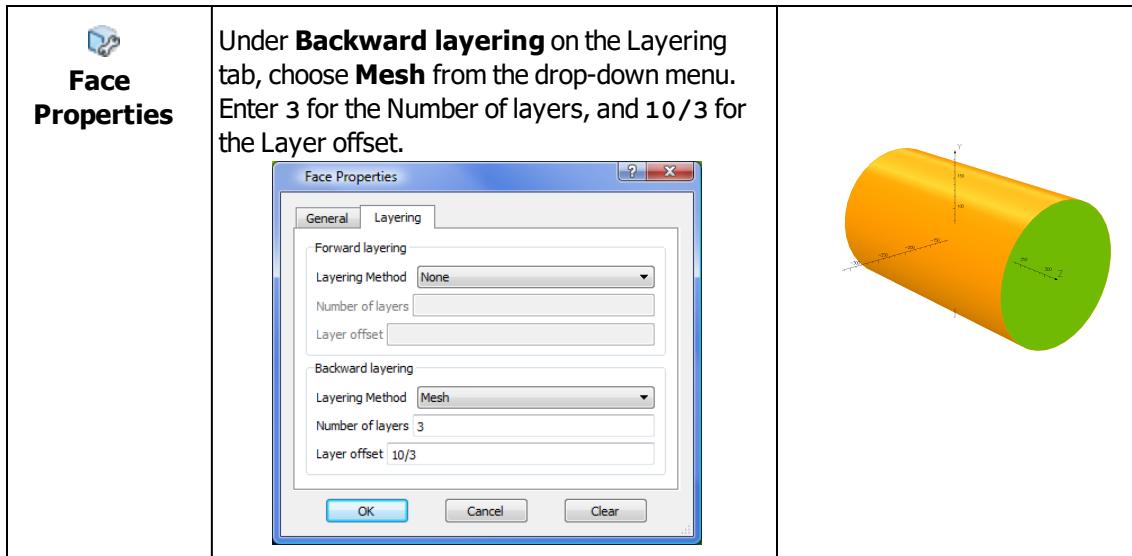
Ensure that **Pick Bodies**, , is selected, and double-click first on the outer tube, then on the lid and base of the shield. Right-click to show the context menu and select **Combine Bodies > Trim Overlap, with Regularization**. This operation removes the region of the cylindrical tube where it overlaps with the conical ends. The result is similar to a mitred joint, and gives the desired suitability for hexahedral meshing.

The data storage level and mesh size of the shield may now be defined. Since all objects created so far are to become part of the shield, we could simply pick all of the materials.

 <b>Pick by Property</b>	<p>Click on <b>Material</b> to high-light this Property in the tree and hit the <b>Add</b> button to make the selection. Click on the <b>Close</b> button to leave the dialog.</p> 	
 <b>Cell Properties</b>	<p>Set a high <b>Data storage level</b> of 30 and a defined mesh size of 15. Click on the <b>OK</b> button to leave the dialog.</p> 	

The mesh size of 15, as set above, is suitable along the shield. However, the mesh should be smaller through the thickness of the wall. This may be achieved in a number of ways, the chosen one here is

to use the face layering feature. Ensure that  **Pick Faces** is selected, and double-click on the two cylindrical faces on the outside of the shield. Right-click to show the context menu and select  **Face Properties**.



The effect of this setting is to force three layers of mesh elements to be generated in the wall immediately behind the face, each with a maximum thickness of, in this case  $10/3$ . This will only become evident once the mesh is created. Note that it is not necessary to define layering on the end walls of the shield; these will be layered automatically to maintain continuity of the mesh.

For more information on layering, please see the layering section of [Meshing in the Modeller \[page 541\]](#).

## Defining the conductor

A solenoid conductor is used in this example to provide a flux of approximately 8 Tesla in the Z-direction. The solenoid is created in a single dialog, but with 2 tab panes. You will need to define

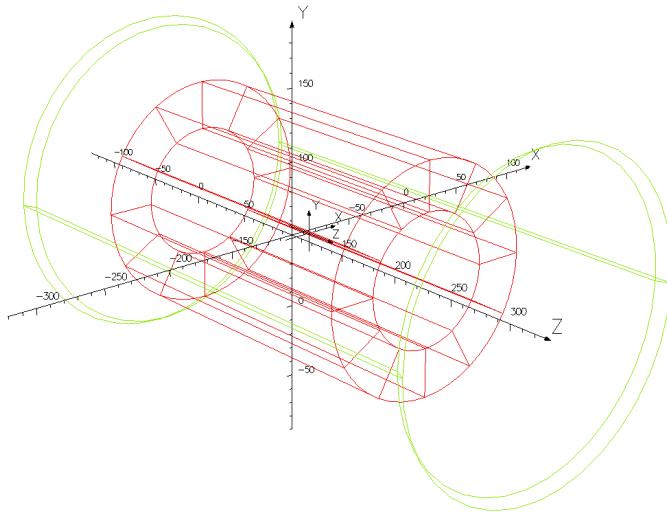
- the geometry,
- the drive (current density and field calculation tolerance in flux density units), and
- the orientation (setting a local coordinate system).

**New Solenoid**

Fill in the **Solenoid Parameters** tab with the dimensions. The solenoid has an inner radius of 50, an outer radius of 87, and is a total of 250 length units long.

Fill in the **Further Options** tab with a current density of 196.373, tolerance of 1.0E-04 and a drive label **one**. The **Coordinate Systems** need to be updated with **Theta**, **Phi** and **Psi** set to 90 each to align the coil along the z axis.

Then press the **OK** button to create the conductor.



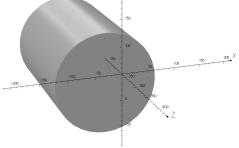
### Creating air regions around the model

Separate cylindrical air regions will be created around the coil and shield to help grade and control the mesh. These will also be used in the Opera Post-Processor to define regions over which force calculations can be performed.



Two of the new air regions will reside inside the aluminium shield. Use the **Select** option to hide the shield so that these air regions will be visible.

The first cylinder of air to be created is used to define a **REDUCED** potential region around the coil. Note that a Biot-Savart conductor must always sit in a region of **REDUCED** potential. The cylinder is first created, and then selected so that the cell properties can be set.

 <b>Create Cylinder</b>	<p>Create a cylinder of Air which extends from <math>z=-140</math> to <math>z=140</math> with a radius of 100.</p> <p><b>Create a Cylinder</b></p> <p>Name: reduced_air</p> <p>Centre of base:</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: -140</li> </ul> <p>Centre of top:</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 140</li> </ul> <p><input checked="" type="radio"/> Cylinder</p> <p><input type="radio"/> Tube</p> <p><input type="radio"/> Cone</p> <p><input type="radio"/> General</p> <p>Radius: 100</p> <p><input type="checkbox"/> Make n-sided prism Number of sides: [ ]</p> <p>Cell Properties:</p> <p>Material: Air Data storage level: [ ]</p> <p>OK Cancel Preview</p>	
 <b>Pick Entity</b>  <b>Pick Cells</b>	<p>Right-click on the new cylinder and select  <b>Cell Properties</b> from the context menu. Set the data storage level to 40 and the mesh size to 15. Set the <b>Potential type</b> to <b>Reduced</b>. Giving a <b>Volume data label</b> of <b>Air_Reduced</b> will allow this volume to be picked using this label in the Post-Processor.</p> <p><b>Cell Properties</b></p> <p>Material label: Air</p> <p>Potential type: Reduced</p> <p>Data storage level: 40</p> <p>Volume data label: Air_Reduced</p> <p>Group label:</p> <p>Mesh control parameters:</p> <ul style="list-style-type: none"> <li>Maximum element size: 15</li> <li>Maximum angle between elements:</li> <li>Maximum deviation from surface:</li> </ul> <p>Element shape preference: None</p> <p>OK Cancel Default</p> <p>Click on <b>OK</b> to close the dialog.</p>	

A background region will be created later as part of the operation to define the **Model Symmetry**. This region will then exist where there is no other geometry, including, in the present model, in the space between the reduced potential region and the screen and in the space outside the screen, up to the boundary of model. Over much of the volume of the background region, the fields vary slowly in space, and so a coarse mesh may be used for computational efficiency. However, close to the screen, the fields may vary more rapidly, so a finer mesh needs to be defined. Two further air

cylinders are used to achieve this; one fills the space between the reduced potential region and the screen, the other has a radius somewhat larger than the screen, so moving the background region away from locations of rapidly varying field, and allowing mesh size grading into the background region.

Each cylinder needs to have its cell properties set up with potential type **Total** and material label **Air** to move the Reduced/Total boundary away from the shield and for use in the Post-Processor later in this example.

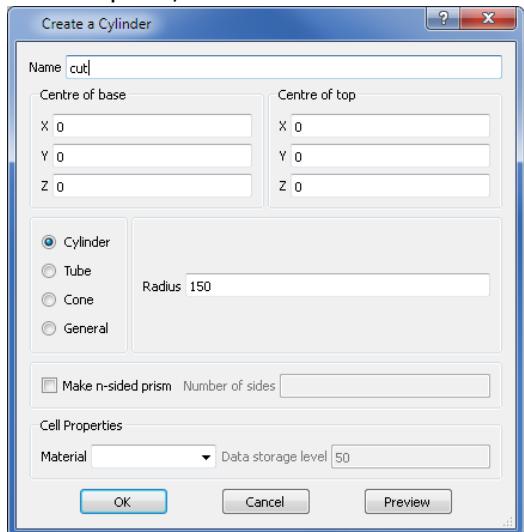
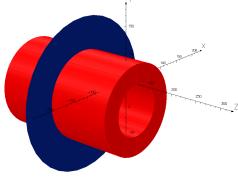
Create each cylinder using the details in this table and set up the cell properties accordingly.

Name	Base Centre	Top Centre	Radius	Data storage level	Mesh size
<b>inside_air</b>	(0, 0, -175)	(0, 0, 225)	120	20	30
<b>outside_air</b>	(0, 0, -220)	(0, 0, 270)	150	15	25

Two additional cylinders of air are required, which are given a high data storage level, and are used to cut the holes in the end walls of the shield. The details of these are:

Name	Base Centre	Top Centre	Radius	Data storage level	Mesh size
<b>cut_1</b>	(0, 0, -175)	(0, 0, -185)	30	50	25
<b>cut_2</b>	(0, 0, 225)	(0, 0, 235)	30	50	25

The final stage in setting up the geometry is to create an infinitely thin plane that will cut the cells when creating the model body. This feature makes meshing quicker because it reduces the number of elements per cell and simplifies the cell shapes.

 <b>Create Cylinder</b>	<p>Notice that the cylinder is created with zero-length, which will create an infinitely thin surface in the X-Y plane; a radius of 150 is used.</p>  <p>Using the  <b>Select</b> option allows just the cut plane and the coil to be viewed.</p>	
---	--	---

## Setup the Analysis Options

Having defined the geometry, the type of analysis to be run and the analysis settings need to be defined. In the **Analysis Settings** window replace the default analysis with a **Transient Electromagnetic**. This changes the available options in the rest of the model definition, and allows the appropriate analysis setting dialog to be accessed.

In the configuration pane of the analysis, set the **Time-stepping** options to **Adaptive**, with an **Initial time-step** of 0 . 05 seconds and a **Maximum error** of 0 . 5 %. The **Adaptive** option allows the time-step to be adjusted, to keep the errors in the solution below the maximum.

In the **Output-times** list enter the following sequence: 0 . 25 ; 2 ; 0 . 25 . This specifies a series of output times starting at 0.25, ending at 2 and with a step of 0.25 seconds.

## Defining the model symmetry

Because the model is symmetric around the Z-axis, it is not necessary to compute the whole model. Instead only a 90 degree section will be analysed.

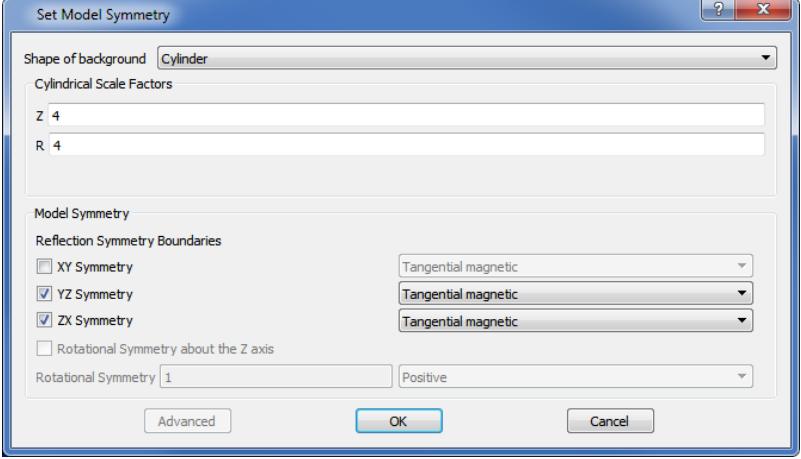
The  **Model Symmetry** option can be used to define, in one easy step,

- the far field volume (Background region),
- the type of symmetry (reflection and/or rotation)
- the boundary conditions to be applied to the symmetry planes.

In this case,  **Model Symmetry** creates a segment of a cylinder whose outer radius and axial length are calculated by scaling the outer dimensions of the rest of the model. Boundary conditions are applied to the surfaces of the segment to define the field symmetry. At a later stage, parts of the model outside this segment will be removed.

  
**Model Symmetry**

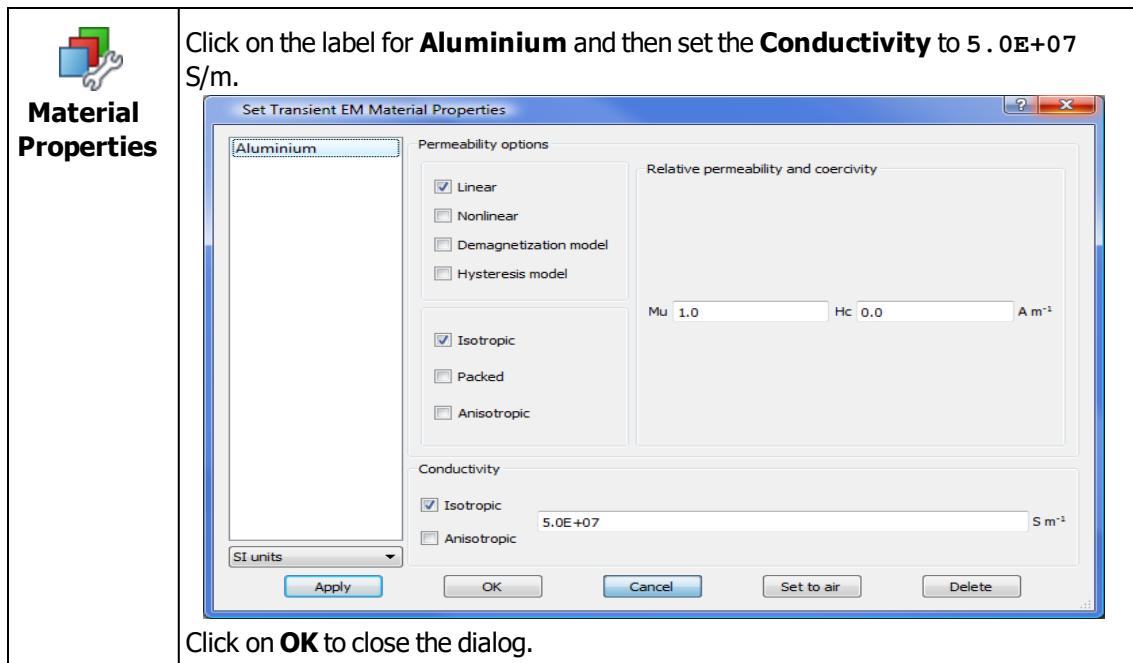
Set the background shape to **cylinder** and scaling factors of **4** in Z and R.  
Set **Tangential magnetic** symmetry in both the **YZ** and **ZX** planes.



Click on **OK** to close the dialog.

## Defining material properties and drives

In order for eddy currents to flow in the radiation shield it needs to have an electrical conductivity defined.

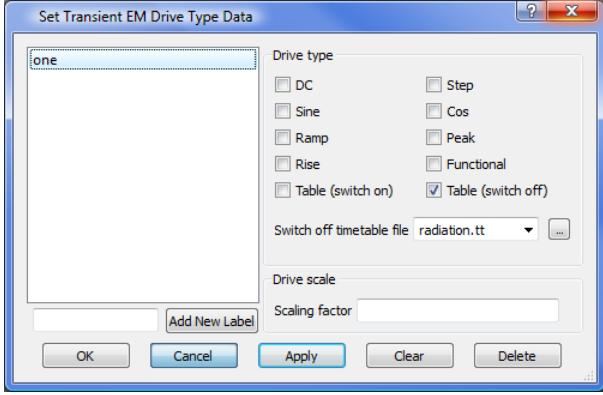


This example is intended to show the effects of a fast ramp down (i.e. a switch-off characteristic) during the quenching of a superconducting magnet. A time table must be created to model this. A time table is simply a two column text file; column one is the time in seconds and column two is the drive scaling factor at these times.

Create a time table using a text editor with the following data and save it in the project folder with the filename ***radiation.tt***.

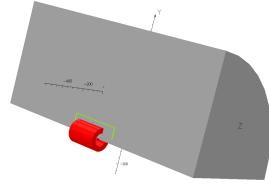
0.0	1.0
0.1	0.98
0.2	0.85
0.3	0.55
0.5	0.25
0.8	0.08
1.3	0.01
1.8	0.0
2.0	0.0

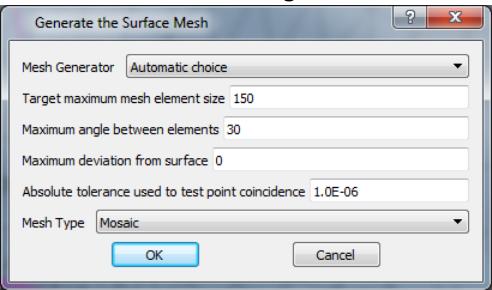
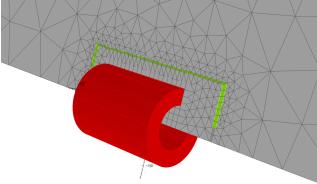
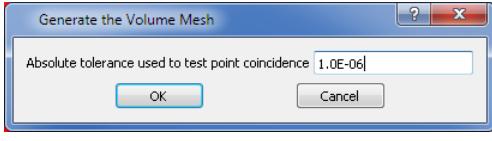
Now that the time table file has been created it must be attached to the coil as a drive function using the coil drive label.

 <b>Drives</b>	<p>Click on the drive label <b>one</b> that was assigned when defining the conductor and tick the <b>Table (switch off)</b> box. Then enter the time table filename in the box provided.</p>  <p>Click on <b>OK</b> to close the dialog.</p>
--	--

## Meshing the Model

The final stage before creating a database and solving it is to create the model body and mesh the model.

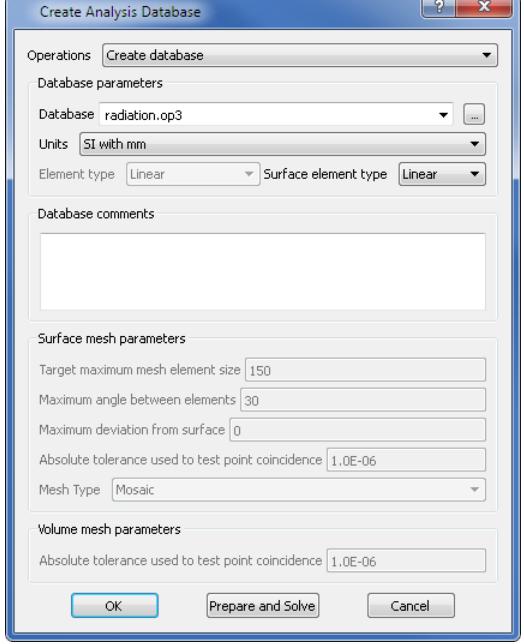
 <b>Create Model Body</b>	<p>Create the model body to combine the volumes and create the background region used to apply the symmetry.</p>	
---	--	---

 <b>Generate Surface Mesh</b>	<p>Set the maximum element size to be 150 for the surface mesh; this will be applied where no more stringent criterion exists. The Mesh Type should be <b>Mosaic</b>. The remaining options may be left at their default settings.</p>  <p>Click on <b>OK</b> to close the dialog.</p>	
 <b>Generate Volume Mesh</b>	<p>Select the default settings for the volume mesh.</p>  <p>Click on <b>OK</b> to close the dialog.</p>	

## Solving the Model

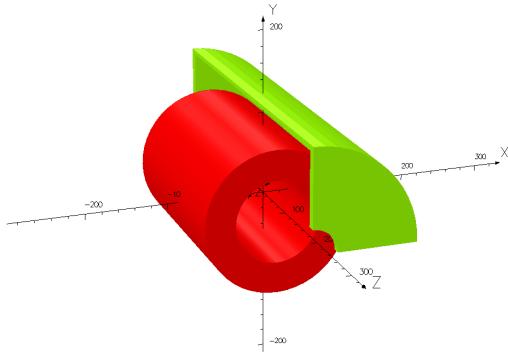
Before the model can be solved, the units and element types must be chosen, and an analysis database (an **op3** file) must be created and saved.

Both files are created with the following command:

 <b>Create Database</b>	<p>Give the database a name, for example, <b>radiation</b>; the extension (<b>.op3</b>) will be added automatically. Set the units to <b>SI with mm</b> and <b>Surface element type</b> to <b>Linear</b>. Clicking on <b>Prepare and Solve</b> will create and save the database, and launch the solver.</p>  <p>This command also saves the model geometry and information on materials and analysis settings to an <b>opc</b> file, which may be read and edited by the Modeller. Note that an <b>opc</b> file may be saved at any point in build process by selecting <b>Save as New Model Data</b>.</p>
---	--

## Post-Processing

Once the solution has completed, pressing the **Post-Process** button at the bottom of the solver window will close the solver window and launch the Post-Processor.



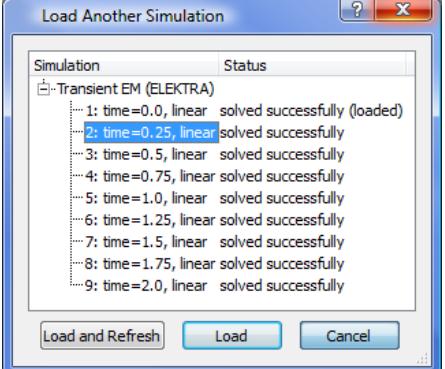
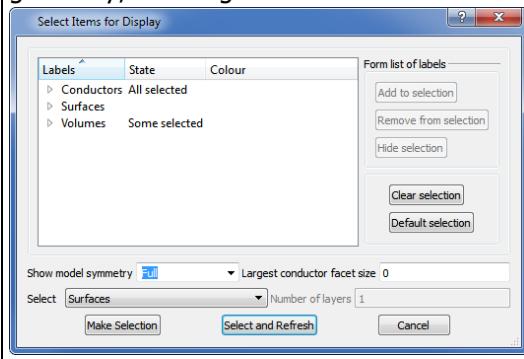
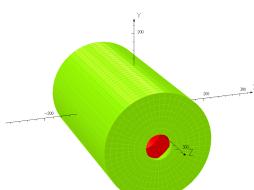
*Figure 5.2 Radiation Shield*

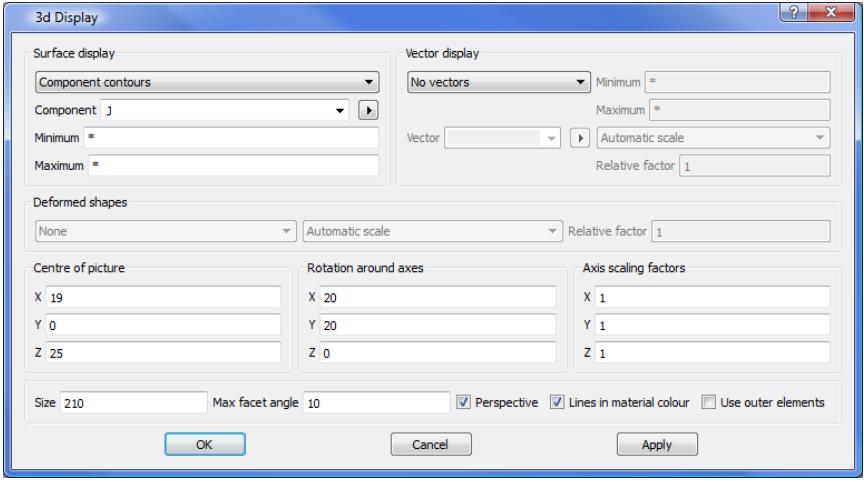
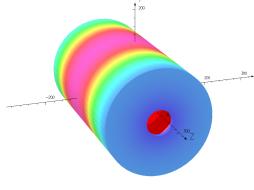
In this example the main interest is in the eddy currents induced in the shield and the forces between the coil and the shield.

As the current in the coil reduces, so the magnetic field will be reduced and this changing field will induce eddy currents in the highly conductive radiation shield. These currents will couple with the field from the magnet and so produce a force. Because the radiation shield is not symmetric about the coil there will be a net axial force which can be calculated.

### Displaying Eddy Current Density

When the Post-Processor is started from the solver window, Case 1 is loaded initially, and the list of all available cases in the database is displayed. Since Case 1 is at time 0, just before the current starts to change, it is necessary to load a different case in order to view the eddy currents and compute the quench forces.

	<p>Select <b>Simulation number 2.</b></p>  <p>Click on <b>Load and Refresh</b>.</p>	
 <b>Select</b>	<p>Set <b>Show model symmetry to Full</b>. Click on <b>Select and Refresh</b> to display the geometry, including the rotations.</p> 	

 <b>3d Display</b>	<p>Display the geometry with contours of <math>\mathbf{J}</math> on the shield surface.</p>  <p>Click on <b>OK</b> to close the dialog.</p>
 <b>Outline View of Model</b>	<p>Remove the mesh lines to make the picture clearer.</p> 

## Computing Forces

Several methods can be used to compute forces in between the quenching magnet and the radiation shield in this model. Each method will have a different accuracy depending on the way the fields are being computed. More generally, the use and applicability of these different methods will depend on the geometry of the model; this is discussed briefly at the end of this example.

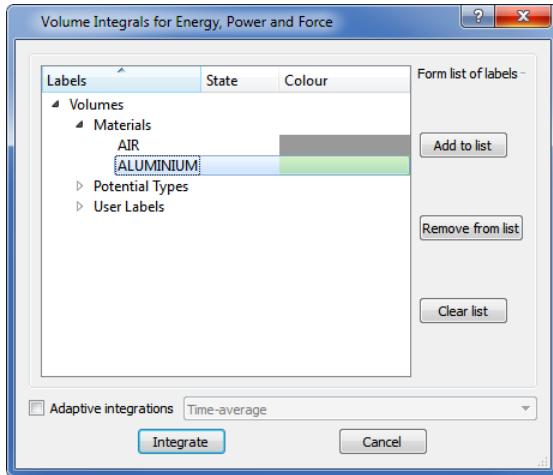
### Lorentz forces in the shield

For this model, where eddy currents are induced in a shield, integration of Lorentz forces ( $\mathbf{J} \times \mathbf{B}$ ) is a very quick and accurate method of calculating forces, because the current and flux densities come directly from the potential solution.



### Energy, Power and Force

First click the **Clear list** button and then add just the material that makes up the shield for the integration. Open the **Volumes** and **Materials** branches of the tree and select **ALUMINIUM**. Then click on **Add to list**.

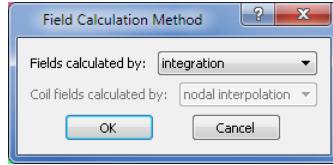
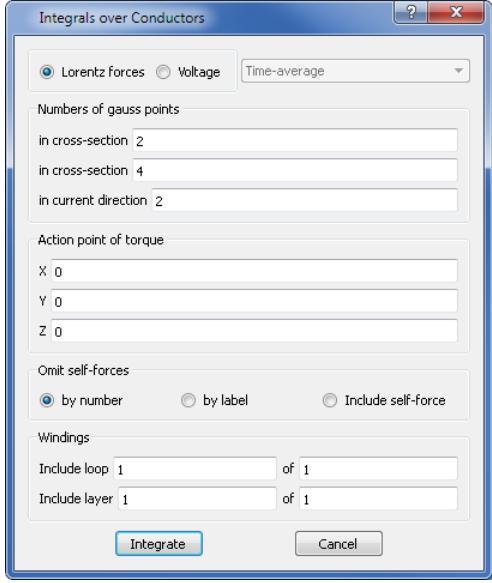


Click on **Integrate** to close the dialog. An information box shows the Z force of around 680 N on the shield.

See the **ENERGY** command in the *Opera-3d Reference Manual* for further details of energy, power and force calculations on structures.

## Lorentz forces in the coil

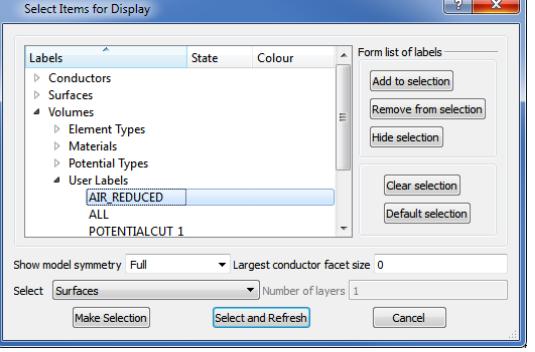
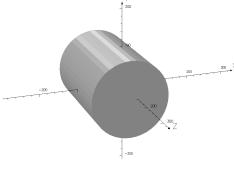
The second method is to integrate  $\mathbf{J} \times \mathbf{B}$  over the coil itself. To get the most accurate result it is necessary to use the most accurate field calculation method: field by integration. The field at each point is calculated by integrating the field from the current density in every element in the shield, rather than from a simple interpolation from the surrounding nodes.

 <b>Field Options</b> > <b>Field Calculation Method</b>	<p>Switch to using Integral Fields.</p> 
 <b>Integrals over Conductors</b>	<p>The more gauss points that are used the more accurate, but slower, the calculation. Using 2 and 4 points, respectively in the radial and axial directions of the coil cross-section, and 2 points in the current direction is sufficient in this instance. Accuracy is improved further by choosing to omit the self forces. Note that the selection of the <b>by number</b> radio button is appropriate for closed-loop conductors.</p>  <p>This time the Z force on the coil is opposing the force on the shield and so is <math>-680</math> N. Adding more gauss points will bring this closer to the result from the previous method.  See the <b>BODY</b> command in the <i>Opera-3d Reference Manual</i> for further details of the calculation of force and other integral functions over conductors.</p>

## Maxwell stresses

The final method considered in this example is to integrate the Maxwell stress over a defined surface in the model. Any volume or surface can be selected for this purpose. In this example, a volume label was added to the reduced potential air region around the coil so that this can be selected for the integration.

Before selecting the integration surface, change the **Field Calculation Method** back to **nodal interpolation** and change the  **3d Display** to show **Material colours**.

 <b>Select</b>	<p>First click on <b>Clear selection</b> before using <b>Add to selection</b> to include just the <b>AIR_REDUCED</b> volume label.</p>  <p>Click on <b>Select and Refresh</b> to close the dialog.</p>	
 <b>Force on Selected Surface</b>	<p>Accept the default settings for the integral over the selected surface.</p>  <p>Click on <b>Integrate</b> to perform the force calculation. The forces are displayed on the screen. Again, as this surface is around the coil, it is the force on the coil that is being reported. See the <b>INTEGRATE</b> command in the <i>Opera-3d Reference Manual</i> for further details of the force calculation on surfaces.</p>	

## Further work I - Improving the Model

---

For the model used in the example, the results from the first two methods of calculating the Lorentz forces give very similar results, and this comparison is a good way of checking for accuracy. However the result from the Maxwell stress integral on this model is significantly different.

In this model, the integral surface chosen is quite close to the coil, where the fields will be changing rapidly. In order to model this effectively, it is necessary to use a finer mesh on this surface. To illustrate the effect of this, return to the Modeller and set the mesh size on the outside faces of the reduced potential air region around the coil. The present mesh is determined by the value defined in the cell properties of the reduced potential air region, namely 15mm. Choose a surface mesh size less than this, re-run the simulation, and examine the results of the three methods of force calculations used previously.

This example illustrates the importance of setting appropriate mesh sizes in different regions of a model.

### Discussion

This example presents a number of ways to calculate the forces in a simple model. In this example, any of the methods implemented in Opera are applicable. More generally, the method, or methods, that may be used in any particular case will depend on the details of the model. This is summarized below.

- Both the Maxwell stress and virtual work methods integrate over selected surfaces to calculate the force and torque on the bodies enclosed by the surface.
  - The Maxwell stress method is the most general, and may be used in all cases.
  - The virtual work method (which is not considered in the present example) is restricted to cases where elements surrounding the selected surface represent linear isotropic materials and use **TOTAL** potential.



Both methods use the **INTEGRATE** command to integrate the forces on bodies enclosed by the selected surfaces.

- Where the force arises only from the interaction between a current and a magnetic field, the Lorentz method (i.e. the evaluation of  $\mathbf{J} \times \mathbf{B}$ ) may also be used to calculate the force. There are two available options, using force densities in the structure or using those in Biot-Savart conductors.
  - Evaluation of forces on the structure uses the **ENERGY** command to integrate the Lorentz force on induced currents in the selected volume. It should be noted that this is the only method of force calculation that does not also provide the torque.
  - Evaluation of forces on Biot-Savart conductors uses the **BODY** command to integrate the Lorentz force over the volumes of the selected source conductors.

The latter will be more accurate than any of the other techniques. However, since it uses integral, rather than nodal fields, it may also be the most expensive in computational resource.

## Further Work II - Extracting Field Data During the Simulation

---

In this example, the database was loaded into the Opera Post-Processor after the simulation had completed. It is also possible to extract, and process, field values during the solution, without using the Post-Processor. This feature is enabled using Python scripts to interrogate a database at defined points during the solution process; for a transient solution, this allows field values at each time-step, and at each output time-step, to be accessed as soon as the time has been reached. Using Python, field quantities can be accessed from the database over the whole model, from defined parts of the model - typically by specifying a label, such as a material or volume label - or by defining the coordinates at which to extract data. Data accessed in this way can be processed immediately using Python utilities.

In this example, the flux density components,  $B_x$ ,  $B_y$ ,  $B_z$ , will be accessed at a user definable position in the model for each time step in the solution and logged to a datafile. The point chosen is on the axis of the coil, level with the end of the screen at  $z=230$  mm.

The elements of Python required to enable its use within Opera are installed along with Opera. These include an embedded Opera-specific Python module, `operafea`, and the array handling library, `NumPy`. Additional libraries are widely available should they be required for specific applications. A full description of the implementation of Python in Opera is given in the ***Opera-3d Reference Manual***.

It is assumed that the user has an understanding of how to program in Python, and only the requirements specific to its implementation in Opera will be described in the remaining part of this chapter.

### Python script

No modifications are required to the model to allow data to be accessed at defined points in the model space. However, before the analysis is launched, it is necessary to set up two scripts to control the extraction of field data, and these must be stored in the same folder as the model file. The first is an Opera command script, named `solver.comi`, which is run once when the solver is launched. It specifies when data are to be extracted from the database, and imports a Python script that contains the required Python commands. The commands in the Python file define, for example, which field components are to be extracted, and the processing that is to be performed on the fields. The maths and array functions in the Python libraries give extremely flexible and powerful processing capability, although in this example, only the simple functions of finding and outputting the B field components at a point will be performed. The field values will be written to the Solver window and `.res` file, and stored in a data file.

For this example, an appropriate `solver.comi` and Python script file, `extract_B_at_point.py` can be found in the Opera-3d Examples folder. The first of these includes the following commands:

```
$python 'import extract_B_at_point as ex'  
$python 'operafea.registerHookCallback('on_timestep_  
end',ex.onTimeStepEnd)'
```

The first line identifies the Python script required to be imported, while the second line identifies the relevant 'hook' that has been inserted in Opera. Hooks are used in the Python scripts to specify when particular commands in the Python script are to be run, in this case, at given stages within the solve process. The hook used in this example is `onTimeStepEnd`, which allows the Python commands to be run at the end of each solution timestep.. By selecting alternative hooks, it is possible to perform the Python commands at other points in the solve process; for example, specifying `onStoreTimeOutput` in ***extract\_B\_at\_point.py***, instead of `onTimeStepEnd`, would extract the field at each solution output time.

Note that the commands within the ***solver.comi*** file use the conventions of the Opera command language; in particular, Python commands are initiated using the Opera command **\$ PYTHON**.

File ***extract\_B\_at\_point.py*** contains standard Python command structures, and its operation should be clear from the comments provided in the script.

Information on the Python functionality supplied in Opera and additional usage examples are given in the Supplied Python Functionality section of this User Guide; the complete list of hooks and detailed information of the implementation of Python in Opera is given in the ***Opera-3d Reference Manual***.

## Running the simulation

Once the ***solver.comi*** and ***extract\_B\_at\_point.py*** files have been saved in the same location as the model, the simulation may be launched in the normal way.

The contents of the ***solver.comi*** will be echoed near the top of the solver window and **.res** file. Any errors encountered with the commands will be shown.

## Simulation outputs and post-processing

As stated above, the ***extract\_B\_at\_point.py*** file used in this example includes commands to extract the components of the B field at a point in the model volume space, and to display and output the results at the end of each time step.

During the simulation, results are displayed in the Solver window and **.res**, in the form:

```
----- Data extraction at end of time step -----  
Fields of interest: B (Tesla)  
Coordinate (mm): (0, 0, 235)  
Times (s): 0.75  
Field components (Tesla): (0, 0, 0.0642)  
Field modulus (Tesla): 0.0642  
---- Data extraction at end of this time step -(complete)----
```

## 221 Chapter 5 Radiation Screen Transient EM Example: Further Work II - Extracting Field Data During the Simulation

The data at each time step are also written to a text file, in a format suitable for plotting in the Opera Post-Processor, or in general-purpose graphing tools. The first few lines of text file for this particular example will contain data similar to that shown:

/TTIME_(s)	x_(mm)	y_(mm)	z_(mm)	Bx_(T)	By_(T)	Bz_(T)	Bmod_(T)
0.000E+00	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	5.890E-01	5.890E-01
5.000E-02	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	5.909E-01	5.909E-01
1.000E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	5.846E-01	5.846E-01
1.750E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	5.481E-01	5.481E-01
2.000E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	5.275E-01	5.275E-01
2.500E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	4.615E-01	4.615E-01
3.500E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	2.987E-01	2.987E-01
5.000E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	1.656E-01	1.656E-01
7.500E-01	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	6.418E-02	6.418E-02
1.000E+00	0.000E+00	0.000E+00	2.350E+02	0.000E+00	0.000E+00	2.048E-02	2.048E-02

Once the simulation has finished, the data may be compared with the values of the field extracted in the Post-Processor. [Figure 5.3](#) shows this comparison; the solid line is the Bz field extracted from the solver at each timestep in the solution, while the points are those produced by the Post-Processor at each output timestep (ie from each case in the database).

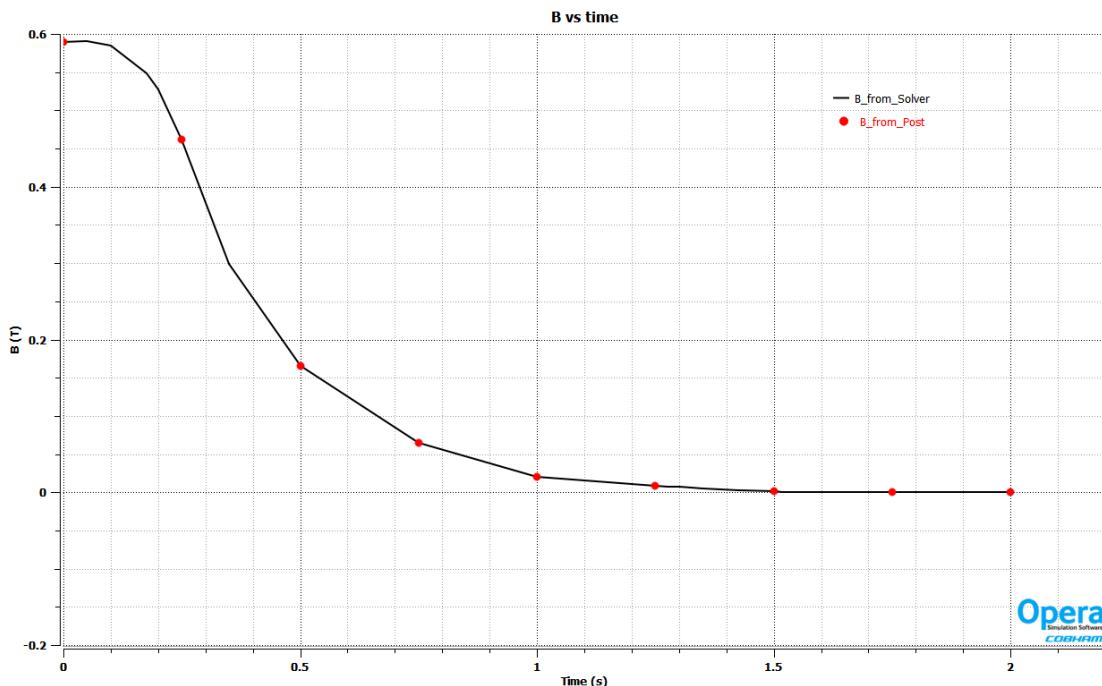


Figure 5.3 B field comparison

# **Chapter 6**

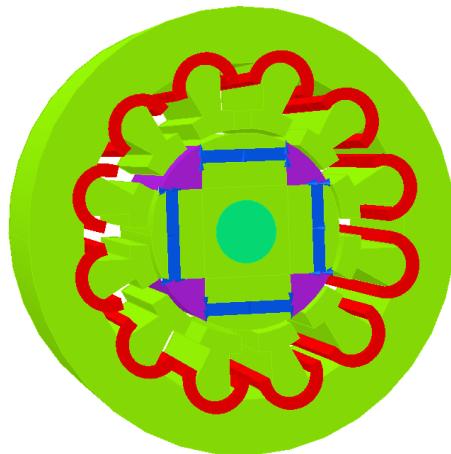
# **Permanent Magnet Synchronous Machine**

## **Introduction**

---

This example details the steps needed for the setup of a 3-dimensional model of a Permanent Magnet Synchronous Machine (PMSM). Two different types of analysis are run: static field analysis, using the **Magnetostatic** solver and a motion analysis, using the **Transient Motion** solver.

The post-processing step is geared towards obtaining results in a form that is relevant to electrical machines designers.



*Figure 6.1 Permanent Magnet Synchronous Machine*

## Magnetostatic model

The model consists of a 4-pole, 3-phase Permanent Magnet Synchronous Machine (PMSM) with concentrated windings and 12 slots. The permanent magnets are embedded in the rotor.

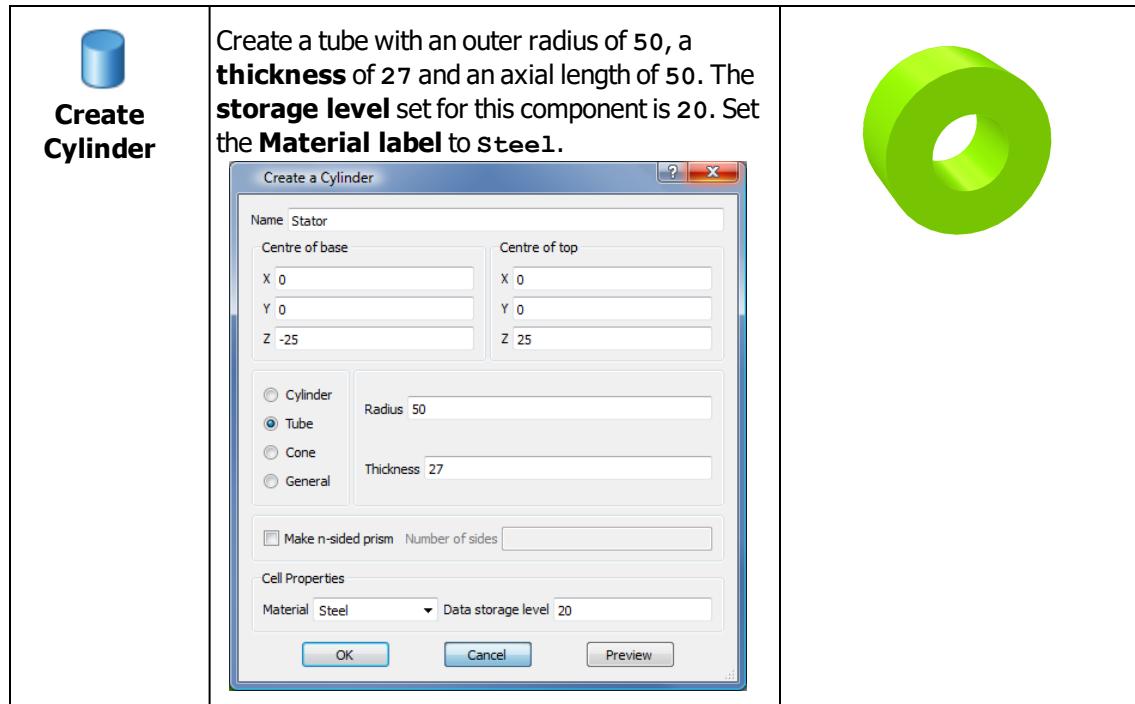
The model building step employs a number of different methods that offer the user an overview of the software capabilities in terms of 3D modelling.

The rotational symmetry that exists in the model is used to reduce the analysed section of the motor to 90 degrees. The model also allows for an axial symmetry boundary, hence only 1/8 of the original model will need to be solved.

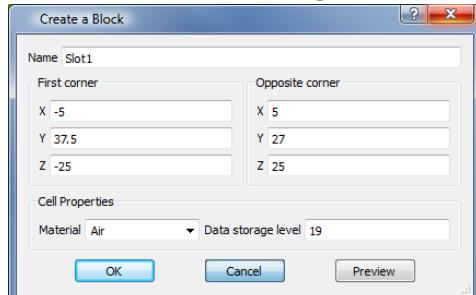
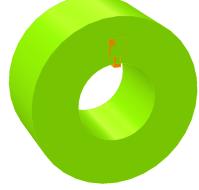
### Creating the geometry

#### Stator

The stator core is built from one cylinder, while the stator slots and teeth are cut out using two blocks.

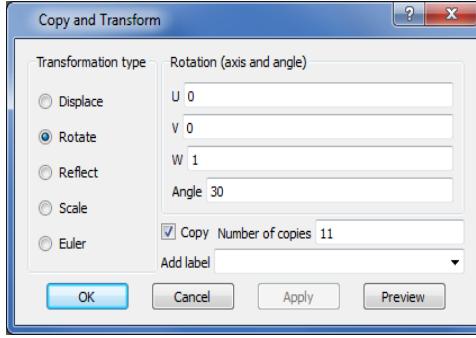


Next, two blocks are used to cut away the slots from the stator back-iron. Since the resulting slots have parallel edges, the stator tooth will be trapezoidal.

 <b>Create Block</b>	<p>Create the block <b>slot1</b> that defines the shape of the upper slot. The corners are at <math>(-5, 37.5, -25)</math> and <math>(5, 27, 25)</math>. The cell material is <b>Air</b> and the <b>storage level</b> is <b>19</b>.</p>  <p>Use the same command to create another block for the lower part of the slot and name it <b>slot2</b>. The coordinates for the second block are <math>(-2, 27, -25)</math> and <math>(2, 20, 7, 25)</math>. Use the same cell properties for <b>slot2</b>.</p>	
--	--	---

The two air regions need to be replicated 11 more times, with a rotational displacement of 30 degrees.

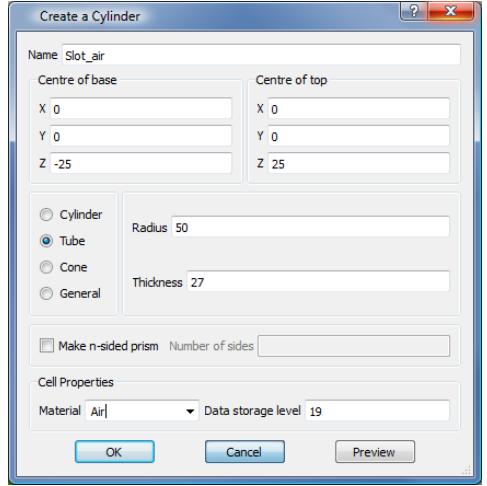
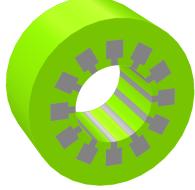
The air regions need to be picked, either by graphically selecting them or by using the  **Pick by Property** command and selecting by name. Once the two air regions have been selected, they can be copied using **Rotation**.

 <b>Copy and Transform</b>	<p>Make 11 copies of the picked bodies using the <b>Rotate</b> option. The rotation is done around the Z-axis with an angle of 30 degrees.</p> 	
--	--	---

The air regions need to be extracted from the stator core using the boolean operation **Subtraction with regularization**. The stator core is selected first and the air regions can then be selected as a

group by their name, in the  **Pick by Property** menu or by graphical picking.

The result of the subtraction is only the stator iron. The slots could be left to be filled later by the **Create Model Body** operation. However, in order to have a more precise control over the mesh size in these regions, it is better to fill the slots with air.

 <b>Create Cylinder</b>	<p>Define a tube called <b>Slot_air</b> with the same internal and external diameters as the <b>stator</b> (outer radius of 50, a thickness of 27 and a length of 50, material name, <b>Air</b> and storage level 19).</p> 	
 <b>Pick by Property</b>   <b>Combine Bodies</b>	<p>In the <b>Name</b> section select <b>slot_air</b> followed by <b>stator</b>. Close the dialog and select <b>Combine Bodies &gt; Trim Overlap with regularization</b>.</p>	

## Windings

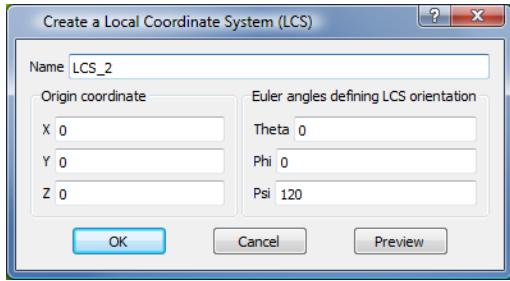
The windings chosen for this example are helical end conductors. These are created using local coordinate systems in order to simplify their positioning.

Since the PMSM has a total of 12 coil and 3 phases, three coordinate systems are needed. The symmetry of the conductors will be 90 degrees.



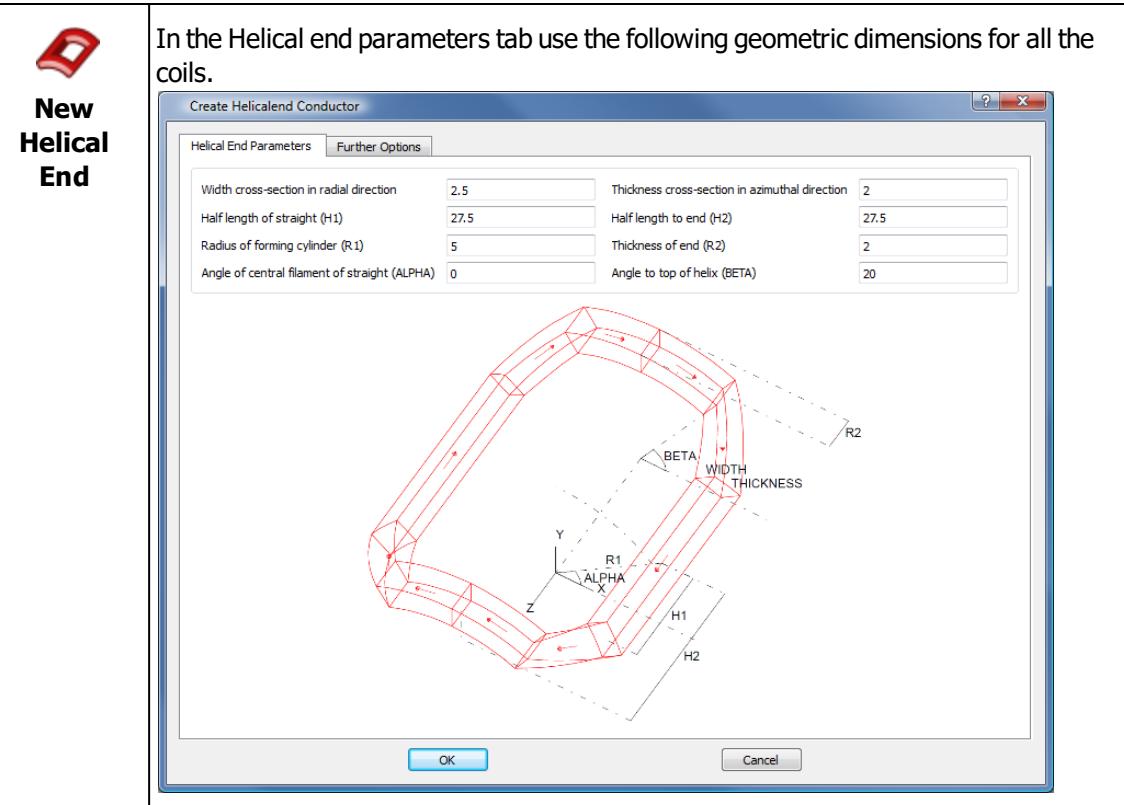
### Create a Local Coordinate System

The **Global Coordinate System** can be used for the first phase. Two LCS are created for phases 2 and 3. The first is rotated by 120 degrees and is named `LCS_2`:

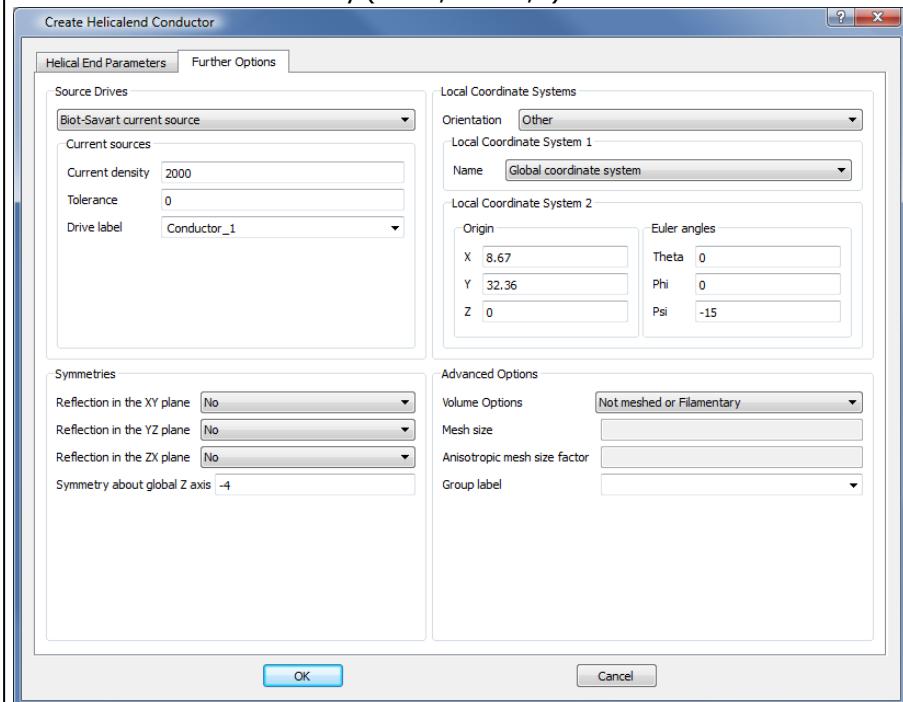


Create a second LCS with Psi equal to 240, named `LCS_3`.

Next, the helical windings are created using the same geometric dimensions but applying a different LCS for each set of coils.



The current density, coil symmetry and LCS are set in the Further Options tab. For the first set of coils, **Local Coordinate System 1** is set to the Global Coordinate System; the other two sets use **LCS\_2** and **LCS\_3**. All the conductors are offset by  $(8.67, 32.36, 0)$  with a **Psi** rotation of **-15**.

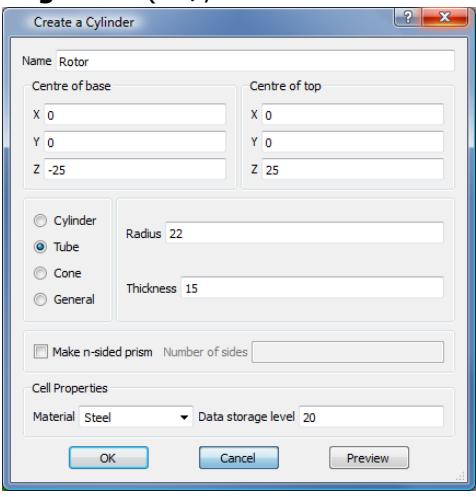
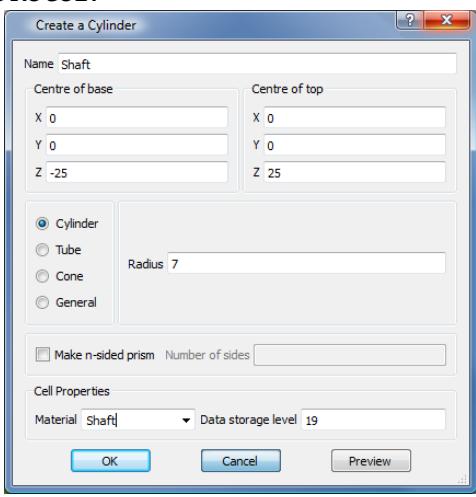
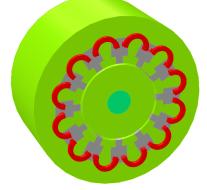


The drive label for the first set of conductors is **Conductor\_1** (the conductors set defined using the Global Coordinate System) while the other two have the drive labels **Conductor\_2** and **Conductor\_3**.

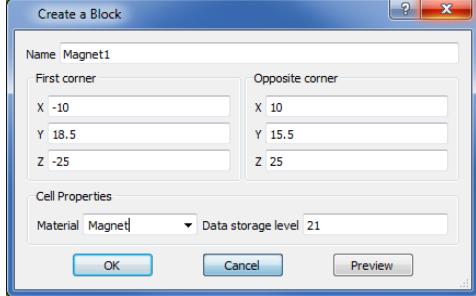
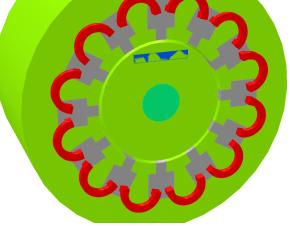
The current density is calculated based on a nominal current per phase of 100 A, with 100 turns in each conductor. The cross-sectional area of the Biot-Savart conductor is  $5 \text{ cm}^2$ . Set a current density of  $100 * 100 / \text{AREA}$  in **Conductor\_1** and 0 in **Conductor\_2** and **Conductor\_3**. The symmetry of the conductors around the Z-axis is -4.

## Rotor

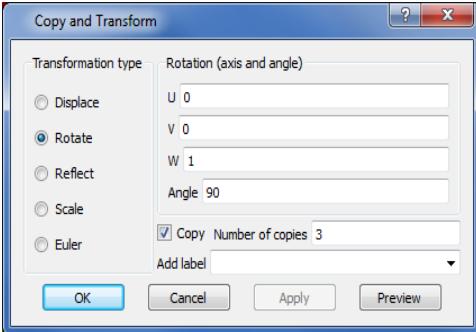
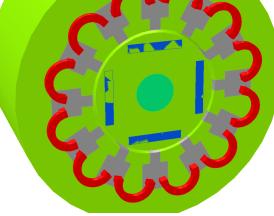
The rotor is built using a cylinder and 4 blocks representing the embedded permanent magnets. The insulating regions that act as a flux barrier between the magnets and help concentrate the magnetic flux in a radial direction are cut from the rotor cylinder using cut planes.

 <b>Create Cylinder</b>	<p>The <b>Rotor</b> is a tube with an outer <b>radius</b> of 22 and <b>thickness</b> of 15. Thus, the air gap thickness will be equal to 1. The rotor back-iron uses the same <b>material name</b> (<b>Steel</b>) and <b>storage level</b> (20), as the stator back-iron.</p>  <p>The <b>Shaft</b> is made of a cylinder with a <b>radius</b> of 7, <b>material name</b> <b>Shaft</b>, <b>storage level</b> 19 and the same base and top coordinates as the <b>Rotor</b>.</p> 	
---	--	---

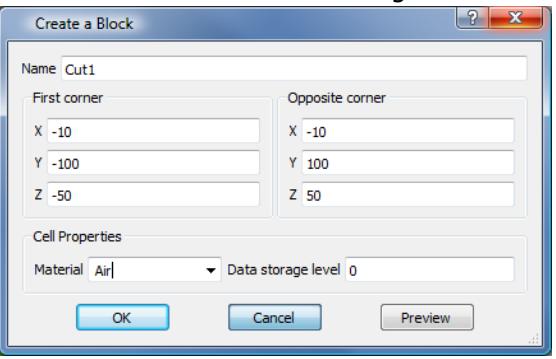
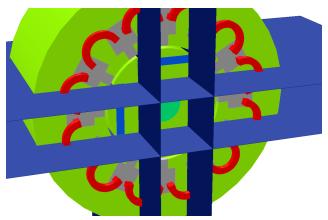
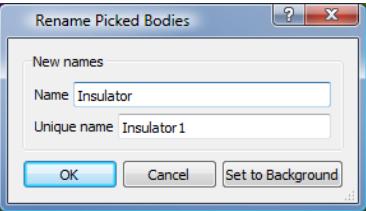
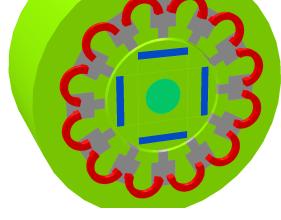
The magnets have a width of 3 and a length of 20. The centre point of the magnet is positioned at a depth of 5 relative to the rotor outer radius.

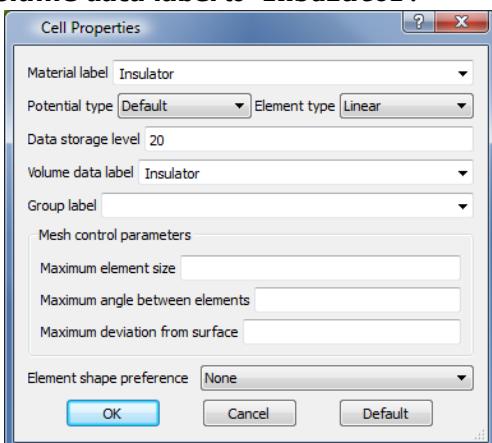
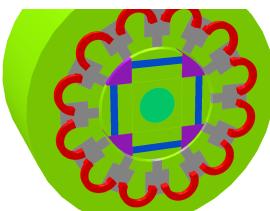
 <b>Create Block</b>	<p>The first magnet is created using the coordinates (-10,18 . 5,-25) and (10,15 . 5,25). The material is set to <b>Magnet</b> and the storage level to 21.</p> 	
--	---	---

Pick the magnet using the graphical interface or  **Pick by Property**.

 <b>Copy and Transform</b>	<p>The magnet is replicated 3 times with a rotational angle of 90 degrees.</p>  <p>Extract the magnets from the rotor iron by picking the rotor and then the magnets and selecting <b>Combine Bodies &gt; Trim Overlap with Regularization</b></p>	
--	--	--

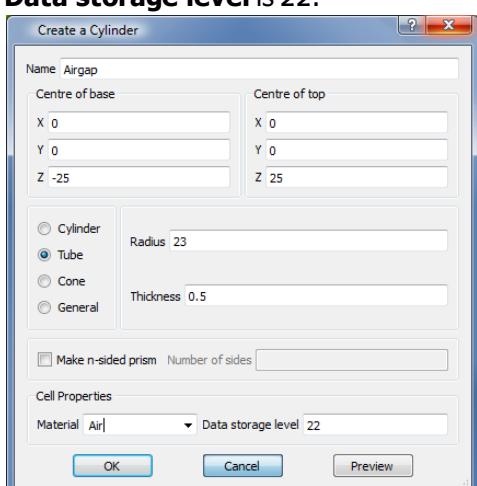
The air regions used to simulate the flux barriers between the magnets are created with the help of four cut planes.

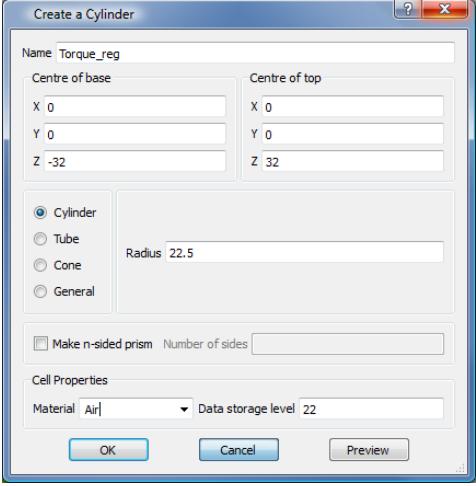
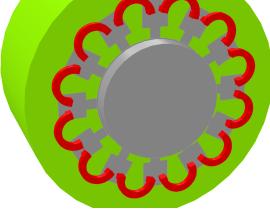
 <b>Create Block</b>	<p>The first cut plane (named <b>Cut1</b>) is created at <math>(-10, -100, -50)</math> and <math>(-10, 100, 50)</math>. The material is set to <b>Air</b> and the storage level to 0.</p>  <p>The other three cut planes have the following coordinates:  <b>Cut2</b> <math>(10, -100, -50)</math>; <math>(10, 100, 50)</math>  <b>Cut3</b> <math>(-100, -10, -50)</math>; <math>(100, -10, 50)</math>  <b>Cut4</b> <math>(-100, 10, -50)</math>; <math>(100, 10, 50)</math>.</p>	
 <b>Pick Bodies</b>	<p>Select the <b>Rotor</b>, followed by the four cut planes and perform a <b>Subtraction with Regularization</b>.</p> <p>Next, pick the <b>Rotor</b> body again, right-click and select  <b>Extract Cells</b> from the context menu.</p> <p>Select each of the four corner regions individually and choose  <b>Rename Body</b>.</p> <p>Give the same body <b>Name (Insulator)</b> to all of the four regions and use <b>Insulator 1-4</b> as their <b>Unique names</b>.</p> 	

 <b>Pick Cells</b>	<p>Select the four cells that make up the <b>Insulator</b> body and chose  <b>Cell Properties</b>. Modify the <b>Material label</b> and the <b>Volume data label</b> to <b>Insulator</b>.</p> 	
--	--	---

## Air gap

In order to improve the accuracy of the torque calculation, the air gap is divided into two regions: one tube next to the stator, named **Airgap** and one cylinder encasing the rotor, named **Torque\_reg**.

 <b>Create Cylinder</b>	<p>The <b>Airgap</b> is a tube with the centre coordinates <math>(0, 0, -25)</math> and <math>(0, 0, 25)</math>, a <b>Radius</b> of 23 and a <b>Thickness</b> of 0.5. The <b>Material</b> name is <b>Air</b> and the <b>Data storage level</b> is 22.</p> 
---	---

 <b>Create Cylinder</b>	<p>The air region used for the calculation of the torque in the static analysis is a cylinder with the following coordinates: <b>Centre of base</b> (0, 0, -32), <b>Centre of top</b> (0, 0, 32), <b>Radius</b> 22.5. The <b>Material label</b> is <b>Air</b> and the <b>Data storage level</b> is 22.</p> 	
---	---	---

The rotor, magnets, shaft and insulator regions need to be extracted from the torque calculation

region. Select  **Pick Bodies**, pick the **Torque\_reg** body, followed by **Rotor**, **Magnet1**, **Shaft** and **Insulator**. The easiest way of doing the selection is by using the **Pick by Name**

 **Pick Entities by Property** dialog. Next, use the **Combine Bodies > Trim Overlap with Regularization** to cut the torque calculation region around the rotor.

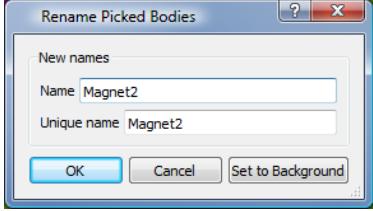
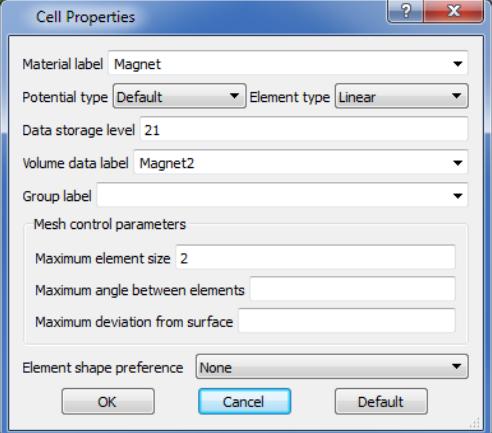
Finally, create an air region called **outside\_air** using the  **Create Cylinder** command. The centre coordinates are (0, 0, -50) and (0, 0, 50), while the **radius** of the cylinder is 75. The **material** for the background is **Air** and the **Data storage level** is set to 1. This completes the geometry.

## Material Properties

The material properties for the steel, magnets and insulating regions need to be specified. The magnetic characteristics of the steel and magnets are set using the material BH curves.

At the same time, the size of the mesh elements and volume orientation are set for the different model regions. Since the magnets have individual magnetization directions, each one of them has to have a unique volume label assigned to it.

In order to facilitate the next operations, select the  **Hide Entity** button and double-click on the  **Pick Entity** after the **Outside\_air** and **Torque\_reg** regions to hide them. Make sure to revert to  **Pick Entity** after the two bodies have been hidden.

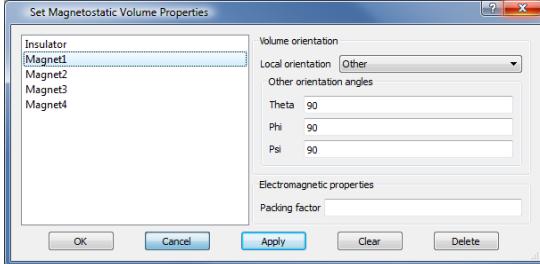
 <b>Pick Bodies</b>	<p>Pick the replicated magnets one at a time and rename them using the  <b>Rename Body</b> command. The magnets should be named <b>Magnet1</b>, <b>Magnet2</b>, <b>Magnet3</b> and <b>Magnet4</b> respectively, starting with the original one and going in a clockwise direction. Both the Name and the Unique Name should be updated in order to avoid errors in later body operations.</p> 
 <b>Pick Cells</b>	<p>Select each of the magnets as cells and add the volume label and material label information for each one of them.</p> <p><b>Material label:</b> all four magnets should have the <b>Magnet</b> material label.</p> <p><b>Volume data label</b> should be unique for each of the magnets (in this case it is the same as the body name), as this is used to orientate the magnetization direction.</p> <p><b>Maximum element size: 2.</b></p> 

The magnetization direction is set using the  **Volume Properties** command. In order for this menu to be available make sure that the **Magnetostatic** analysis is selected.



### Volume Properties

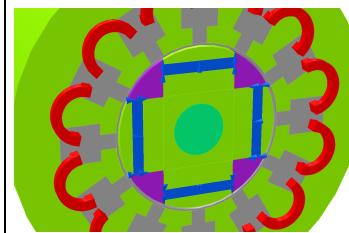
For the **Magnet1**, the magnetization vectors are **Theta=90 Phi=90 Psi=90**.



The orientations for the other three magnets are:

- Magnet2: 90, 180, 90**
- Magnet3: 90, 270, 90**
- Magnet4: 90, 360, 90**

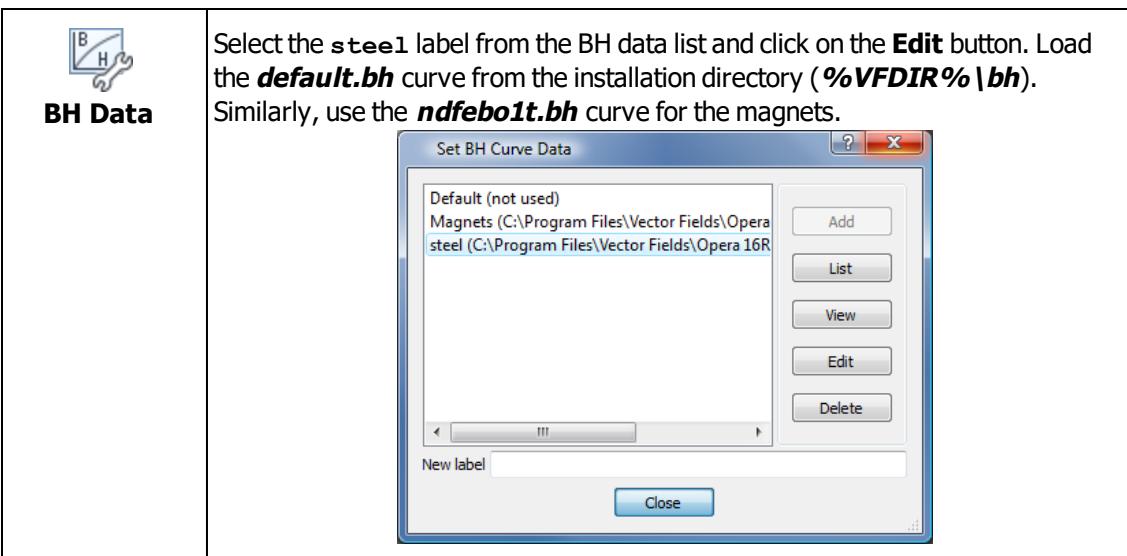
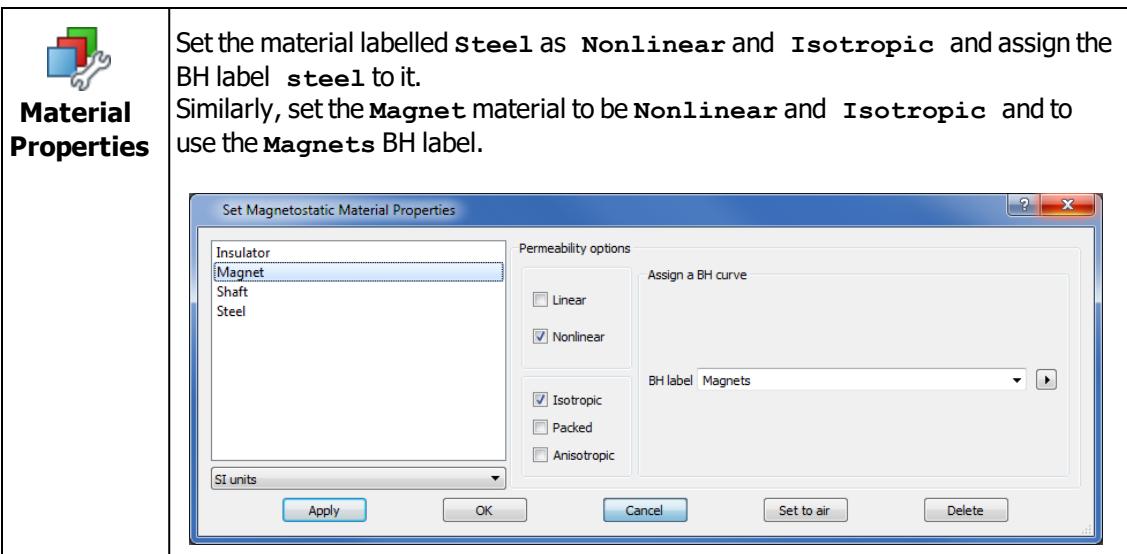
The magnetization direction of the volumes can be visually inspected using the  **Vectors** option and a scale factor of 3.



The mesh sizes, group labels and volume labels attributed to the different regions through the  **Cell Properties** dialog are:

Name	Mesh size	Group Label	Volume Label
<b>Stator</b>	2	<b>Stator</b>	
<b>Airgap</b>	1	<b>Gap</b>	
<b>Insulator</b>	-	<b>Rotor</b>	
<b>Torque_reg</b>	1	<b>Rotor</b>	
<b>Magnet1-4</b>	2	<b>Rotor</b>	<b>Magnet1-4</b>
<b>Rotor</b>	2	<b>Rotor</b>	
<b>Shaft</b>	15	<b>Rotor</b>	

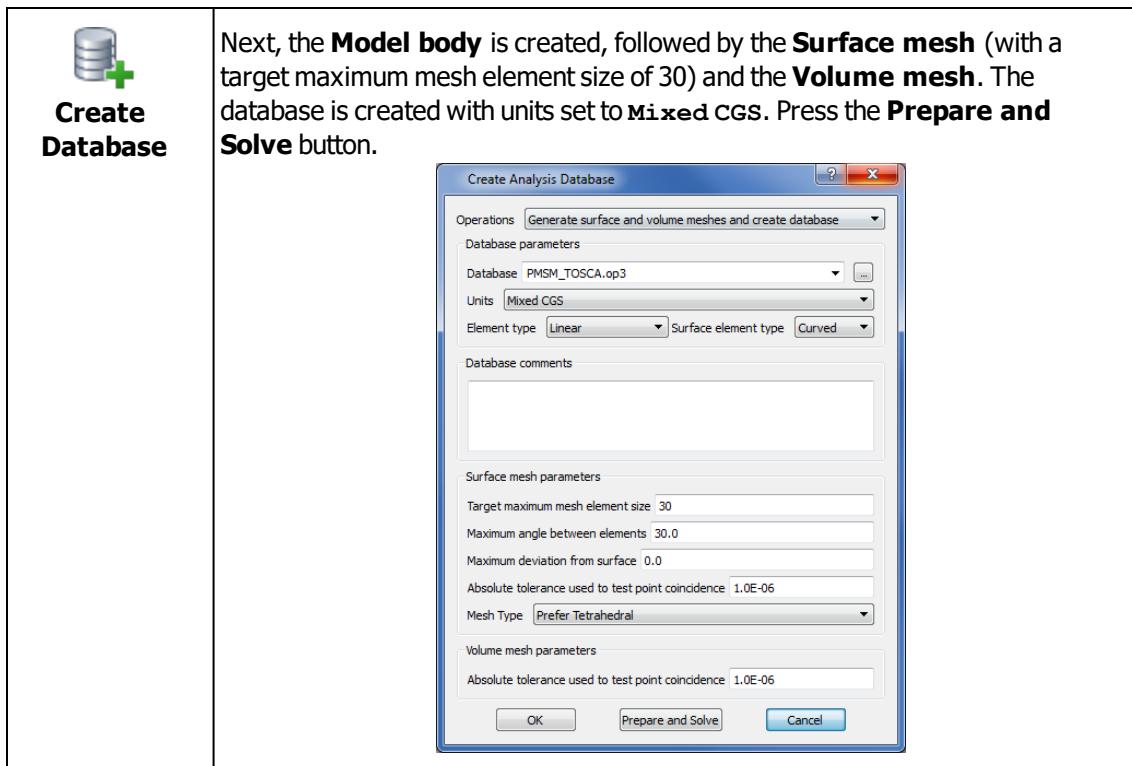
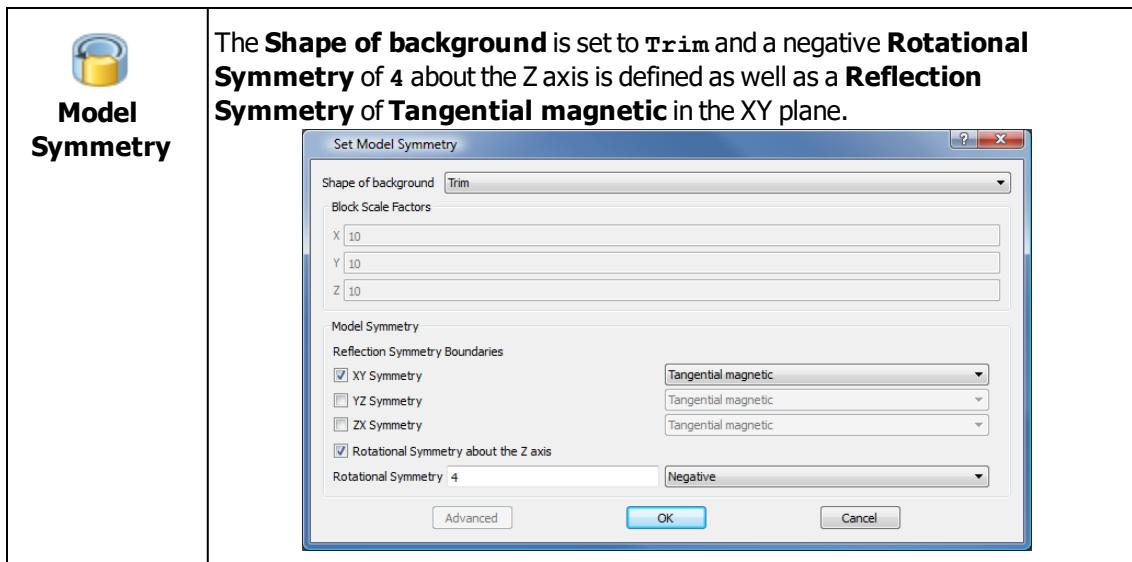
The nonlinear characteristic curves need to be assigned to material labels that already exist in the model. The back-iron for both the stator and the rotor uses the same material label and hence will have the same BH curve. The permanent magnets will make use of the same BH characteristic, only their volume orientation being different.



## Solver Settings

Open the **Analysis Settings** dialog, make sure the **Magnetostatic** analysis is selected and the **Material Options** are set to **Nonlinear materials**. Press the **Submit** button to apply the changes and close the dialog.

Next, the model symmetry is applied.



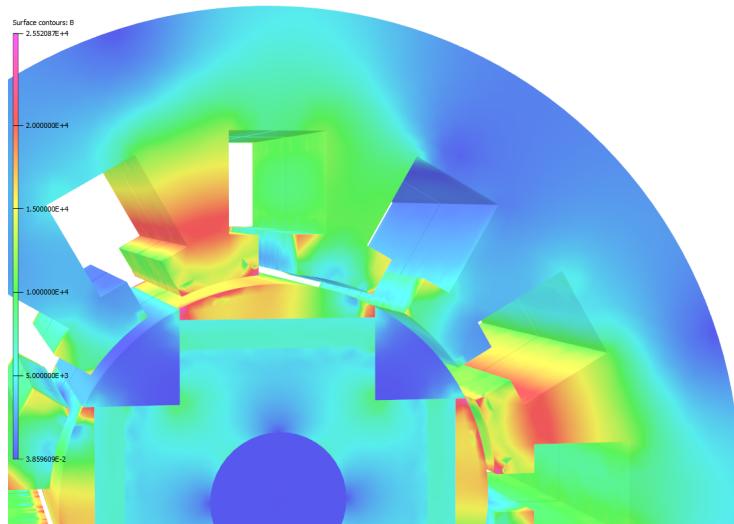
## Post-Processing

Once the model has been solved, launch the **Post-Processor** and load the solved database.

By default, only the solved symmetry section of the device is loaded. In order to have the full model visible, open the  Select dialog and change the **Show model symmetry** to **Full**.

## Static Field Distribution

The field distribution in the model and the flux vectors can be plotted using the  **3d Display** command. The flux density, in Gauss, is shown in [Figure 6.2](#) and the distribution of the flux density vectors is given in [Figure 6.3](#).



*Figure 6.2 Flux density in the PMSM*

The torque produced in the machine can be computed using an integral of the Z component of the Maxwell stress over a polar patch through the air gap.

The polar patch should cover the entire extent of the symmetry section of the machine, at the mid-

 **Polar Patch** and in the **Cylinders and Segments of Cylinders** tab create a complete cylinder with a radius of 22.5, from  $Z=-25$  to  $Z=25$ . Use 100 measuring points along the length and 720 on the arc and set the component for the map to MSTZ (the Z-component of the Maxwell stress). The result of the integral is reported in the console and below the scale of the map contours. The torque value is for the entire machine and in the model units (Ncm).

The polar patch is displayed in [Figure 6.4](#).

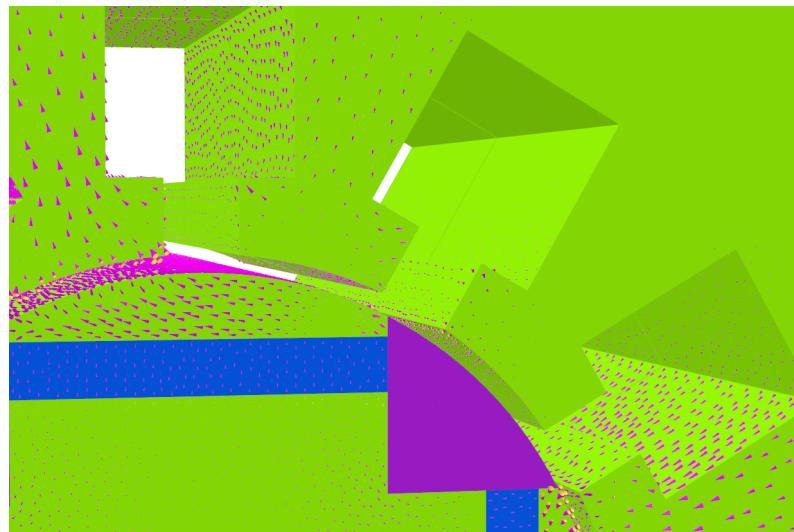


Figure 6.3 Flux density vectors in a section of the PMSM

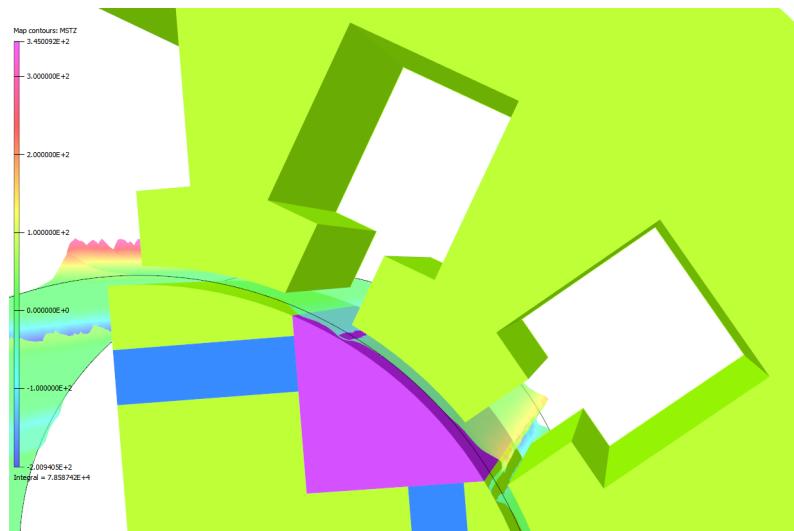


Figure 6.4 Polar plot of the Z-component of Maxwell stress in the air gap

## Magnetostatic model at different rotor positions

### Machine Characterization

A static analysis provides a snapshot of the magnetic fields in a machine at a particular rotor position. The characterization of electrical machines usually implies a number of measurements over a certain period. This can be simulated by solving a number of static cases at different rotor positions. Thus, the torque, back-EMF or cogging torque can be obtained with respect to the rotor position.

### Script for Modeller

The most efficient way of setting up a number of **Magnetostatic** models with different rotor/stator alignments is to use a script for the Opera-3d Modeller that features a loop.

This script loads the model that was built in the previous section of this chapter. The model body from the initial step needs to be deleted and the moving regions (**Rotor**, **Magnet1**, **Magnet2**,

**Magnet3**, **Magnet4** and **Insulator**) need to be picked and rotated using the  **Copy** command. The magnetization direction in the magnets also needs to be adapted accordingly with the rotation step. The currents in the conductors can be varied in order to simulate a sinusoidal wave form. However, in this example, the currents are kept constant over the entire simulation range.

Once the modifications have been made, the model body is created again. Then the model is meshed and solved. Note that since the mesh has changed due to the rotation of the rotor regions, the solution cannot be added to the previous database, hence each step will have to be a standalone database.

The command lines needed to implement the iterative method of creating stepped solutions are presented in the script **Modeller\_steps\_automation.comi**, which can be found in the examples folder.

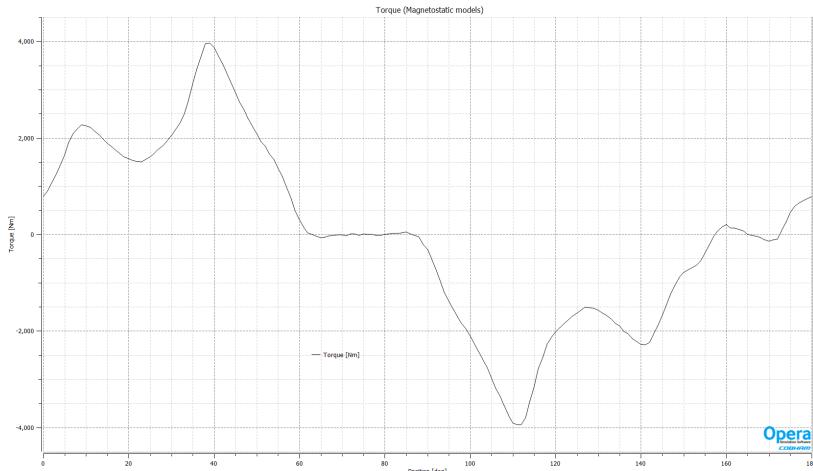
The number of steps and the period over which the calculations need to be done depend mainly on the machine type, topology and electrical parameters.

The kind of result which is aimed for should also be considered when doing step jobs. For steady-state torque calculations, a discretization step of one degree or even more, over a few electric cycles is often enough to obtain a valid result. However, for other types of calculations, where the field varies more rapidly, as is the case of cogging torque or back-EMF, a higher discretization is needed to capture the correct evaluation of the characteristics.

### Script for Post-Processor

The solutions need to be loaded one at a time in the **Post-Processor** and the fields evaluated in order to obtain the desired output. The code presented in the file **Post\_steps\_automation.comi** in the examples folder, uses a polar patch for computing the torque versus position from the Z-

component of the Maxwell stress. The patch is positioned at the mid-gap radius of the machine. The graph of torque versus position is shown in [Figure 6.5](#).



*Figure 6.5 Torque versus Position*

## Cogging Torque

The cogging torque and the back-EMF produced in the PMSM can also be evaluated using the method of "stepped jobs", i.e. many static solutions at different rotor/stator alignments.

In case of the cogging torque, the current density in all three phases needs to be set to 0 and in general a higher step discretization needs to be selected.

After the databases are solved, the same **comi** script can be used that loads the databases into the **Post-Processor** and retrieves the torque values in the air gap can be used to obtain the final results. The cogging torque, over a period of 45 mechanical degrees, is presented in [Figure 6.6](#). Note that depending on the polarity of the magnets and the direction of rotation, the cogging torque polarity might be reversed.

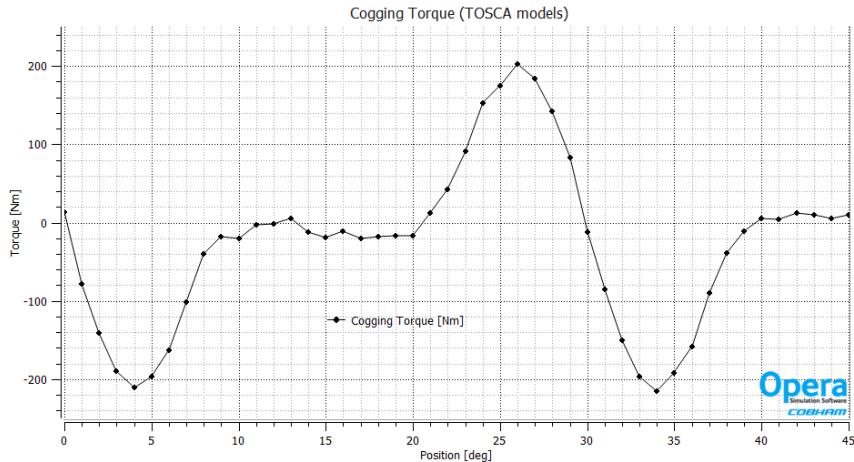


Figure 6.6 Cogging torque in the PMSM

## Back-EMF

For the back-EMF calculations, the first step is to calculate flux linking the coil. The flux linkage is given by the integral of the normal component of the magnetic field on a patch going through the middle of the coil. In this case, the radial component of flux density ( $B_r$ ) is normal to the patch going through the middle of the coil and hence can be used for calculating the flux linkage. The command for obtaining the value of the integral of  $B_r$  using a polar patch is:

```
POLAR R1=0.34 R2=R1 R3=R1 R4=R1 T1=3 T2=T1 T3=27 T4=T3 Z1=-0.25 Z2=0.25 Z3=Z2 Z4=Z1 N1=40
N2=80 Buffer=Polar | MAP FILE=TEMP COMPONENT=Br
```

Once the flux linkage is written to the text file, a new line representing the back-emf can be plotted using the **Interpolation type: Derivatives** option. The Y-component of the line will be **flux linkage\*2\*PI\*f\*rpm\*360/60**, where **flux linkage** is the column in the file containing the values of flux linkage, **f** is the frequency (50 Hz in this case) and **rpm** is the rotor speed (1500).

Figure 6.7 shows an example.

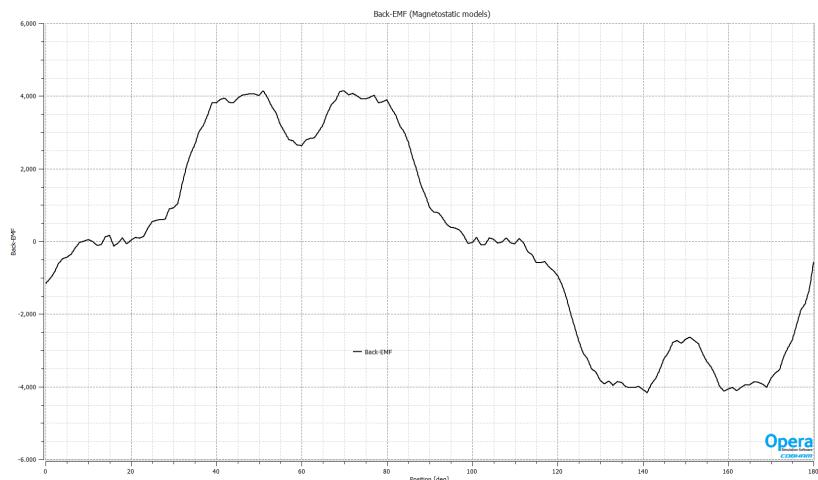


Figure 6.7 Back-EMF in the PMSM

## Transient motion model

---

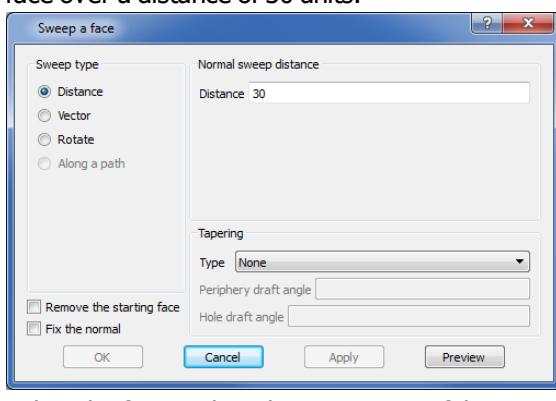
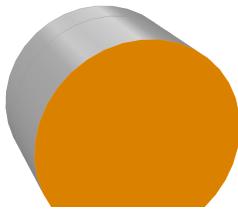
The model of the PMSM built previously can be modified for use in a dynamic simulation with the **Transient Motion** solver. The modifications that need to be done to the model definition are outlined in the following.

Note that the resulting models are larger than the Magnetostatic solution and it might take several hours to analyse. The time can be reduced by requesting a smaller final output time or by selecting linear material properties.

### Model update

#### Modifying regions

Load the static model created previously and delete the model body if it exists.  
The first step is to extend the outer air region in axial direction in order to better include end effects.

 <b>Pick Faces</b>	<p>The two external faces of the outer air should have the normal direction pointing outwards from the background region. This can be checked using the  <b>Vectors</b> command and a scale factor of 2 for the Face normal (picked faces only) option.</p> <p>If the vectors are pointing towards the inside of the air region, double click them to change their orientation.</p> <p>Pick the external face of the background region at z=50 and select  <b>Sweep Face</b>. Sweep the face over a distance of 30 units.</p>  <p>Select the face at the other extremity of the background region at z=-50 and do a sweep with a distance of 30.</p>	
--	--	---

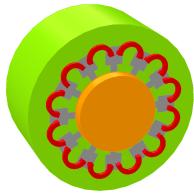
All regions in a **Transient Motion** model need to be included in one of three groups: STATIC, MOVING or GAP. Regions in the STATIC group are not allowed to touch regions in the MOVING group. Thus, a GAP group is needed to separate the two groups.

Several configurations for a GAP region are discussed in [Machine Analysis using Motional EM solver \[page 684\]](#). For this model, Configuration A will be used.

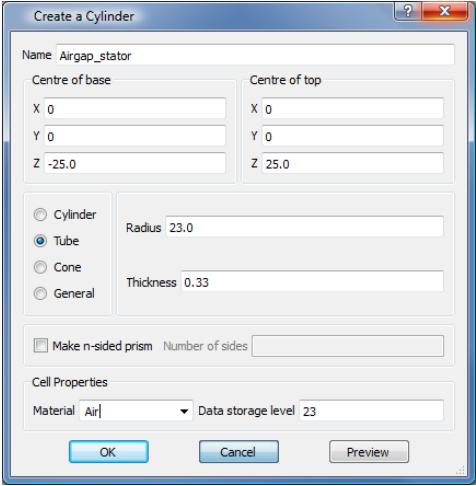
In this example three layers of air are created in the air gap in order to have a better representation of the flux paths going between the stator and the rotor. The GAP group consists only of the middle layer of air, the two other layers being a part of the STATIC and MOVING groups, respectively.

The background region can be hidden from view at this point, in order to facilitate the work on the model. This can be done by deselecting the `outside_air` region in the  **Select** menu or by moving the mouse above the region and hitting the `H` key on the keyboard.

The two air regions created in the gap in the static model need to be deleted and three new air regions are created in their place.

 <b>Pick Bodies</b>	Select the two air regions that make up the air gap ( <b>Airgap</b> and <b>Torque_reg</b> ). An alternative way of picking them is by using their name in the  <b>Pick by Property</b> dialog.  <b>Delete</b> the regions.	
---	--	---

The new air gap will be made from three regions of equal thickness.

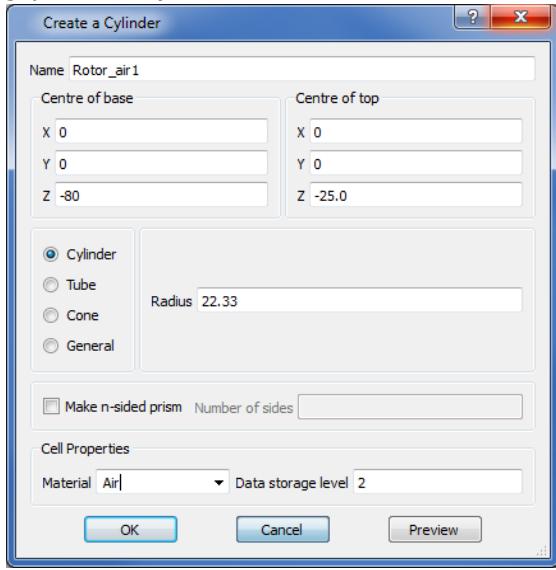
 <b>Create Cylinder</b>	<p>The air region closest to the stator is named <b>Airgap_stator</b>, and has the following dimensions: <b>Centre of Base</b> (0 ; 0 ; -25), <b>Centre of Top</b> (0 ; 0 ; 25), <b>Radius</b> 23 and <b>Thickness</b> 0 . 33. The <b>Material</b> is set to <b>Air</b> and the <b>Storage Level</b> to 23. This cell will be a part of the STATIC group.</p>  <p>The next layer of air, which will be included in the GAP group, is named <b>Airgap</b>. This cell has to extend to the outer boundaries of the model in the Z direction. The coordinates for the <b>Centre of Base</b> are (0 ; 0 ; -80) and those for the <b>Centre of Top</b> are (0 ; 0 ; 80). The <b>Radius</b> is 22 . 67 and the <b>Thickness</b> 0 . 34. Again, the <b>Material</b> is set to <b>Air</b> and the <b>Storage Level</b> to 23.</p> <p>Finally, the air region closest to the rotor is labelled <b>Airgap_rotor</b> and has the following dimensions: <b>Centre of Base</b> (0 ; 0 ; -25), <b>Centre of Top</b> (0 ; 0 ; 25), <b>Radius</b> 22 . 33 and <b>Thickness</b> 0 . 33. The same <b>Material</b> and <b>Storage Level</b> as previously are used. This cell will be a part of the MOVING group.</p>	
---	--	---

The rotor also has to extend to the outer boundaries of the model, hence two new air regions are created. The two regions will be included in the MOVING group.

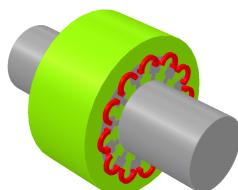


**Create Cylinder**

The two cylinders are named **Rotor\_air1** and **Rotor\_air2** and both have a **Radius** of **22.33** (external radius of the rotor plus 1/3 of gap thickness).



The **Centre coordinates** for the first region are  $(0 ; 0 ; -80)$  and  $(0 ; 0 ; -25)$  while the second region starts at  $(0 ; 0 ; 25)$  and ends at  $(0 ; 0 ; 80)$ .  
The **Material** used is **Air** and the **Storage level** is 2.



opera

## Cell data

The properties of the cells need to be updated in order to include the group labels and new mesh sizes needed for the dynamic simulation.

Name	Mesh size	Group Label
Outside_air	-	STATIC
Stator	2	STATIC
Airgap_stator	1	STATIC
Slot_air	2.5	STATIC
Airgap	2	GAP
Airgap_rotor	1	MOVING
Rotor_air1	-	MOVING

<b>Rotor_air2</b>	-	<b>MOVING</b>
<b>Rotor</b>	2	<b>MOVING</b>
<b>Magnet1-4</b>	2	<b>MOVING</b>
<b>Insulator</b>	2	<b>MOVING</b>
<b>Shaft</b>	15	<b>MOVING</b>

## Analysis settings

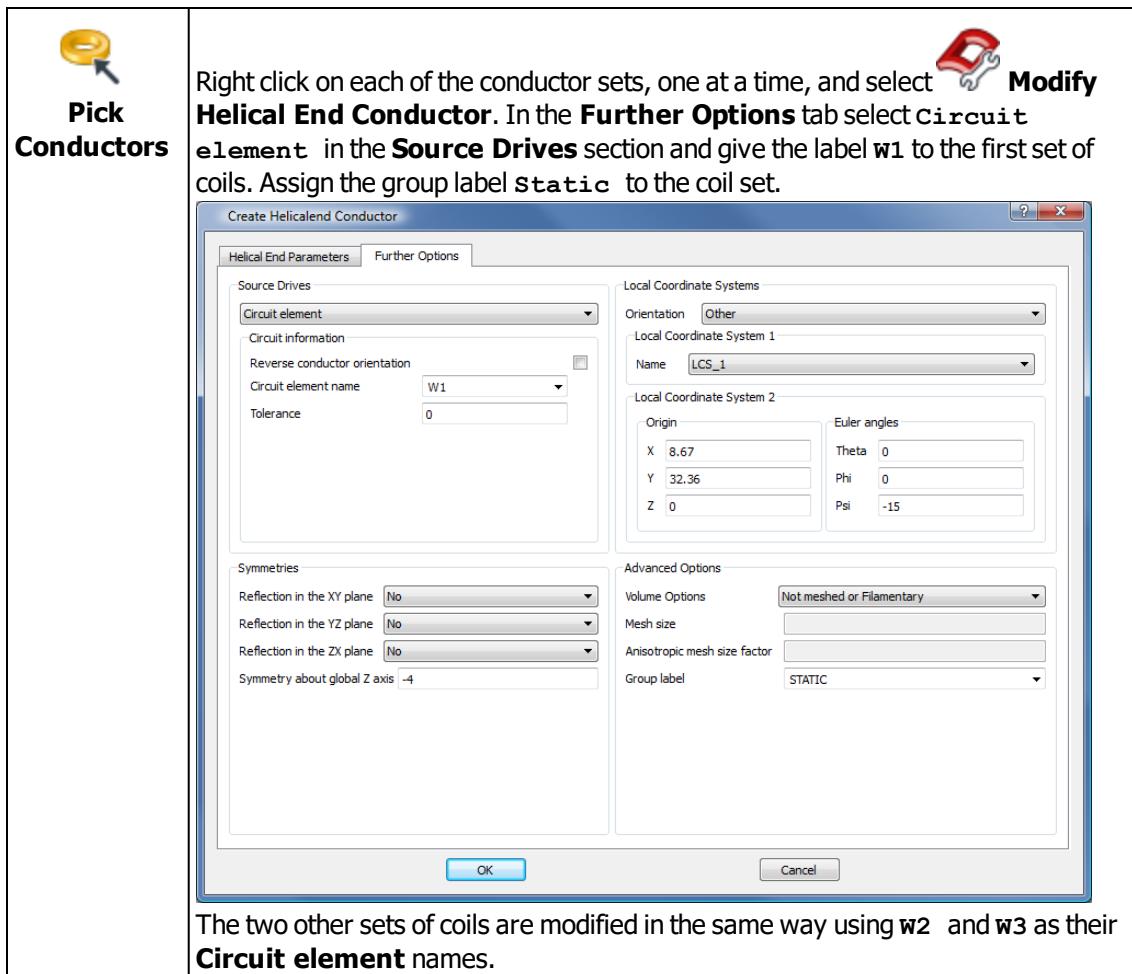
The analysis type needs to be changed to **Motional EM** and the analysis properties and output times need to be set. In this example the model is setup for an open circuit analysis at 1500 rpm. Hence, the range of movement of the rotor is limited to 45 degrees and a small-enough timestep is used.

Set the following properties in the Configuration pane of the **Motional EM** analysis.

Property	Value	Description
<b>Material options</b>	Non-linear materials	
<b>Fixed time-step</b>	2.2E-05	
<b>Output times</b>	0;5E-03;2.2E-05	
<b>Logging variables</b>	TTIME	Simulation time
	MO_ThetaZ	Angle of moving part (rad)
	MO_TorqueZ	Torque of moving part
	W1_V	Phase 1 voltage
	W2_V	Phase 2 voltage
	W3_V	Phase 3 voltage
	W1_I	Phase 1 current
	W2_I	Phase 2 current
	W3_I	Phase 3 current

## Circuit

An electrical circuit is used to drive the **Motional EM** model, therefore the properties of the conductors need to be changed. The conductors also need to be included in the STATIC group.



The circuit defined for this model is presented in [Figure 6.8](#). It consists of three independent current loops. Each of the three windings (**w1**, **w2** and **w3**), representing the three phases, has 100 turns. The windings will use the model symmetry. The total phase resistance of  $4\Omega$  is set using the **R1**, **R2** and **R3** resistors. The open circuit state is simulated with the help of three resistors (**R\_oc1**, **R\_oc2** and **R\_oc3**) that have a very high resistance ( $1 M\Omega$ ).

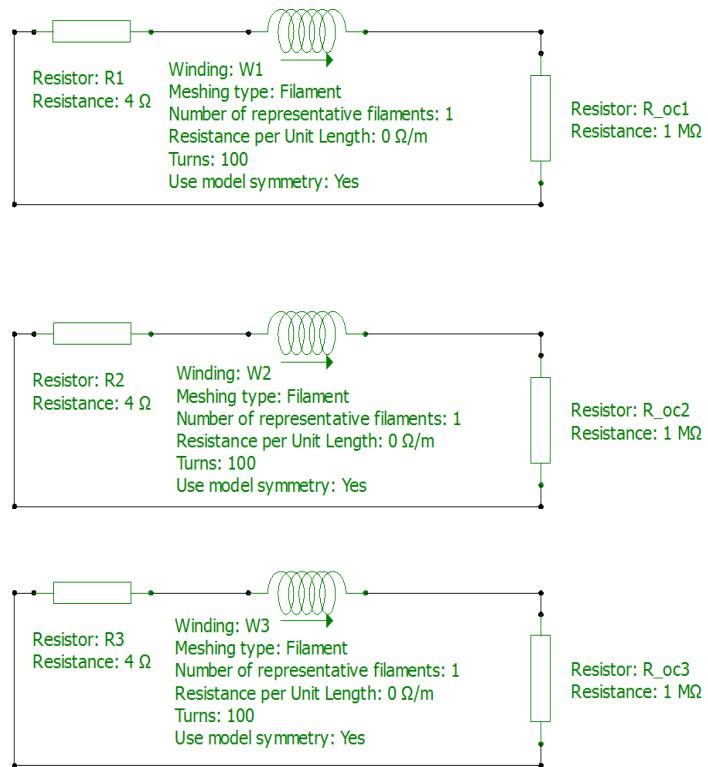


Figure 6.8 Circuit for PMSM Transient Motion model

## Total Potential

In the Magnetostatic model no potential type was set explicitly. In this case the software sets the potential type to **Default**. This is fine for the Magnetostatic model, because **POTENTIAL=DEFAULT** will give a **REDUCED** potential for cells with **MATERIALLABEL=AIR**, and **TOTAL** potential for all other cells. This way, the general rule that Biot-Savart conductors must be embedded in **REDUCED** potential is obeyed.

In the previous step the conductors have been changed from the Biot-Savart conductor type to the Filamentary conductor type. Now, the potential surrounding the conductors needs to be changed to **TOTAL** potential. The magnetic materials in the model also have to be set to **TOTAL** potential. The potential can be assigned to all cells in the model using 1 simple step:

 <b>Pick Cells</b>	To pick all cells of the model, use  or the keyboard shortcut <b>CTRL+A</b> . Then go to <b>Cell Properties</b> , and set the potential type to <b>Total</b> .	
--	---	--

## Motion control

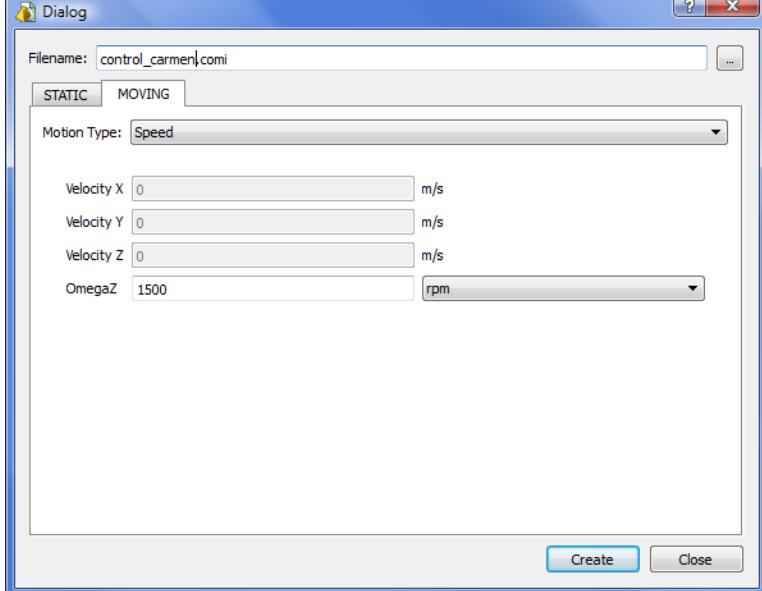
The motion control can be defined by the user in the Command File Editor. A menu exists to facilitate the definition of the movement equations.

The user can select from four types of dynamic behaviour for the groups defined earlier:

**Stationary, Position, Speed, Acceleration or Mechanical Coupling.** The parameters for the specified type of movement are automatically translated into motion equations and saved to the ***control\_carmen.comi*** file.

A simple, fixed speed control is chosen for this motional analysis. The nominal speed of the PMSM is

set to 1500 rpm. The following steps are done in the  **Command File Editor**

<b>File &gt; New Carmen control file.</b>	<p>Select the MOVING group and enter a speed of 1500 rpm. If needed, modify the path for the control file and select <b>Create</b>.</p> 
---	---

The generated ***control\_carmen.comi*** file contains the following lines:

```
/ Control commands for the group MOVING
$STRING MO_MOTIONCONTROL SPEED
$CONSTANT #MO_OMEGAZ 1500*2*PI/60

/ Control commands for the group STATIC
$STRING ST_MOTIONCONTROL STATIONARY
```

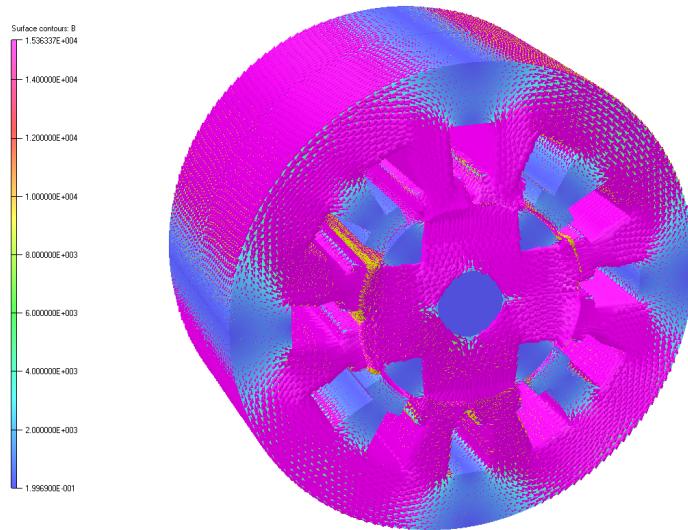
## Analysis

Use the same model symmetry as that of the **Magnetostatic** model and create the model body. Mesh the model using a **Surface mesh size** of 30. Create the Analysis database using **Mixed** CGS units, save and solve the model.

## Post-processing

After the model has been successfully solved, launch the **Post-Processor** and load the **op3** file.

The distribution of the flux in the machine is presented in [Figure 6.9](#).



*Figure 6.9 Flux distribution and flux density*

The cogging torque from the open circuit analysis is shown in [Figure 6.10](#).

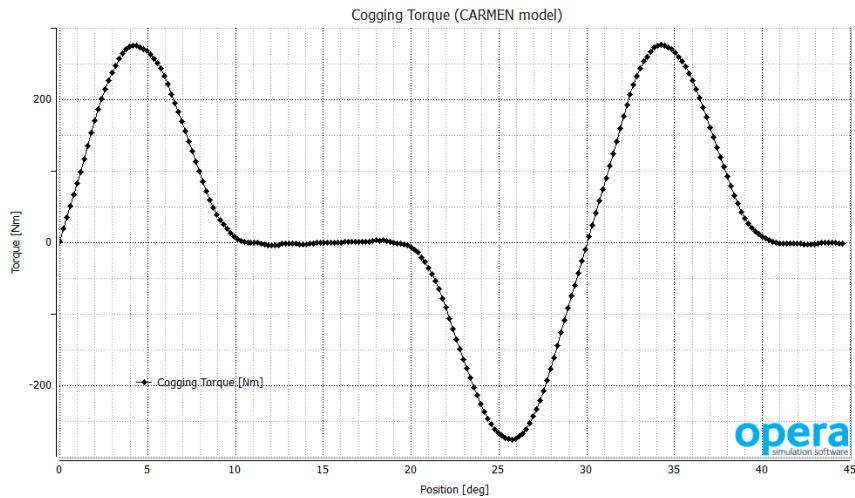


Figure 6.10 Cogging torque from the Transient Motion analysis

For the back emf characteristic the open circuit analysis needs to be run for more steps. The resulting curves from a simulation run to 180 degree (or 2E-02 seconds) is shown in the Figure 6.11.

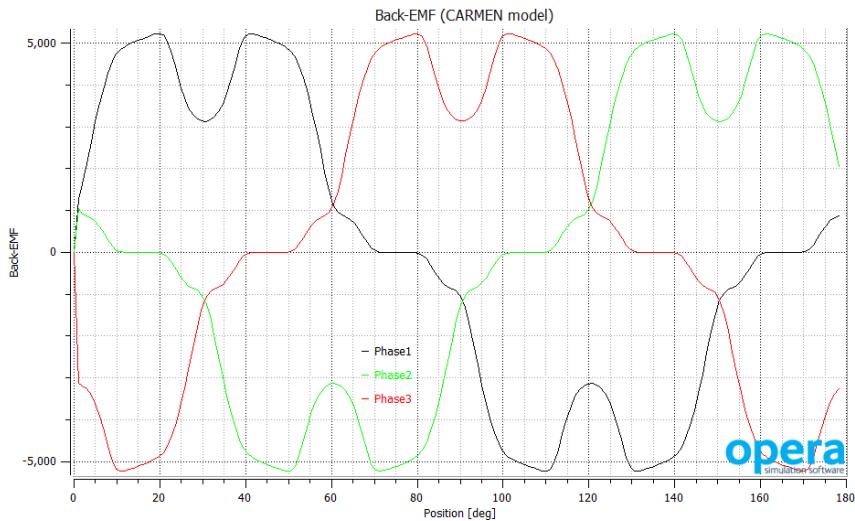


Figure 6.11 Back EMF from the Transient Motion analysis

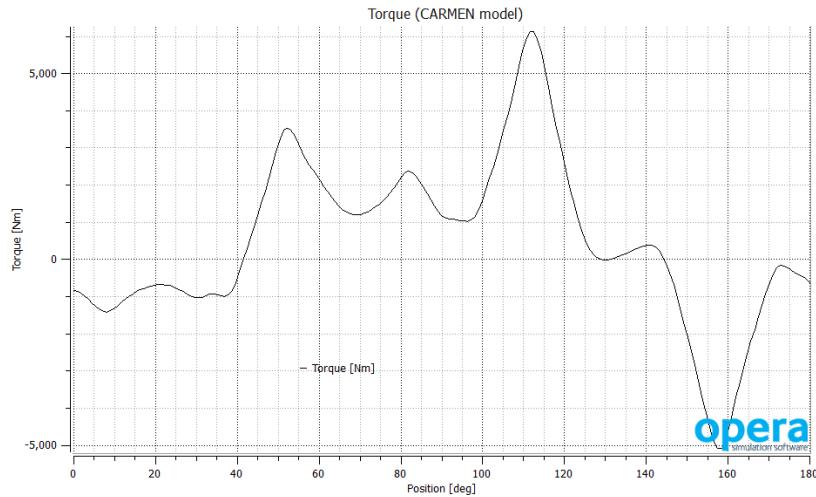
The difference between the results obtained from the static model and the ones from the dynamic model are due to the finer step size and a higher quality mesh in the air gap.

## DC torque

In order to look at the torque produced by a DC excitation in one of the phases, the circuit needs to be modified to include the voltage source(s) and to remove the open circuit resistors.

A coarser step is used (**1E-04**) and the simulation time is extended to **2E-02** so that it covers 180 degrees of movement.

The torque presented in [Figure 6.12](#) is obtained when driving the conductor **w\_1** from a DC source of **400V**.



*Figure 6.12 Torque from the Transient Motion analysis (DC excitation)*

# **Chapter 7**

## **Winding Tool**

### **Introduction**

---

The Winding Tool is a toolbox dedicated for electrical machine design within Opera FEA software. The main purpose of this tool is to help users assess feasibility and optimality of different winding configurations.

The Winding Tool will provide users with the following output<sup>1</sup>:

- Optimal winding layout (circular and linear representation)
- Star of Slots representation
- Winding factor harmonics
- Winding mmf harmonics
- Görge's diagram

The winding tool can be launched using the toolbutton  or by using the **WINDING** command.

---

<sup>1</sup>The output from the winding tool will be available as an Opera Python Object. For more information please refer to the **WINDING** command in the Reference Manual.

## Definitions and Terminology

---

Following is a list of symbols and their definitions<sup>1</sup> that will be referred to in the tool:

### Slots and coils per pole per phase

The slots per pole per phase ( $q$ ) is defined as:

$$q = \frac{Q_s}{m^*2^*p} = \frac{qn}{qd} \quad (7.1)$$

Where  $Q_s$  is the number of stator slots,  $p$  is the number of pole pairs,  $m$  is the number of phases. The reduced fraction form of  $q$  is represented by the fraction  $qn/qd$ .

The number of coils per pole per phase is defined as:

$$qc = \frac{Q_c}{m^*2^*p} = \frac{qcn}{qcd} \quad (7.2)$$

Where  $Q_c$  is the number of coils, and the fraction  $qcn/qcd$  represents the reduced form of  $qc$ .

### Average coil pitch

The coil pitch (also called the coil span) is defined as the peripheral angle between the two coil sides. It is practical to express the coil pitch in terms of the number of slots. Therefore, it is an integer number. However, the average coil pitch can be a real number and is defined as:

$$yp = \frac{Q_s}{2^*p} \quad (7.3)$$

The actual (or practical) coil pitch must be an integer. Therefore, it is defined as

$$yd = \max((\text{int}(yp) \pm k), 1) \quad (7.4)$$

Where  $k$  is an integer value, which users can add or subtract from the integer close to the average coil pitch. A negative  $k$  value gives a short pitched arrangement; a positive  $k$  value gives an over pitched arrangement.

### Basic winding

The smallest repetitive segment is called the basic winding. Due to symmetry only the basic winding needs to be determined. If  $qd$  is less than  $p$  the winding is composed of  $t$  identical basic windings, i.e.

$$t = \gcd(Q_c, p) \quad (7.5)$$

---

<sup>1</sup>J. Germishuizen and M. Kamper, "Classification of Symmetrical Non-Overlapping Three-Phase Windings", XIX International conference on Electrical Machines – ICEM 2010, Rome.

Where  $Q_c$  is equal to  $Q_s$  for double layer windings and  $Q_c$  is equal to half of  $Q_s$  for single layer windings. The  $gcd$  in the equation refers to the greatest common divisor.

## Winding feasibility

For a winding to be balanced the number of coils used in each of the phases must be equal. So for a winding to be feasible the following criterion needs to be satisfied:

$$\text{mod}\left(\frac{Q_c}{\text{GCD}(Q_c, 2^*p)}, m\right) = 0 \quad (7.6)$$

## Winding Classification

---

The following winding parameters are chosen for the classification of windings:

- the reduced form of the number of coils per pole per phase;
- the average coil pitch; and
- the number of layers.

Using these parameters the following definitions are associated with windings:

### Overlapping and non-overlapping:

In overlapping windings the coil end-windings overlap and the coil pitch ( $y_d$ ) is greater than one. If the coil pitch ( $y_d$ ) equals one then the coil end-windings do not overlap. An example of an overlapping winding is where the "Go" side of a coil is in slot 1 and the "Return" side is in slot 6, and the next coil starts in slot 2 and returns in slot 7.

### Single and double layer:

These windings are differentiated by the number of coils ( $Q_c$ ) compared to the number of stator slots ( $Q_s$ ). In single layer windings the number of coils ( $Q_c$ ) equals half the number of stator slots ( $Q_s$ ), while for double layer windings the number of coils is equal to the number of stator slots. Physically, in double layer windings, it means that the "Go" side of one coil and the "Return" side of another coil occupy the same slot. Please note that only single and double layer windings are considered where all slots are fully occupied (no empty slots).

### Full-pitch and fractional pitch:

If the coil pitch ( $y_d$ ) equals the average coil pitch ( $y_p$ ) it is called a full-pitch winding, and implies  $k=0$  in equation (7.4). If the coil pitch is not equal to the average coil pitch then the winding is charded and is called a fractional pitch winding. In the special case where the number of slots per pole per phase ( $q$ ) equals one, the winding configuration is called full-pitch. Concentrated windings are described in more detail later in this section.

### Integral and fractional slot:

If the denominator ( $qd$ ) of the slots per pole per phase ( $q$ ) equals one then it means that  $q$  is an integer and the configuration is called an integral slot winding. If  $qd$  is greater than one, it means that  $q$  is non-integer and is termed a fractional slot winding. In addition, the average coil pitch ( $y_p$ ) is also non-integer and hence, a fraction.

## Distributed and concentrated windings:

If the coils per pole per phase ( $qc$ ) is greater than 0.5 then the winding is distributed. If  $qc$  is less than or equal to 0.5, the winding is concentrated.

Now that we have defined various winding types based on the classical winding properties, the following flow chart will show the choice of the winding type based on these properties:

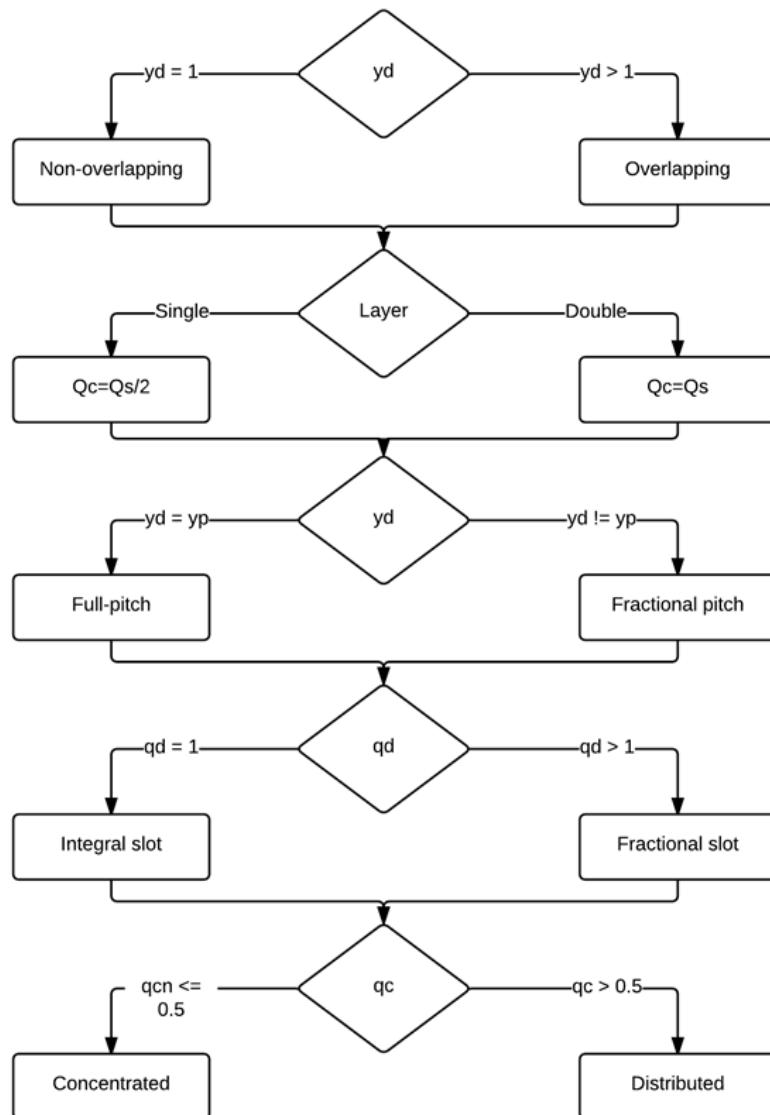


Figure 7.1 Winding classification scheme

## Winding Harmonics

---

### Winding Factor ( $k_w$ )

The winding factor for a specific winding describes the ratio of flux linked by that winding compared to the flux that would have been linked by a single-layer full-pitch non-skewed integer-slot winding with the same number of turns and one single slot per pole per phase. The winding factor is proportional to the electromagnetic torque, i.e. an electrical machine with a low winding factor needs to compensate its lower torque with higher current or larger number of turns, which are both inversely proportional to the winding factor.

The winding factor is defined as the product of the pitch, distribution and skew factors which are calculated individually. The values for each of these will be dependent on which spatial harmonic order in the flux waveform is being considered.

$$k_w = k_p \times k_d \times k_s \quad (7.7)$$

Where  $k_p$  is the pitch factor,  $k_d$  is the distribution factor and  $k_s$  is the skew factor.

### Pitch Factor ( $k_p$ )

The pitch factor<sup>1</sup> (sometimes also called coil-span factor or chording factor) reflects the fact that windings are often not fully pitched, i.e. their turns are reduced in length and do not cover a full pole-pitch (fewer slots between go and return than would cover a full pole-pitch).

$$k_p(n) = \sin\left(n \cdot \frac{\pi}{2} \cdot \frac{y_d \cdot 2}{Q_s}\right) \quad (7.8)$$

Where  $n$  is the harmonic order,  $y_d$  is the coil pitch.

### Distribution Factor ( $k_d$ )

The distribution factor<sup>2</sup> (sometimes also called breadth coefficient) reflects the fact that the winding coils of each phase may be distributed in a number of slots.

---

<sup>1</sup>F. Magnussen and C. Sadarangani, "Winding factor and Joule losses of permanent magnet machines with concentrated windings".

<sup>2</sup>N. Bianchi and M. Pre, "Use of Star of Slots in designing fractional-slot single-layer synchronous motors".

$$\begin{aligned}
 kd(n) &= \frac{\sin\left(\frac{n}{p} \cdot \frac{qph}{2} \cdot \frac{\alpha ph}{2}\right)}{\frac{qph}{2} \cdot \sin\left(\frac{n}{p} \cdot \frac{\alpha ph}{2}\right)} \dots \text{if } qph \text{ is even} \\
 kd(n) &= \frac{\sin\left(\frac{n}{p} \cdot qph \cdot \frac{\alpha ph}{4}\right)}{qph \cdot \sin\left(\frac{n}{p} \cdot \frac{\alpha ph}{4}\right)} \dots \text{if } qph \text{ is odd}
 \end{aligned} \tag{7.9}$$

Where  $\alpha ph$  is the angle between two adjacent spokes and  $qph$  is the number of spokes per phase, for more information See "Star of Slots" on page 264.

Note that the value of  $kd$  is explicitly 1 if the winding is concentrated.

## Skew Factor ( $ks$ )

The skew factor<sup>1</sup> reflects the fact that a winding can be angularly twisted from one end of the machine to the other (often in order to eliminate the cogging torque), which results in an angular spread of the conductors and hence a reduced winding factor.

$$ks(n) = \frac{\sin\left(n \cdot \frac{\gamma}{2}\right)}{n \cdot \frac{\gamma}{2}} \tag{7.10}$$

Where  $\gamma$  is the Skew Angle, defined as a mechanical angle in radians. Note that the value of skew angle<sup>2</sup> will lie between 0 and  $2\pi/Q_s$ . Also, note that the value of  $ks$  is explicitly 1 if there is no skew defined.

## Winding MMF

The winding mmf<sup>3</sup> for a single phase is given by

$$F(n) = \frac{\sqrt{2} \cdot m \cdot N \cdot kw(n)}{\pi \cdot n} \cdot I \tag{7.11}$$

Where  $N$  is the number of turns in each phase,  $I$  is the rms value of the phase current and  $n$  is the harmonic order.

## Harmonic Order

The harmonic orders that will be observed in a machine are determined using the following formulae evaluated for  $g=1, 2, 3, 4, \dots$  while  $n$  is less than a specified maximum harmonic:

<sup>1</sup>F. Magnussen and C. Sadarangani, "Winding factor and Joule losses of permanent magnet machines with concentrated windings".

<sup>2</sup>M. Jagiela, E. Mendrela, P. Gottipati, "Investigation on a choice of stator slot skew angle in brushless PM machines".

<sup>3</sup>G. Joksimovic, "AC Winding Analysis using Winding Function Approach".

- Single layer:

- modulo( $Q_s, t$ ) = 0 and modulo( $Q_s, 2t$ ) = 0:

$$n = (2g - 1)t \quad (7.12)$$

- modulo( $Q_s, 2t$ ) > 0:

$$n = gt \quad (7.13)$$

- modulo( $Q_s, t$ ) > 0:

$$n = \frac{gt}{2} \quad (7.14)$$

- Double layer:

- modulo( $Q_s, t$ ) = 0

$$n = (2g - 1)t \quad (7.15)$$

- all other cases

$$n = gt \quad (7.16)$$

## Star of Slots

---

A classical drawing of star of slots<sup>1</sup> is used to determine the harmonic amplitude. The star of slots is formed by  $Q_s$  phasors, where  $Q_s$  is the number of slots and the machine periodicity  $t$  is given by equation (7.5).

The star of slots is the phasor representation of the main EMF harmonic induced in the coil side of each slot characterized by  $Q_s/t$  spokes, with each spoke containing  $t$  phasors. The angle between the phasors of two adjacent slots is the electrical angle (= mechanical angle x number of pole pairs).

$$\alpha_s^e = p \cdot \beta \quad (7.17)$$

Where  $\beta$  ( $=2\pi/Q_s$ ) is the slot angle in mechanical radians. The way in which the Star of Slots representation is calculated is by evaluating the electrical angles between the slots. This can be obtained based on the above equation as:

$$\alpha_s^e(\text{slot}) = \text{mod}(((\text{slot} - 1) \cdot p \cdot 360/Q_s), 360) \quad (7.18)$$

The angle between two adjacent spokes,  $\alpha_{ph}$ , is given by

$$\alpha_{ph} = \frac{2\pi}{(Q_s/t)} \quad (7.19)$$

The distribution factor also depends on the number of spokes per phase  $qph$ , given by:

$$qph = \frac{Q_s}{m \cdot t} \quad (7.20)$$

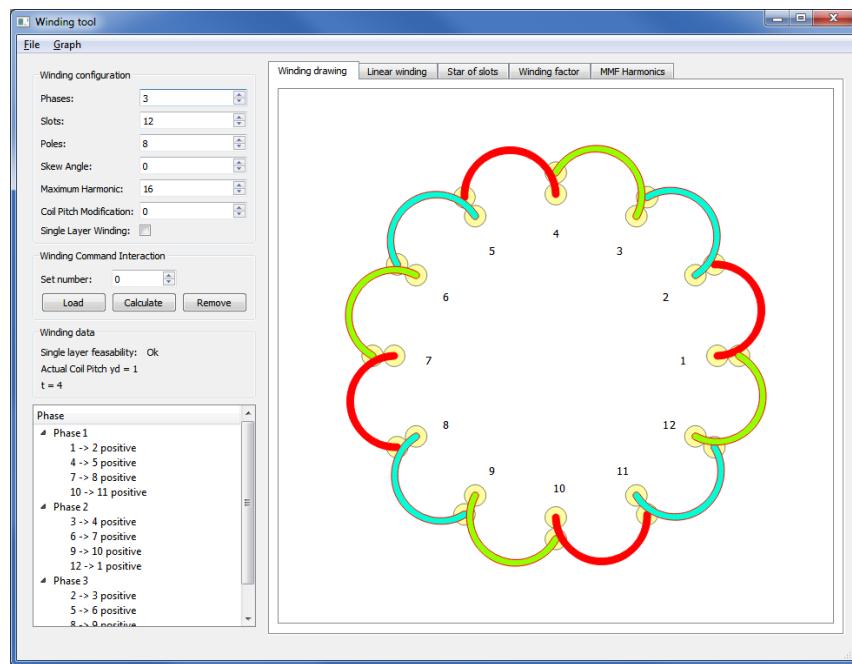
---

<sup>1</sup>N. Bianchi and M. Pre, "Use of Star of Slots in designing fractional-slot single-layer synchronous motors".

## Basic Usage

### Launching the winding tool

The winding tool can be launched using the toolbutton  from Opera-2d/Pre and Post Processor and Opera-3d/Modeller. The tool is launched in a new window as shown in [Figure 7.2](#)



*Figure 7.2 Default winding tool window*

### Use case 1: Fixed number of slots and poles

Users can enter a combination of number of slots and poles to investigate possible winding connections.

#### Winding Layout

[Figure 7.3](#) shows an optimal winding layout for a fractional slot, fractional pitch, 54-slot, 10-pole, 3-phase machine. [Figure 7.4](#) shows a linear representation of the winding layout for the above combination.

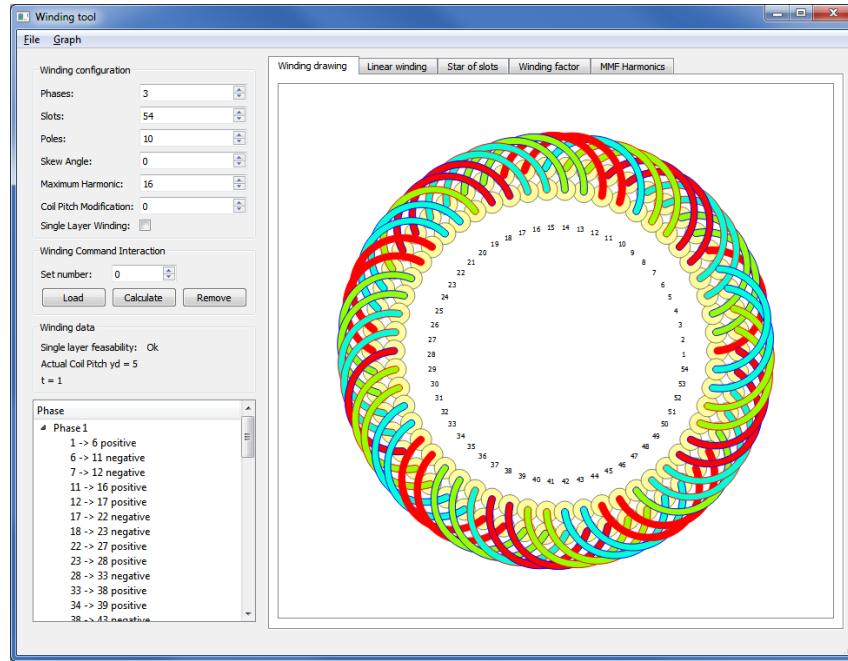


Figure 7.3 Winding layout for a 54-slot, 10-pole, 3-phase machine

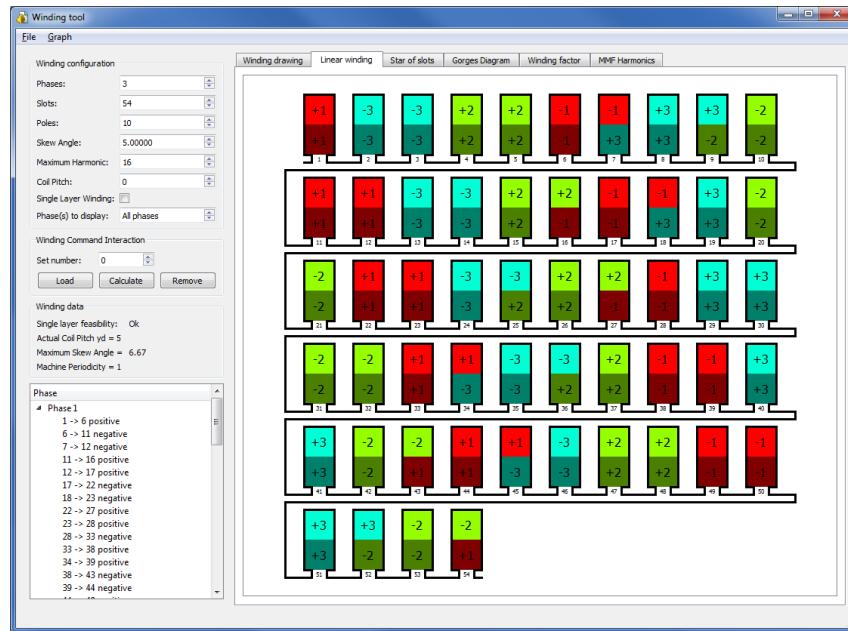


Figure 7.4 Linear representation of the winding layout for a 54-slot, 10-pole, 3-phase machine

## Winding Harmonics

Figure 7.5 and Figure 7.6 show the winding factor and winding mmf harmonics for the above machine.

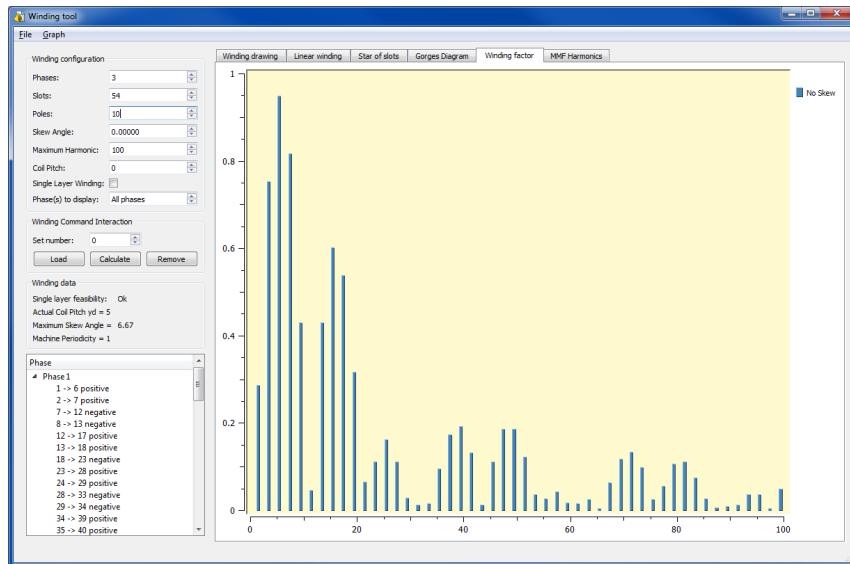


Figure 7.5 Winding factor harmonics for a 54-slot, 10-pole, 3-phase machine

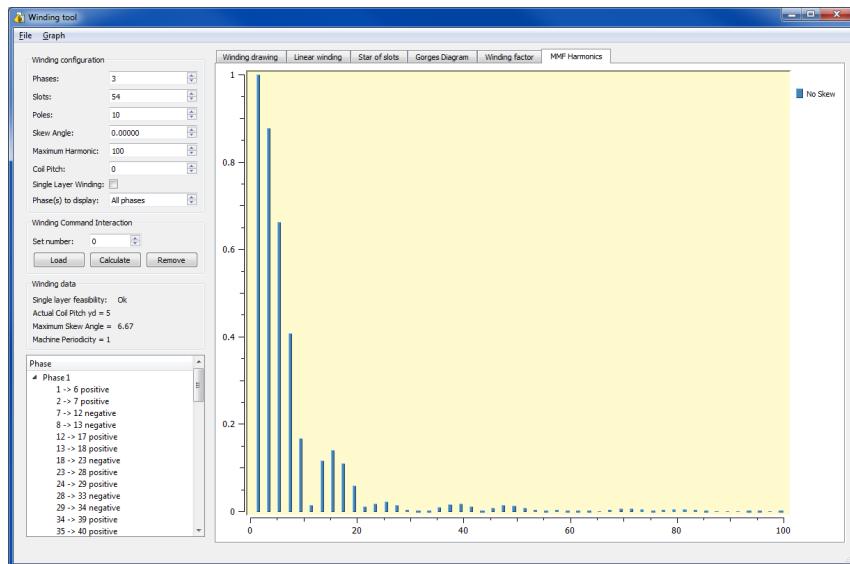


Figure 7.6 Winding mmf harmonics for a 54-slot, 10-pole, 3-phase machine

## Winding Skew

Users can also enter a skew angle for the windings to investigate how the harmonics vary when a skew is considered. Figure 7.7 shows winding factor harmonics for a configuration without skew and one with a skew of 5 degrees.

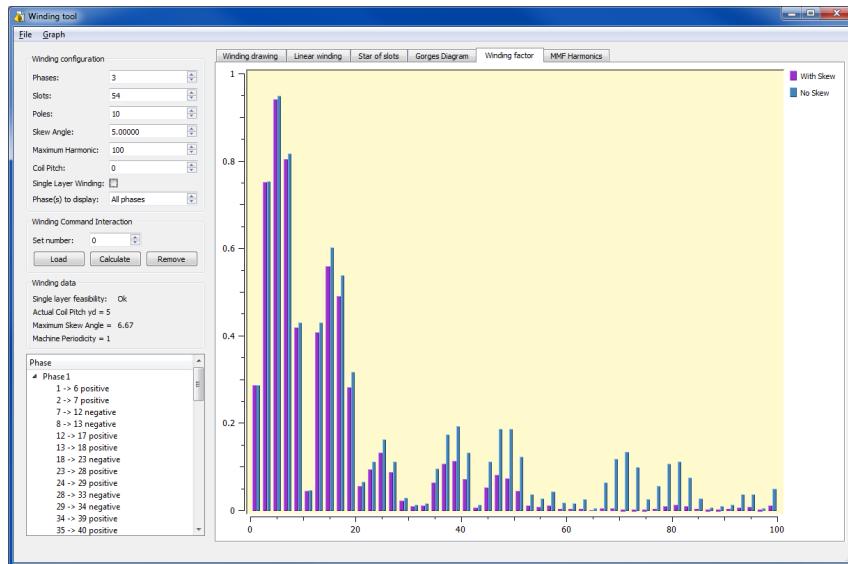


Figure 7.7 Winding factor harmonics for a 54-slot, 10-pole, 3-phase machine with skew of 5 degrees.

## Star of Slots

In Figure 7.8 the star of slots representation for the 54-slot, 10-pole, 3-phase winding is shown.

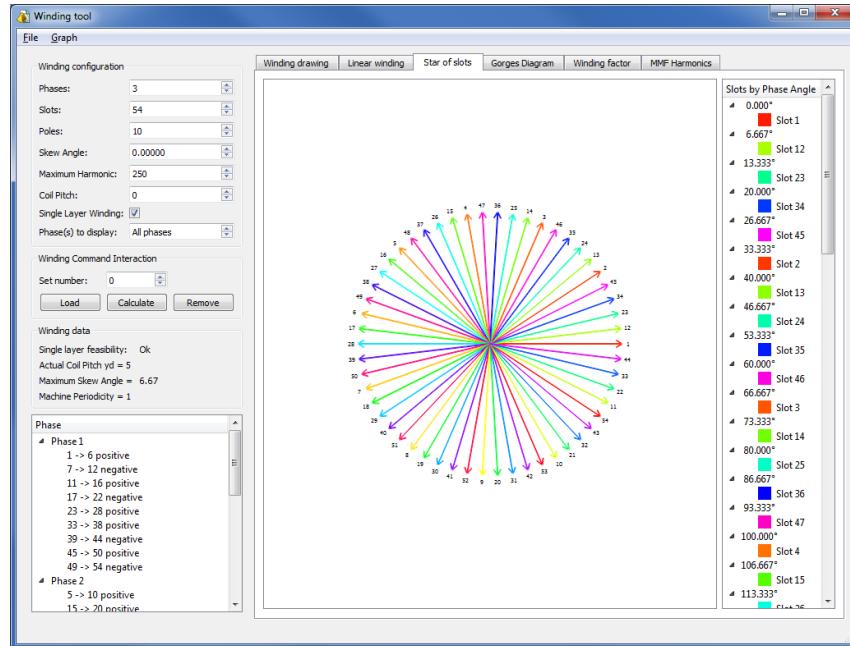


Figure 7.8 Star of slots representation for 54-slot, 10-pole, 3-phase winding.

## Görges Diagram

The voltage phasor diagram, also known as Görges diagram, obtained from the star of slots representation and the winding layout, is also presented to the user. Figure 7.9 shows the voltage phasor diagram for the 54-slot, 10-pole winding.

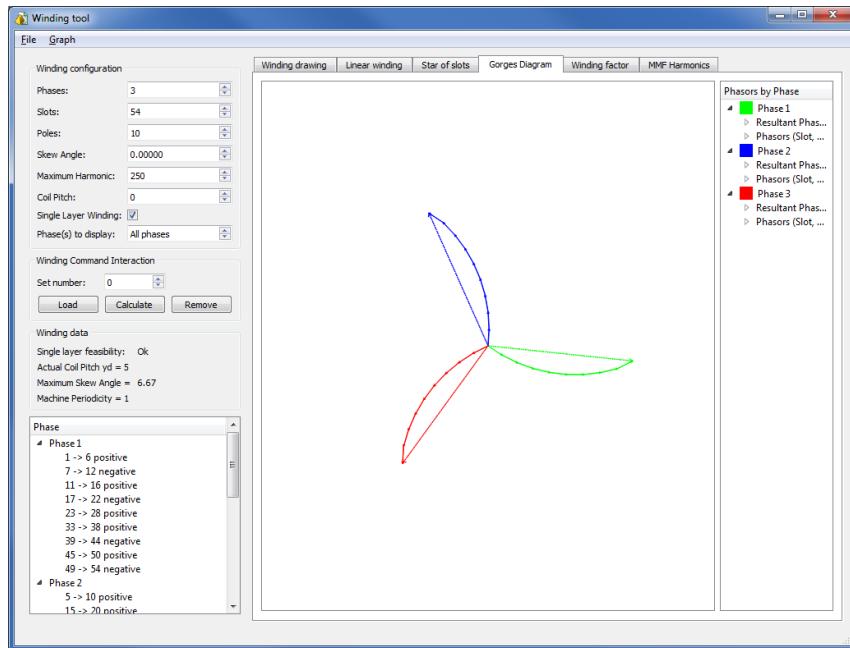


Figure 7.9 Görges diagram for 54-slot, 10-pole winding

## Winding Command Interaction

Users can interactively issue the **WINDING** command to either:

- **CALCULATE** a set of results for a particular slot-pole combination; or
- **LOAD** a previously calculated set of results.

The **OPTION=CALCULATE** parameter of the **WINDING** command will calculate a set of results and also create an Opera Python object, which will store all the information for that particular combination. The **SET** parameter of the command (used along with **OPTION=CALCULATE**) allows users to store a particular combination in memory, in order to view the results later.

The **OPTION=LOAD** parameter then allows users to switch between previously stored **SETS**.

## Use case 2: Range of number of slots

Users can enter the range of slots and poles to understand how a particular slot-pole combination of machine winding will perform.

A winding wizard can be used to characterize windings based on the fundamental winding factor and the lowest common multiple (LCM) of number of slots and number of poles, which determines the number of cogging torque periods per full mechanical rotation.

The wizard can be launched from the "File" menu of the Winding Tool window.

Figure 7.10 shows all the feasible windings for the entered range of slots and poles.

The screenshot shows a Windows application window titled "Form". Inside, there's a section labeled "Instructions" with a note: "Select a range of slots and poles, and click the "Search windings" button. The feasible combinations will be shown in the table." Below this are two sets of input fields: "Slots Q" (From: 3, To: 21) and "Poles 2p" (From: 2, To: 16). A "Search windings..." button is located between them. To the right is a large table with columns: Slots (Q), Poles (2p), Winding Factor (kw), Periodicity (t), Pitch (yd), LCM, and Single Layer Feasible. The table lists 20 rows of data. At the bottom are "Reset", "OK", and "Cancel" buttons.

Slots (Q)	Poles (2p)	Winding Factor (kw)	Periodicity (t)	Pitch (yd)	LCM	Single Layer Feasible	
1	3	2	0.866025	1	1	6	false
2	3	4	0.866025	1	1	12	false
3	3	8	0.866025	1	1	24	false
4	3	10	0.866025	1	1	30	false
5	3	14	0.866025	1	1	42	false
6	3	16	0.866025	1	1	48	false
7	6	2	1	1	3	6	true
8	6	4	0.866025	2	1	12	true
9	6	8	0.866025	2	1	24	true
10	6	10	0.5	1	1	30	true
11	6	14	0.5	1	1	42	true
12	6	16	0.866025	2	1	48	true
13	9	2	0.945214	1	4	18	false
14	9	4	0.945214	1	2	36	false
15	9	6	0.866025	3	1	18	false
16	9	8	0.984808	1	1	72	false
17	9	10	0.984808	1	1	90	false
18	9	12	0.866025	3	1	36	false

Figure 7.10 Winding Wizard showing feasible windings for selected range of slots and poles

A particular slot-pole combination can be selected and winding characteristics can be viewed in detail. Figure 7.11 shows a particular combination (21-slot, 16-pole, 3-phase winding) being selected from the Winding Wizard.

This screenshot is identical to Figure 7.10, but the last row of the table is highlighted in blue, indicating it is selected. The selected row contains the values: Slots (Q) 21, Poles (2p) 16, Winding Factor (kw) 0.930874, Periodicity (t) 1, Pitch (yd) 1, LCM 336, and Single Layer Feasible false.

Figure 7.11 Selecting 21-slot, 16-pole combination

By selecting the "OK" button the winding configuration is loaded into the winding tool, where the winding layout (as shown in Figure 7.12) and other characteristics can be viewed for the selected combination.

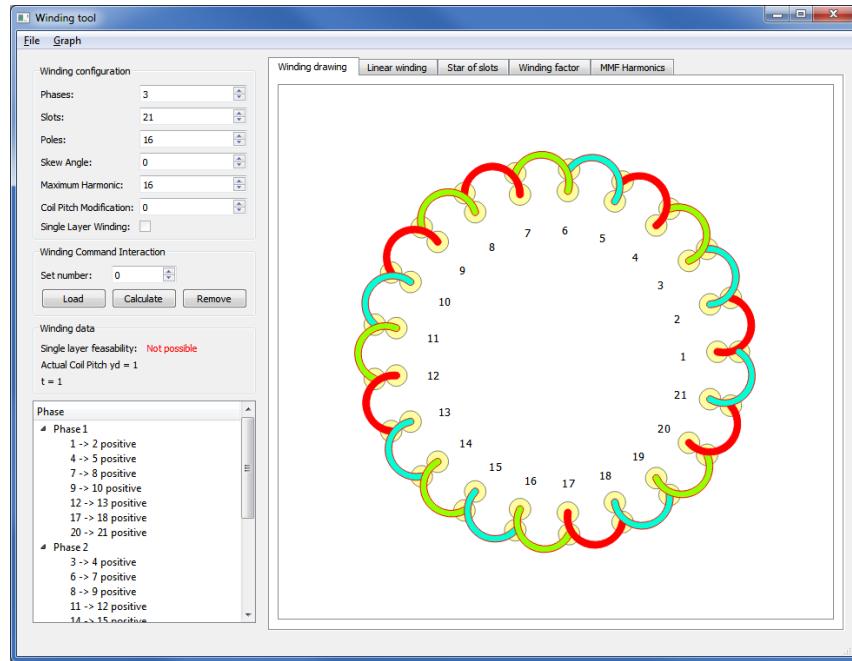


Figure 7.12 Winding layout for a 21-slot, 16-pole, 3-phase machine winding

# **Chapter 8**

# **Induction Heating: Thermal and Multiphysics**

## **Introduction**

---

This example shows the use of a **Static Thermal** analysis to determine the temperature distribution in a copper plate. Two analyses are performed. The first determines the temperature distribution assuming that the heat density<sup>1</sup> throughout the plate is uniform. In the second analysis the heat density is imported from a **Harmonic Electromagnetic** solution of the plate in the presence of an induction coil at several frequencies.

The analysis includes the specification of thermal material properties and heat transfer boundary conditions. In addition, layering is used to produce a suitable mesh for modelling skin-effect in the **Harmonic Electromagnetic** solution.

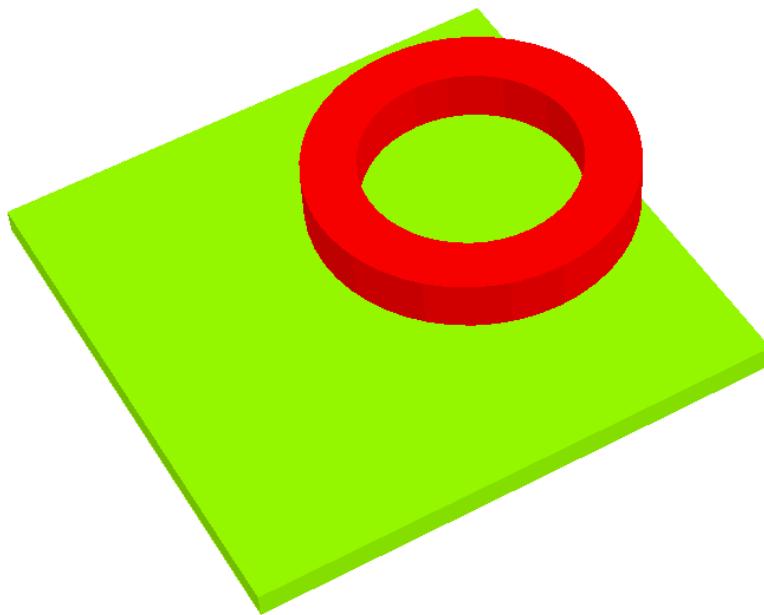
## **The Model**

Figure 8.1 shows the geometry of the example. The copper plate is 20 x 20 in square with a thickness of 1 in<sup>[2]</sup>. The solenoid coil is positioned 2 inches above the plate and is centred on the centre-line of the plate. The outer edge of the coil is directly above the edge of the plate. The solenoid comprises 50 turns carrying a current of 300 A rms at 100 Hz.

---

<sup>1</sup>The term "heat density" is used for the power per unit volume.

<sup>2</sup>The model could be defined in SI units with factors of 0.0254 included in all the dimensions. To demonstrate the use of unit sets, this example uses "SI with inches".



*Figure 8.1 Induction heating coil and copper plate*

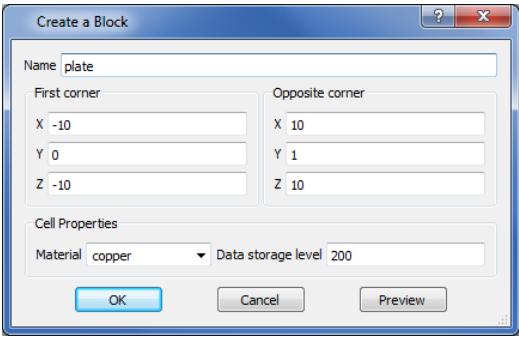
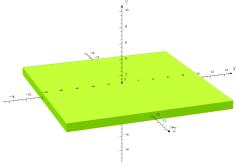
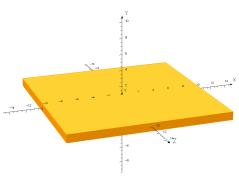
The material properties of the copper plate are:

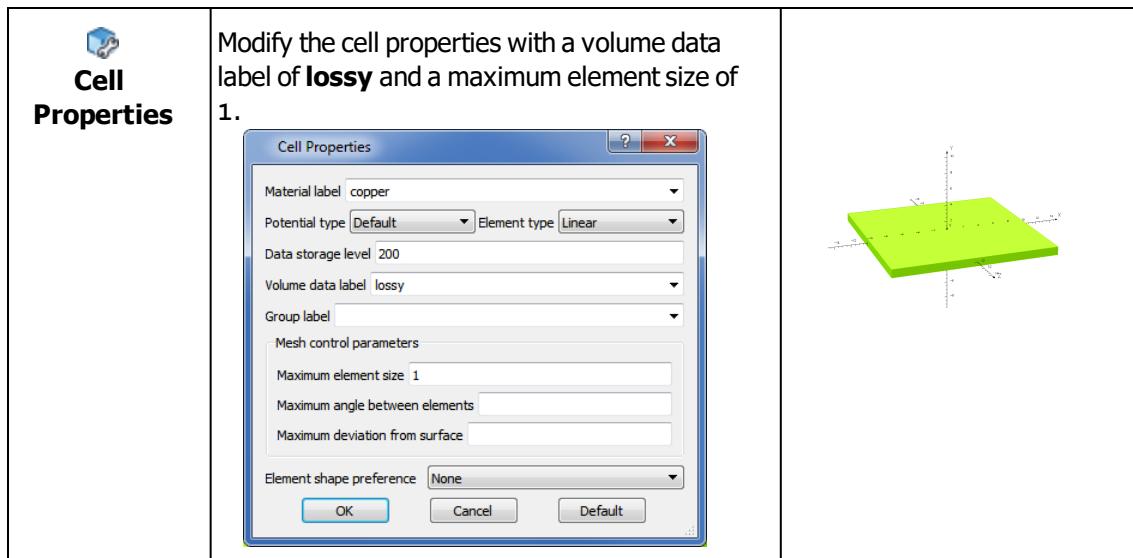
- Electrical conductivity  $1.27 \times 10^6$  S/in
- Thermal conductivity 10 W/(in K)
- The plate is cooled by air with a temperature of 20°C and the heat transfer coefficient is 0.002 W/(in<sup>2</sup> K)

## Building the Thermal Model

### Building the Plate Geometry and Mesh

#### Plate geometry

 <b>Create Block</b>	<p>Create a plate from (-10, 0, -10) to (10, 1, 10) made of copper with a data storage level of 200.</p> 	
 <b>Pick Cells</b>   <b>Pick all filter type entities</b>		

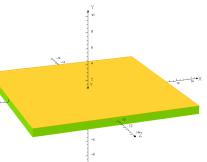
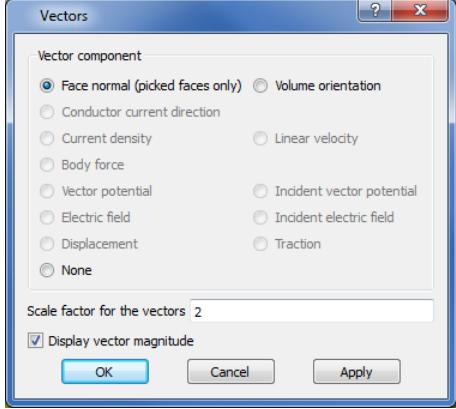
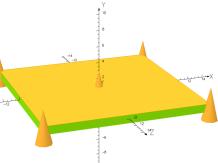


The volume data label will be used to define the heat density in the plate.

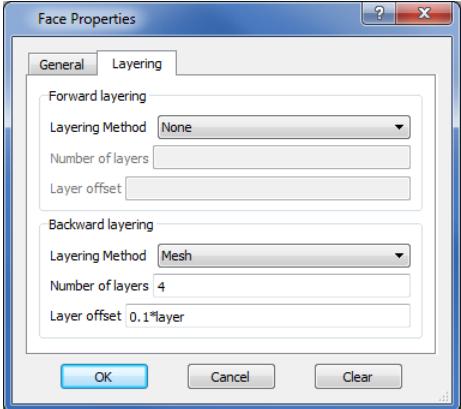
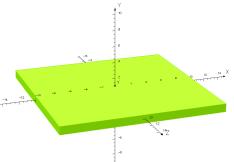
## Building layers

In the model with the induction heating coil later in this chapter, the eddy currents induced in the plate will be subject to skin-effect. The surface current density on the square face of the plate on the same side as the coil will be high and will decay exponentially into the depth of the plate. The skin-depth of copper at 100 Hz is about 0.25 inch. Consequently the finite element mesh needs to be sufficiently fine to model the decay with at least two elements in the first skin depth and one in the second. One way to achieve this is to make all the elements on the surface at  $Y=1$  the same size as that required to model the first skin-depth. However this results in over-discretization as the variation in  $X$  and  $Z$  is much less rapid.

A better way to achieve a fine discretization in the  $Y$  direction without making very small elements in  $X$  and  $Z$  is to use layering. The user may specify layers of elements of constant or functionally varying depth that are "lofted" from a face. The discretization pattern of the original face will be repeated on the surface between each layer. The system variable **layer** may be used to specify the variation of the layer depth.

 <b>Pick Faces</b>	<p>Pick the square face at Y = 1 by double-clicking when the <b>Pick faces</b> option is selected.</p>	
 <b>Vectors</b>	<p>Select the option for <b>Face normal (picked faces only)</b> and set the <b>Scale factor</b> to 2</p> 	

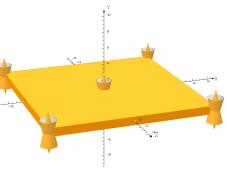
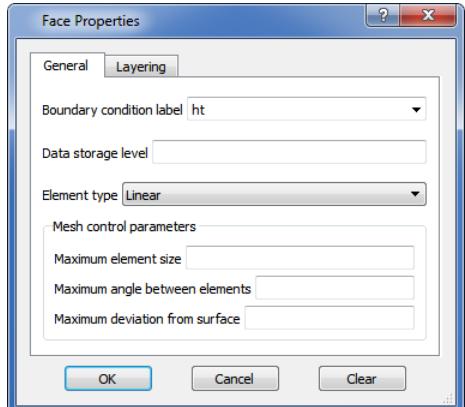
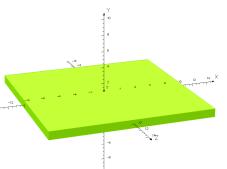
To use the layering facilities the user must know which are the forward and backward directions. The vectors point in the "forward" direction.

 <b>Face Properties</b>	<p>Select the <b>Layering</b> tab, and for <b>Backward layering</b>, set:  <b>Layering Method: Mesh</b>,  <b>Number of layers: 4</b> and  <b>Layer offset: 0.1*layer</b>.</p> 	
---	---	---

The definition of the layers has no immediate effect; the layers will be created during mesh generation. With the above parameters there will be 4 layers of elements with thicknesses 0.1, 0.2, 0.3 and 0.4 in. For more information, see [Layering \[page 546\]](#).

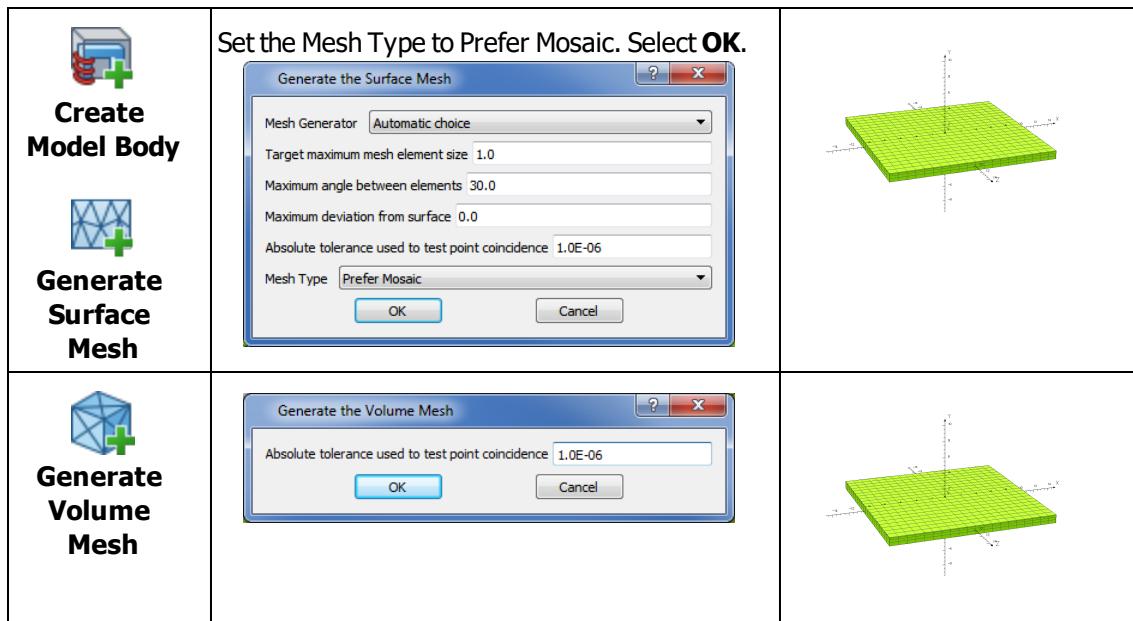
## Boundary condition labels

Boundary condition labels are now set as follows. The vectors can now be turned off if required

 <b>Pick Faces</b>   <b>Pick all filter type entities</b>	Pick all the faces of the plate.  Click on the <b>General</b> tab, and set boundary condition label <b>ht</b> .	
 <b>Face Properties</b>		

## Meshing

The surface and volume meshes are created.



## Static Thermal Analysis

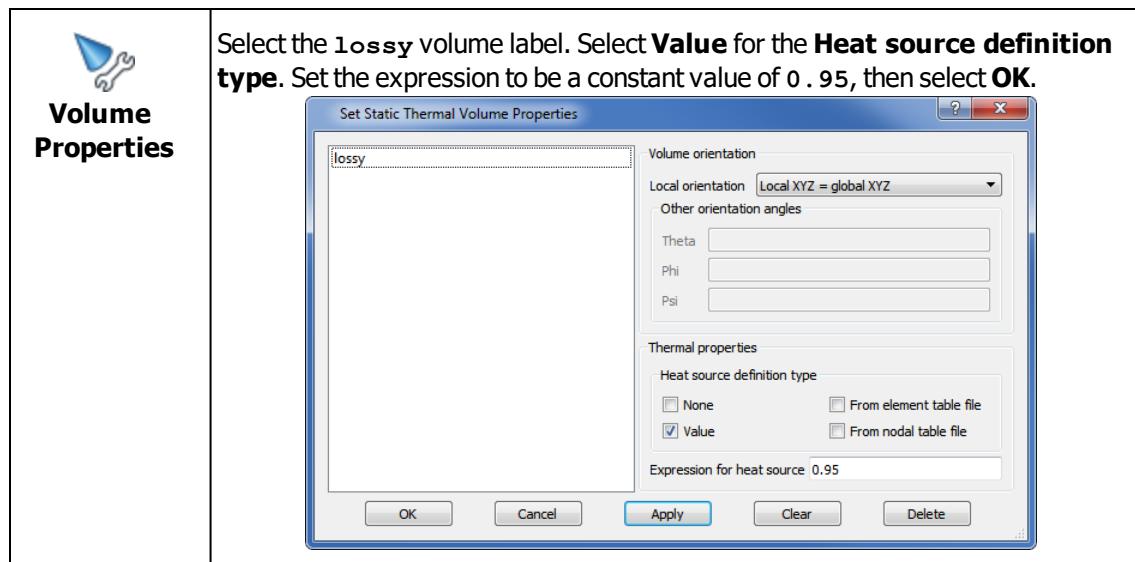
### Setting up the Thermal Analysis

Set the analysis type to **Static Thermal**. This will determine the nature of some of the following dialogs, and the kind of information that will be required. Accept the default options in the analysis configuration pane.

### Setting the Thermal Properties

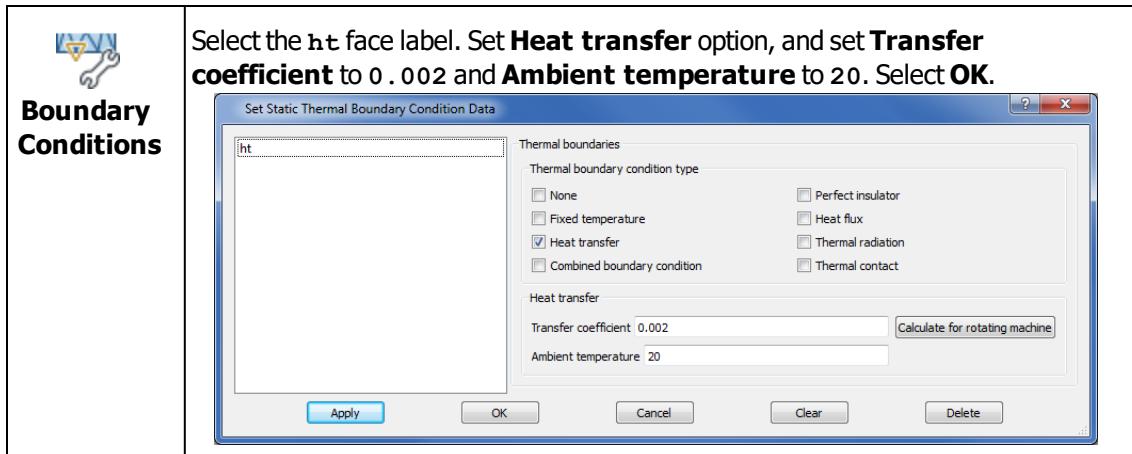
#### Heat input

For the first example a uniform heat density of 0.95 W/in<sup>3</sup> is applied throughout the plate.



#### Boundary conditions

A heat transfer coefficient of 0.002 W/(in<sup>2</sup>K) is applied to all surfaces of the plate. The ambient temperature outside the plate is 20 °C.

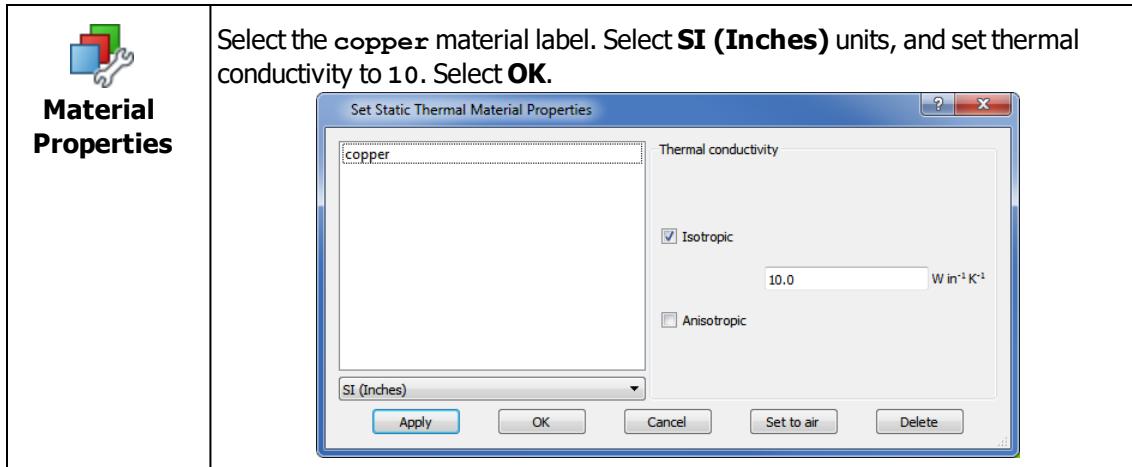


## Temperature units

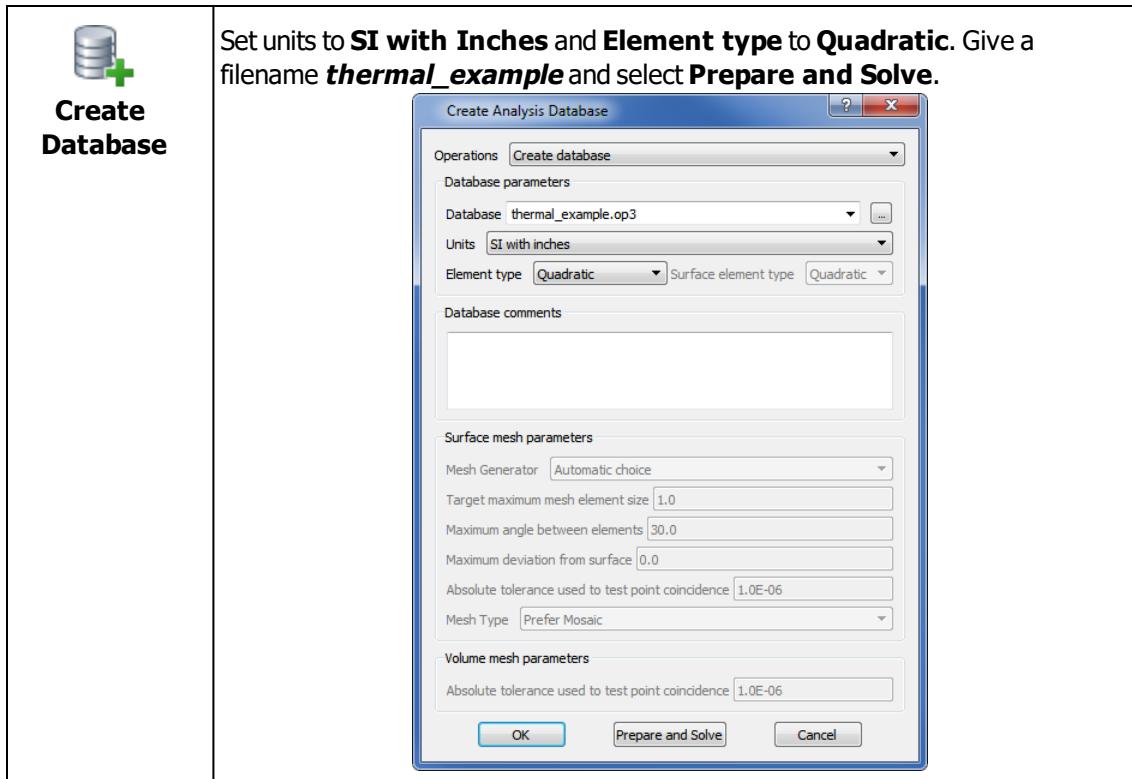
Note the use of temperatures in  $^{\circ}\text{C}$ . The user is free to use either  $^{\circ}\text{C}$  or K for temperature as long as all temperatures are expressed in the same unit. In compound units involving "per temperature unit" (heat transfer coefficients, thermal conductivities, etc.)  $^{\circ}\text{C}$  and K are equivalent.

## Thermal conductivity

A thermal conductivity of 10 W/(in K) is applied to the plate.



## Save Model and Create the Database



A Modeller data file (**thermal\_example.opc**) and an analysis database (**thermal\_example.op3**) will be created.<sup>1</sup>

When the analysis is complete, select Post-Process from the solver window.

---

<sup>1</sup>The Modeller data file is also provided in the Opera installation - see the Examples Folder from the Opera Manager File menu.

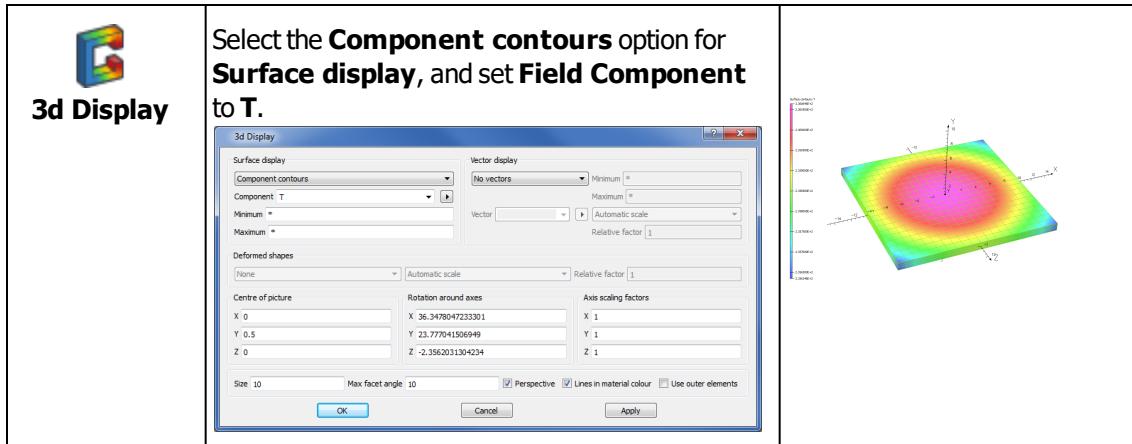
## Post-Processing the First Thermal Analysis

### Thermal Results

The thermal analysis solves for the temperature at each node in the finite element mesh. Other results available include the heat flow and the temperature gradient, both of which are vector quantities. The estimated numerical error in the heat flow is also computed. Thermal models can be examined in the same way as other analysis **op3** files.

#### Displaying temperature

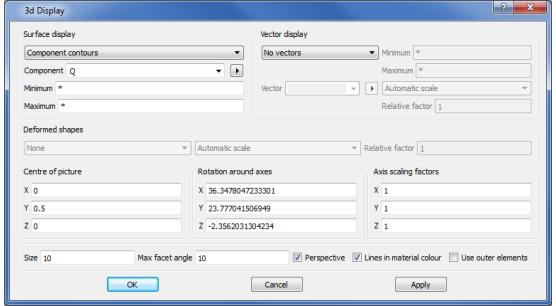
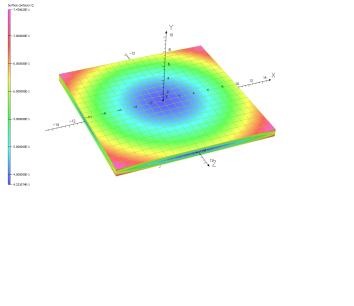
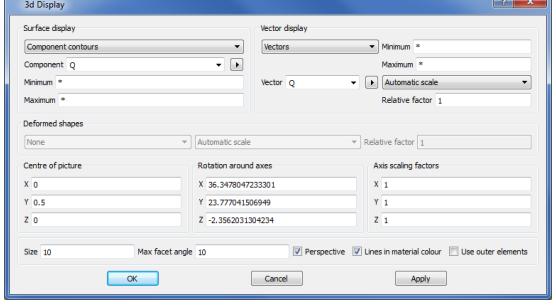
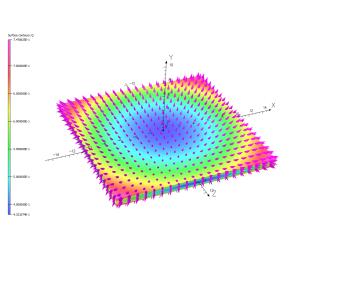
The temperature can be displayed by selecting component **T**.



As would be expected, the temperature distribution is very even over the whole plate ranging from 235.6 to 236.1 °C.

#### Heat flow

Heat flow can also be obtained.

 <b>3d Display</b>	<p>The magnitude of heat flow can be displayed using <b>Q</b> or <b>QMOD</b> and the vector components by <b>QX</b>, <b>QY</b> and <b>QZ</b>.</p> 	
 <b>Outline view of model (toggle)</b>   <b>3d Display</b>	<p>Enable <b>Vectors</b>, and set <b>Vector</b> quantity to <b>Q</b>.</p> 	

The Post-Processor can now be closed.

## Multiphysics Analysis



If the Modeller is still running, return to it and select **Delete model body**

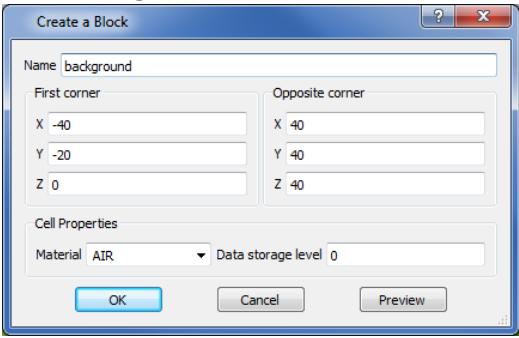
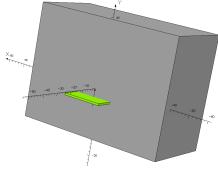
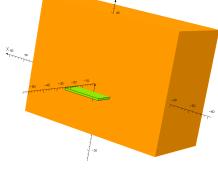


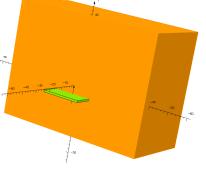
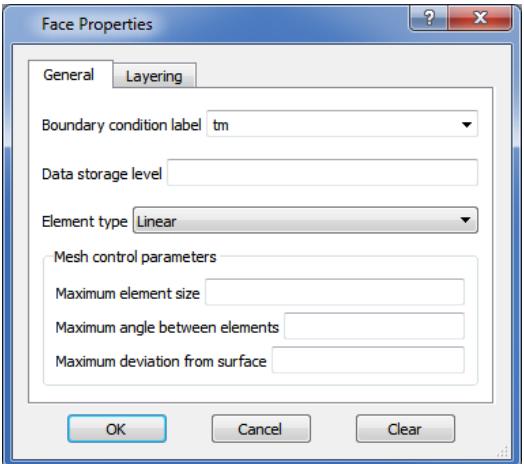
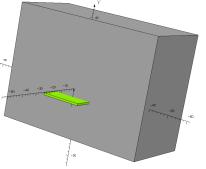
Otherwise, re-start the Modeller and read in the **thermal\_example.opc** file. Delete the model body, as above.

The **Harmonic Electromagnetic** analysis model requires that the plate is surrounded by a region of air which uses reduced potentials (because it will contain the coil). The symmetry of the model will be exploited by using a volume that bisects the coil and the plate.

The body of air will be given the special name, **BACKGROUND**. When the model body is created, the program will retain only the intersection of the model with the body named **BACKGROUND**. (This way of providing a background region is an alternative to the Model Symmetry as explained previously in [Surrounding Air and Boundary Conditions \[page 50\]](#). It allows background regions of other shapes than blocks, cylinders etc., although in this case only a block is used.)

The magnetic field on the symmetry plane will be tangential to the plane. All the outer boundary faces of the background will also be assumed to be tangential magnetic.

 <b>Create Block</b>   <b>Reset to Initial view</b>	<p>Create a <b>background</b> region of <b>Air</b> from <math>(-40, -20, 0)</math> to <math>(40, 40, 40)</math> with <b>Data storage level</b> set to 0.</p>  <p>Select  <b>Initial View</b> to resize the picture to include all of the background region. Rotate view to obtain a view similar to that shown on right.</p>	
 <b>Pick Entity</b>   <b>Pick Cells</b>	<p>Double-click on the background cell</p>	

 <b>Pick Faces</b>   <b>Change type of picked entities</b>	<p>Note that the display of entities in the status bar changes from 1 picked cell to 6 picked faces, i.e. the faces that make up the background cell. The display remains unchanged (if the vectors showing normal directions are still visible - these can be turned off using ).</p>	
 <b>Face Properties</b>	<p>Apply a boundary condition label of <code>tm</code> to the faces.</p> 	

## Analysis Settings

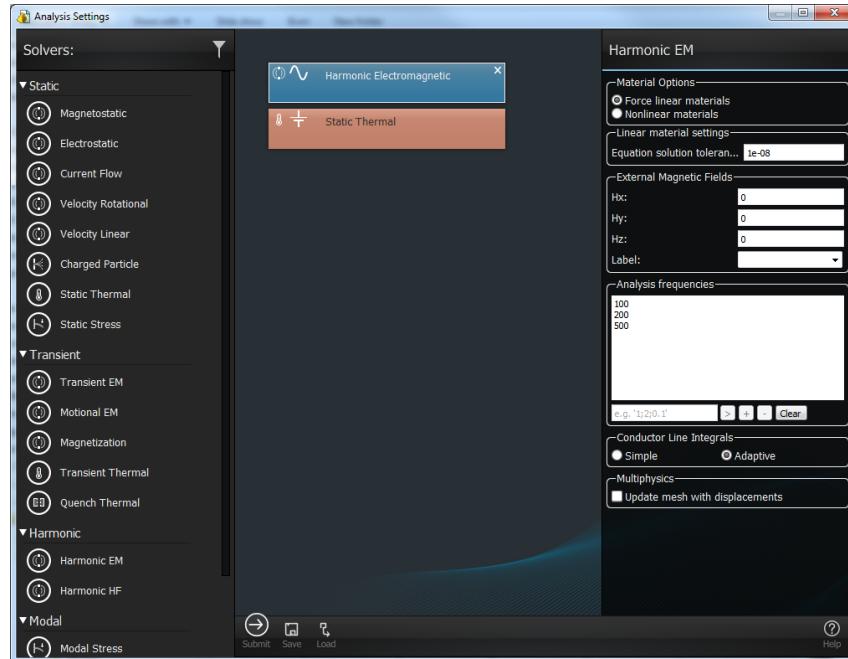
Multiphysics simulations are defined as a sequence of analyses. In this model, the induction heating coil will be energized with an AC current and the resulting eddy current losses in the plate (calculated automatically by the multiphysics simulation) will be used as the heat density for the thermal analysis.

When the eddy current analysis is defined, three frequencies will be analysed. The subsequent thermal analysis will use each resulting heat density distributions as the source in three separate simulations.

In the **Analysis Settings** window clear any existing solvers and add a **Harmonic Electromagnetic** solution followed by a **Static Thermal** one.

The analysis settings can be adjusted for each stage of the multiphysics simulation. For the thermal solution set the same settings as in the first example. For the **Harmonic Electromagnetic** analysis, the frequencies should be modified to 100, 200 and 500 Hz. Leave all the other settings to their default values (see [Figure 8.2](#)). Setting multiple frequencies in this way means that all subsequent stages will be run after each frequency. In this case there will be a total of 6 analyses:

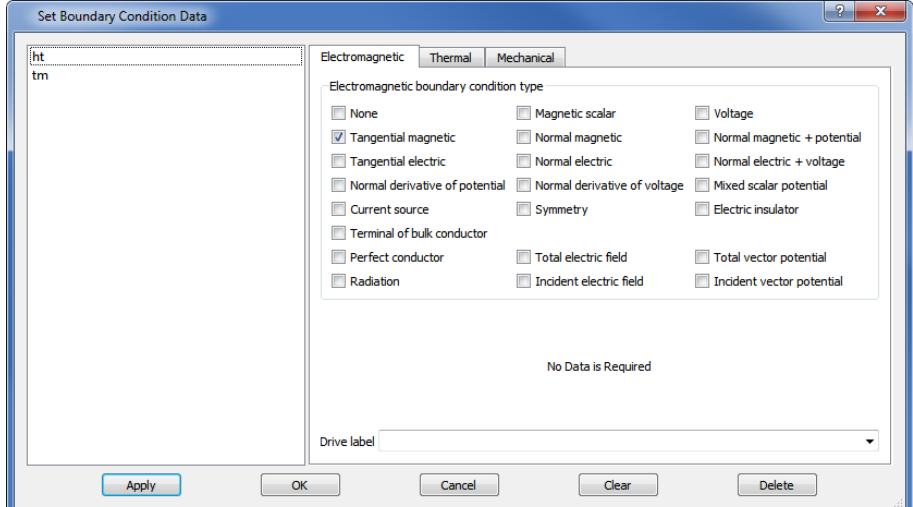
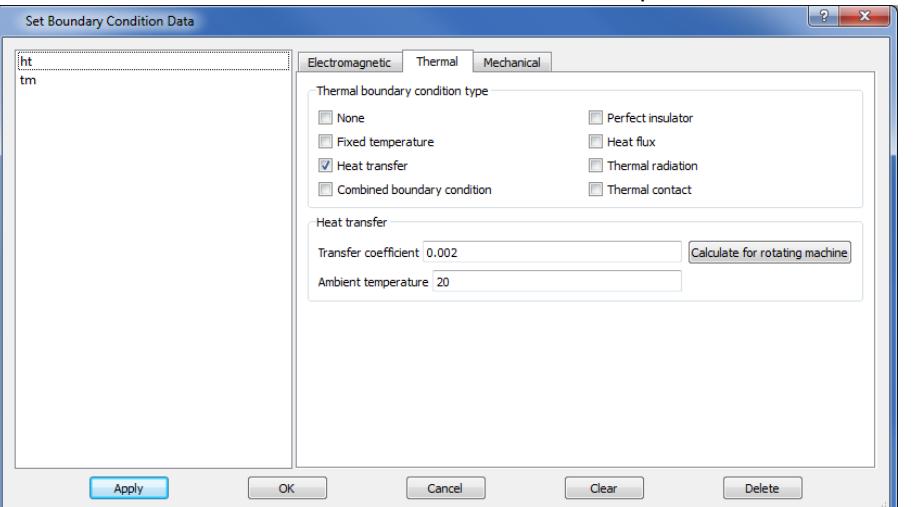
1. **Harmonic Electromagnetic** at 100 Hz.
2. **Static Thermal** using heat from stage 1.
3. **Harmonic Electromagnetic** at 200 Hz.
4. **Static Thermal** using heat from stage 3.
5. **Harmonic Electromagnetic** at 500 Hz.
6. **Static Thermal** using heat from stage 5.



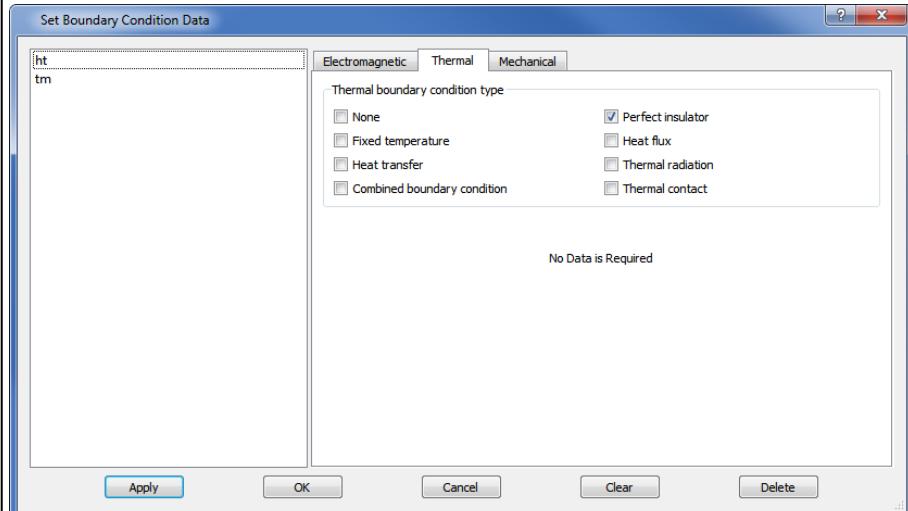
*Figure 8.2 Analysis settings dialog for Harmonic EM and Static Thermal solution*

## Boundary conditions

Set the electromagnetic boundary condition to be Tangential Magnetic for the **tm** label. The heat transfer boundary condition on the outside of the plate set in the first thermal example will be imported into this example. The thermal boundary condition on the **tm** label will be set as a perfect insulator, implying the symmetry of the geometry, heat source and boundary conditions.

 <b>Boundary Conditions</b>	<p>On the <b>Electromagnetic</b> tab, set boundary condition label <b>tm</b> to be <b>Tangential Magnetic</b>.</p>  <p>The dialog shows the 'Electromagnetic' tab selected. In the left pane, 'ht' is listed under 'Boundary condition label'. In the right pane, 'Tangential magnetic' is checked under 'Electromagnetic boundary condition type'. Other options like 'None', 'Normal magnetic', and 'Voltage' are also listed.</p>
	<p>Now move to the <b>Thermal</b> tab. High-light the <b>ht</b> boundary condition label which shows that the heat transfer condition from the first example has been retained.</p>  <p>The dialog shows the 'Thermal' tab selected. In the left pane, 'ht' is listed under 'Boundary condition label'. In the right pane, 'Heat transfer' is checked under 'Thermal boundary condition type'. Other options like 'None', 'Fixed temperature', and 'Perfect insulator' are listed. Below, 'Transfer coefficient' is set to 0.002 and 'Ambient temperature' is set to 20.</p>

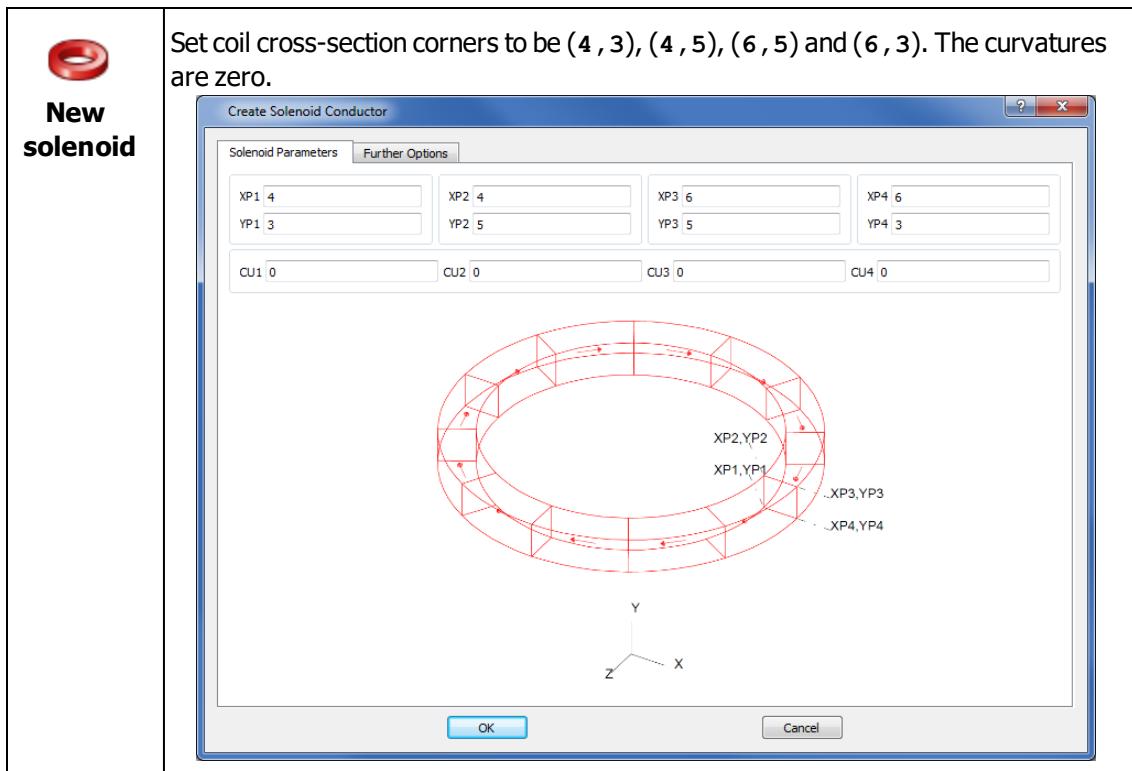
Staying on the **Thermal** tab, set the **tm** boundary condition label as a **Perfect insulator**



Click on **OK**

## Creating the solenoid

The current carrying coil is a simple solenoid, and is defined as follows. The **OK** button in the **Create Solenoid Conductor** dialog is disabled until all the necessary data have been provided.

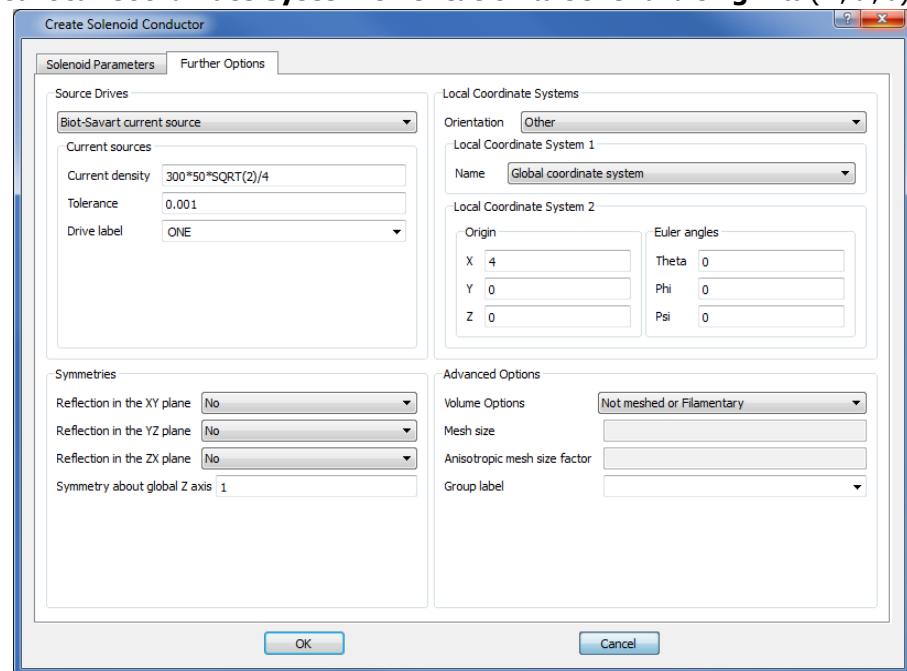


Use an algebraic expression to define the current density in the coil: 300 A in 50 turns in a cross-sectional area of 4 in<sup>2</sup>. Include a factor of root 2 to convert the rms current to the amplitude.

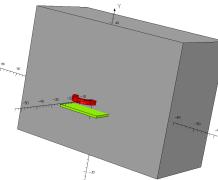
Set **Current density** to  $300 * \text{sqrt}(2) * 50 / 4$ .

Set **Tolerance** to 0.001, and **Drive label** to ONE.

Set **Local Coordinate System Orientation** to **Other** and **Origin** to (4, 0, 0).

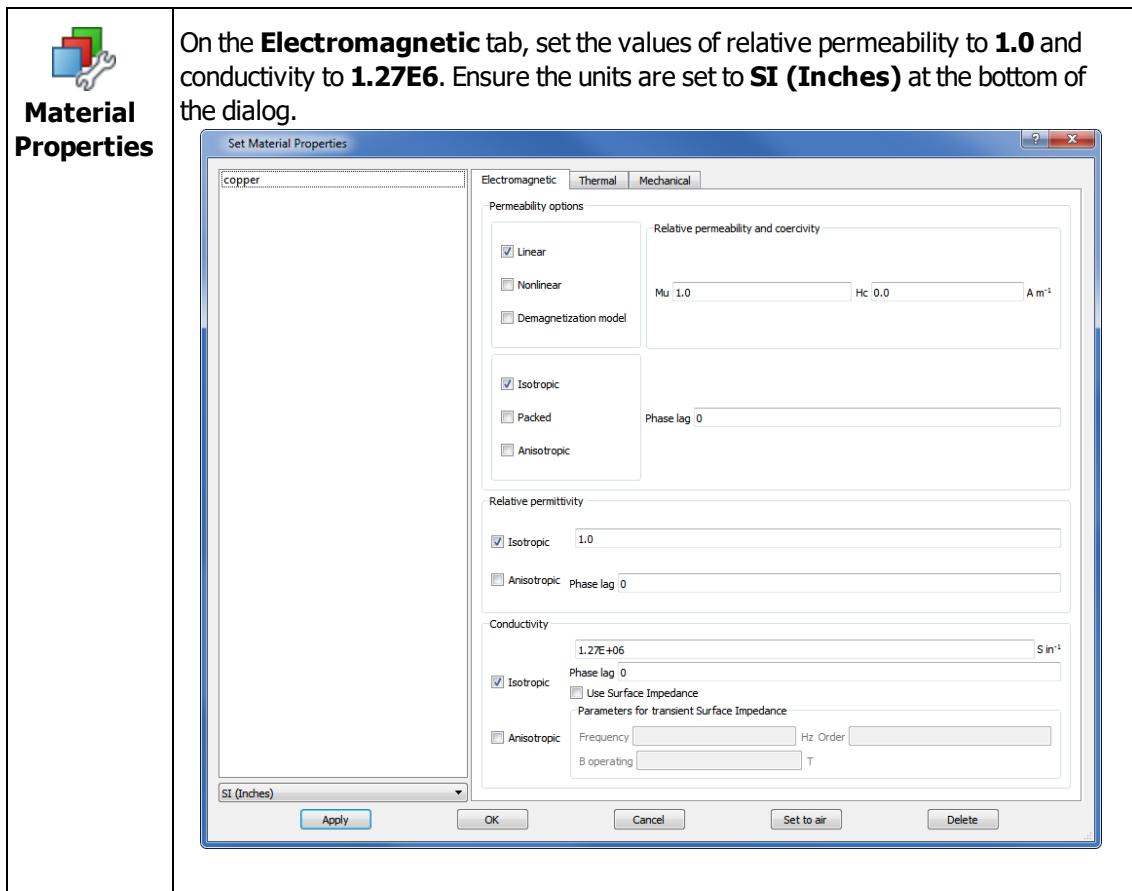


Select **OK** to create and display the conductor.



## Electromagnetic material properties

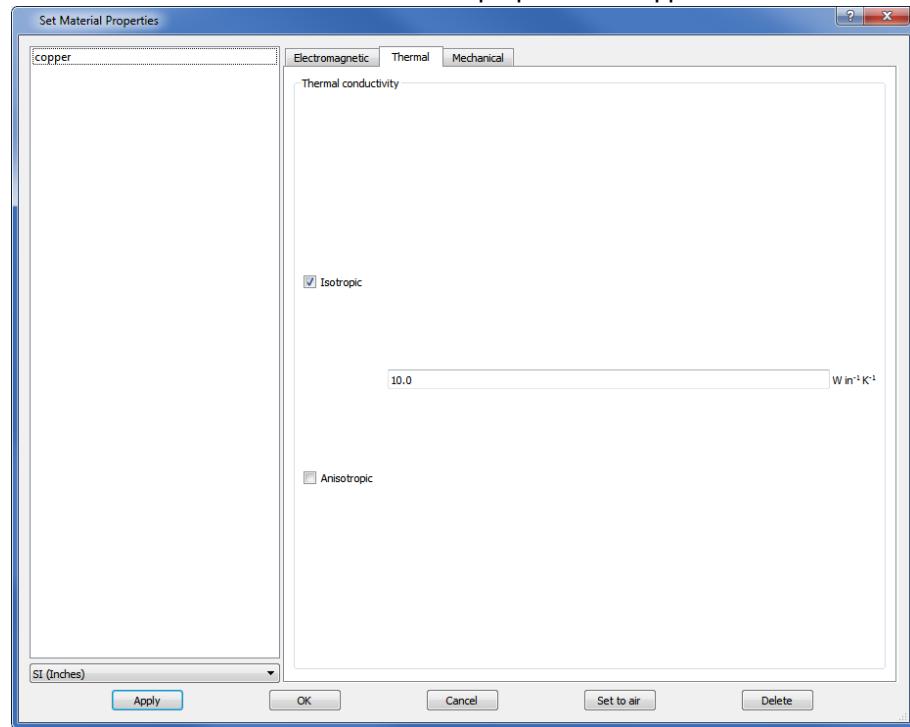
All non-air materials in an eddy current analysis require a value of permeability and electrical conductivity.



Note that the dialog gives the possibility of setting a coercive field strength for the material, as this may be required in a subsequent multiphysics analysis. However, it is not valid and will be ignored in **Harmonic EM**.

As with the boundary conditions, the thermal material properties will be copied automatically from the first model.

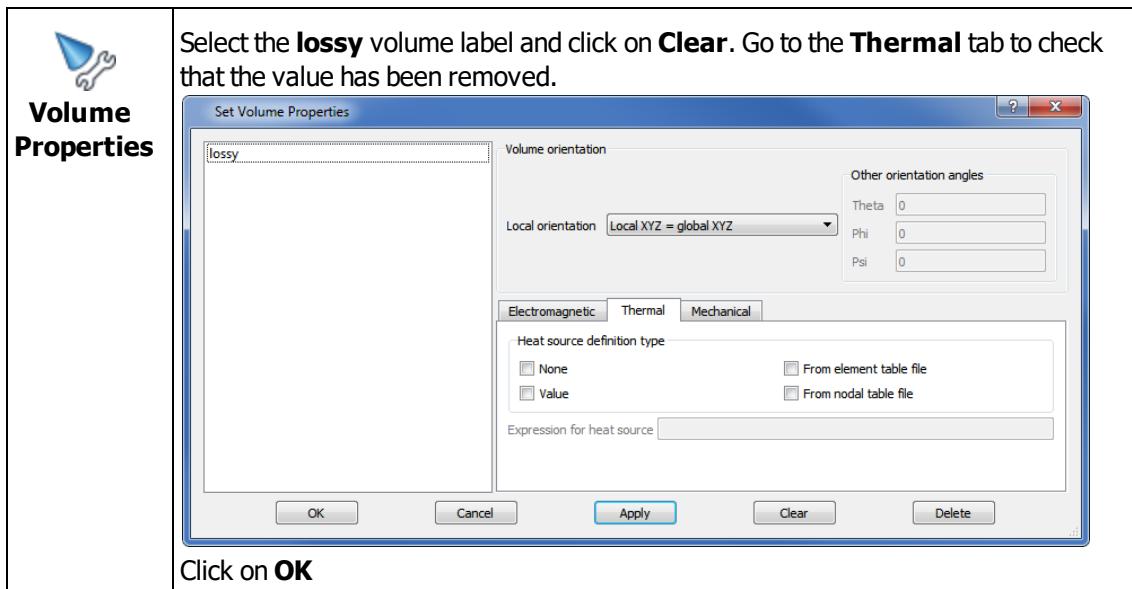
Open the **Thermal** tab to check the thermal properties of copper.



Click on **OK**

## Volume label

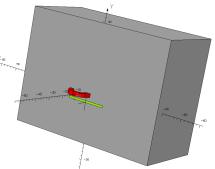
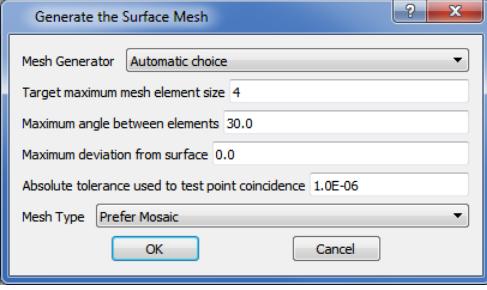
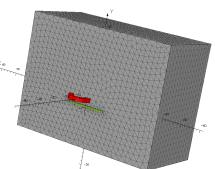
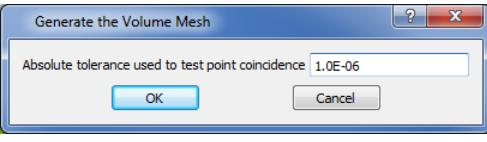
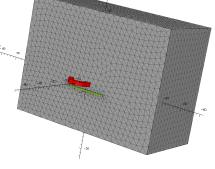
The final task to set up the model is to remove the heat density value set in the first example, as the multiphysics example calculates the losses automatically.

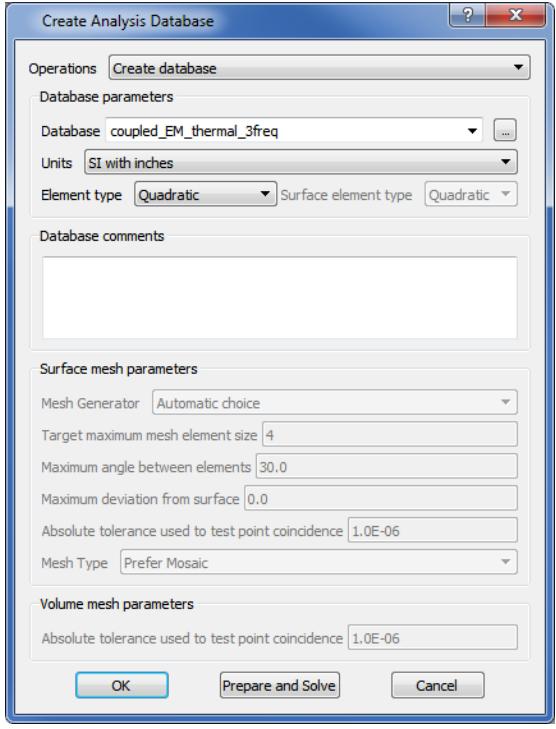


Note that clicking on **None** in the **Heat source definition type** is not the same as clearing. **None** will instruct the multiphysics simulation not to calculate losses in this volume, overriding the automatic calculation.

The multiphysics model is now complete and can be meshed.

## Mesh Generation and Database Creation

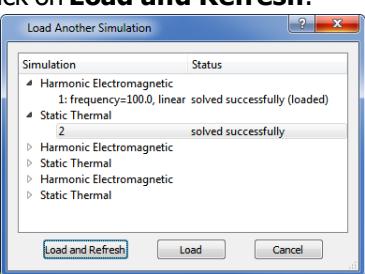
 <b>Create Model Body</b>		
 <b>Generate Surface Mesh</b>	<p>Set the target maximum element size to 4.</p> 	
 <b>Generate Volume Mesh</b>		

 <b>Create Database</b>	<p>Give a database filename, <b><i>coupled_EM_thermal_3freq.op3</i></b>. Note that multiphysics simulations can use <b>Quadratic</b> elements but the edge variable solvers, such as <b>Harmonic EM</b>, only use <b>Linear</b> elements for these stages.</p>  <p>Select <b>Prepare and Solve</b> to start the analysis.</p>
---	--

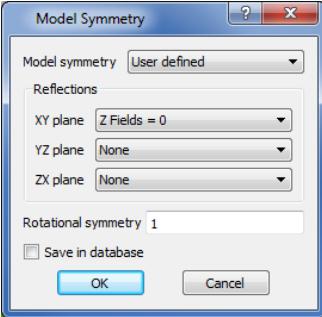
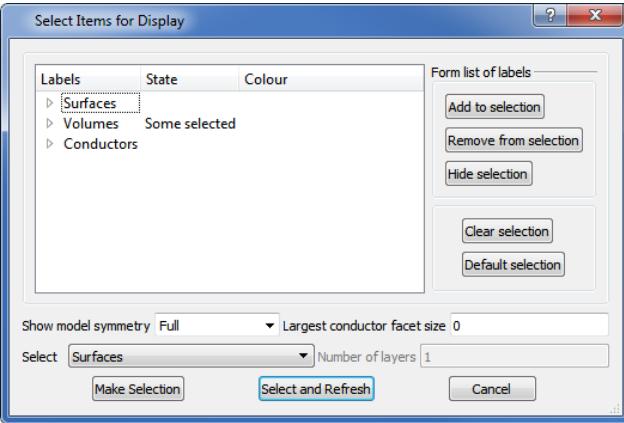
When the six analyses are complete, open the Post-Processor from the Solver window.

## Post-Processing the Coupled Example

Starting the Post-processor from the Solver window will automatically open the first case of the multiphysics simulation - the steady-state analysis at 100 Hz. To examine the thermal solution at this frequency, it is necessary to open the second simulation.

	<p>Choose simulation <b>2</b>. Click on <b>Load and Refresh</b>.</p> 
--	---

The model was set up to exploit the symmetry in the XY plane. In the Post-Processor, it is possible to reconstruct the full model of the plate.

 <b>Model Symmetry</b>	<p>Select <b>Model symmetry</b> as <b>User defined</b> and set <b>XY plane reflections</b> to <b>Z Fields=0</b>.</p> 
 <b>Select</b>	<p>Use <b>Default selection</b> and set <b>Show model symmetry</b> to <b>Full</b> before clicking on <b>Select and Refresh</b>.</p> 

The solution can now be examined.

**3d Display**

Select **Component contours** in **Surface display**. Set the component to **T** to display the new temperature distribution.

Heat flux, temperature gradient and error in heat flux can also be displayed, as for the first model.

To examine the thermal results from the eddy current distributions at the other frequencies, the tool-buttons to load other simulations can be used.

**Select Case**

In the same way that simulation case 2 was selected (see earlier in this section), select simulation case **4** to see the temperature rise at 200 Hz.

Click on **Load and Refresh**

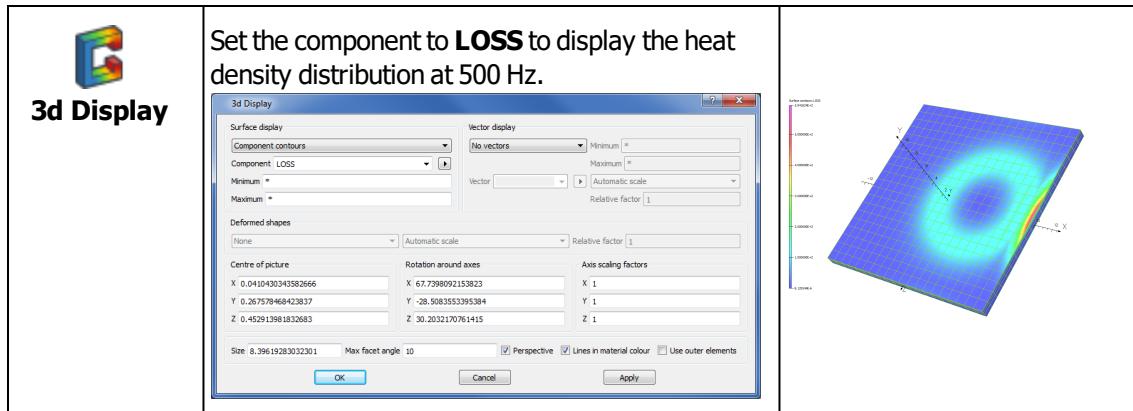
**Load the next case in the database**

**Load the next case in the database**

Pass over simulation case 5 (the steady-state AC solution at 500 Hz) and display results from simulation case 6 (the temperature rise at 500 Hz). This automatically refreshes the 3D display with the same field quantity as was being displayed previously.

The temperature distribution patterns are very similar for all 3 frequencies, although the values are very different. This is because the highest heat density values are confined to the top edge at  $x = 10$ ,  $y = 1$  in all the electromagnetic simulations.

Even though a thermal simulation is loaded, it is also possible to display the electromagnetic simulation results from the corresponding **Harmonic EM** analysis, such as the heat density (**LOSS**), the flux density and the eddy current density.



This completes the thermal and coupled electromagnetic/thermal multiphysics examples.

# **Chapter 9**

## **A Quench Example**

### **Introduction**

---

This example shows the use of the transient thermal analysis program, **Quench**, to determine the temperature distribution in a superconducting coil during a "quench". Quenching is the term applied to the behaviour of a superconductor when it changes from its superconducting to its resistive state. A quench can occur for one or more of the following reasons:

- the temperature of the superconductor becomes too high;
- the superconductor experiences a high magnetic field;
- the current density in the superconductor exceeds its critical value.

Superconducting materials are poor conductors when they are not in the superconducting regime. For this reason, the wire in a superconducting coil is generally composed of superconducting filaments embedded in a metallic conductor. This allows a low electrical resistance path for current that was being carried by the zero resistance superconductor prior to the quench.

In this example, the superconducting magnet consists of two solenoidal coils, connected to an external protection circuit. The temperature rise and quench is initiated by a heat source. Resistive heating will start generating additional heat when the temperature exceeds the critical temperature. The analysis of the quench includes the evaluation of the magnetic field produced by the solenoidal coils. The magnetic field is used to determine the critical temperature.

**Quench** can be used in two ways:

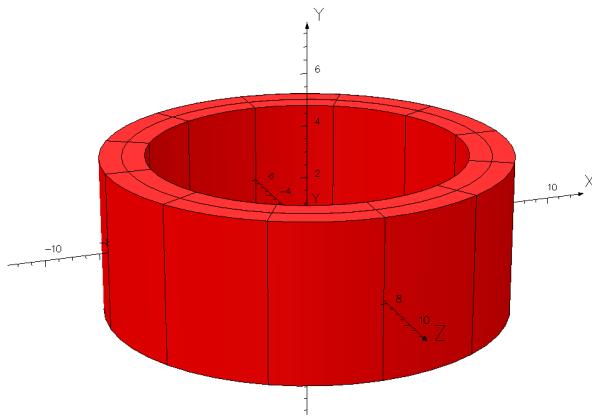
- **Quench Thermal.** The magnetic fields from the conductors are evaluated by the **Quench** solver. The stored energy in the magnetic field and the magnetic coupling between the sections of the coil must be modelled by self and mutual inductances provided by the user.
- **Quench Multiphysics.** The magnetic fields are calculated by the **Transient Electromagnetic** solver running at the same time as **Quench** and therefore can include eddy current effects in the aluminium supporting structure. The two analyses share values of electromagnetic fields and temperature at every time-step.

This example demonstrates the use of the Modeller to build a solenoid coil, specify thermal material properties, wire material properties and boundary conditions and set up a quench circuit. Tabulated

functions are used to define nonlinear material properties. Some detailed considerations for efficient modelling with **Quench** are also discussed. Both types of analyses are described.

All the data in this example are supplied in cgs units.

## The Model



*Figure 9.1 Coil*

Figure 9.1 shows the geometry of the superconducting coil<sup>1</sup>. It has a cross-section of  $1.68 \times 6$  cm and an inner radius of 6 cm. There are 1180 turns of wire with a cross-sectional area of  $0.006 \text{ cm}^2$ .

The model is rotationally symmetric but the flow of heat around the coil reduces the symmetry to reflections in the XY and ZX planes. This symmetry will be exploited in solving only one quarter of the coil, thus reducing the solution time. The heater which causes the quench is modelled by using a Heat Flux boundary condition. The resistive heating is modelled by a Joule heat source calculated from the electrical currents and local resistivity.

The material properties defined for the coil are:

- Nonlinear anisotropic thermal conductivity  
This is represented by three tabulated functions for radial, axial and azimuthal components.
- Nonlinear specific heat capacity  
This is modelled by blending values of the specific heat capacities of NbTi and copper, each of which is defined by a tabulated function.

---

<sup>1</sup>The coil is built from 2 concentric solenoids to control the mesh size and match the connectivity of the circuit.

- Mass density

This is calculated from those of NbTi and copper.

- Electrical conductivity of the wire

This is the conductivity of the wire in its normal state. It is calculated from conductivity of copper and the proportion of copper in the wire (the copper factor).

- Area of wire cross-section

- Critical current

This is calculated from critical current density of NbTi, area of wire cross-section and proportion of superconductor in the wire (the superconductor factor).

The surfaces of the coil are assumed to be perfectly thermally insulating, except where the heater is applied.

## Building the Model

### Building the Solenoid Coil

#### Coil geometry

Because QUENCH is a thermal solver, only solid objects - in this case the solenoid coil volume - are included in the geometry. The surrounding environment is modelled by a thermal boundary condition.

#### Anisotropic mesh

In the early stage of a quench, the front between superconducting material and normal conductor will propagate in the azimuthal direction much faster than in the transverse directions. This is because the thermal conductivity in the direction of current flow is much greater than the thermal conductivity across the conductor. Consequently the temperature rise occurs over very small distances in the axial and radial directions. An anisotropic mesh with much smaller element sizes in the transverse directions must be used to model these extremely steep temperature gradients. If the mesh is not sufficiently fine, numerical noise will be excessive, resulting in non-physical temperature distributions. This noise dies away as the analysis proceeds and has little affect on the solutions later in the transient.

#### Coil conductor

Although QUENCH is a thermal solver, the magnetic field distribution must be calculated, because it affects the critical current density. The solenoid coil is used in three ways:

- as meshed volumes for the thermal analysis;
- as solenoid conductors to compute the field using the Biot-Savart expressions;
- as windings in a circuit for modelling the transient currents.

#### SOLENOID command

The Modeller provides a user-friendly interface for creating a conductor, adding a circuit winding and creating a volume with anisotropic mesh control parameters using a single **SOLENOID** command. A table of parameter values and pictures of the dialogs which issue the **SOLENOID** command follow the description of the parameters.

In the **Create Solenoid Conductor** dialog, input the usual geometric solenoid parameters in the **Solenoid Parameters** tab and then open the **Further Options** tab to define the following parameters:

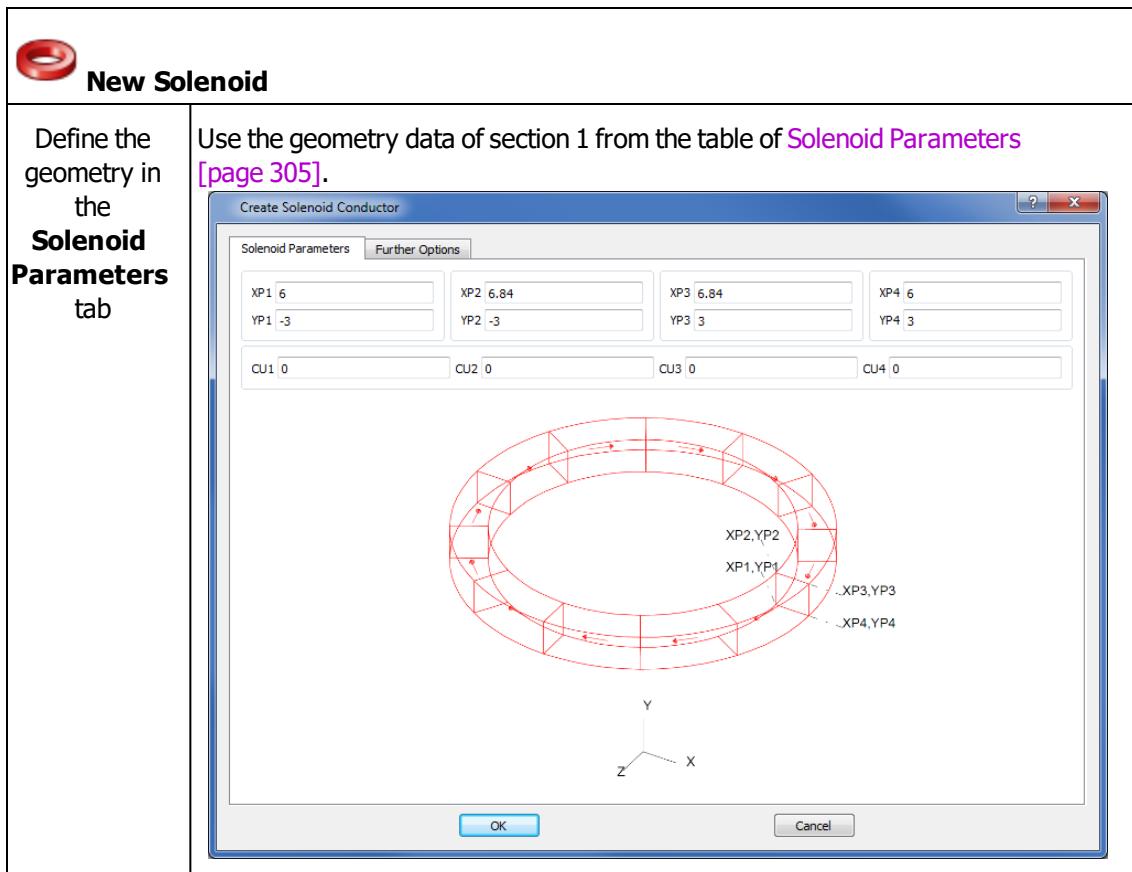
- Under **Source Drives**, check **Circuit element**, which will add a winding to be used in a circuit, and provide a **Circuit element name** which is used to name both the circuit winding and the material label.
- Under **Advanced Options**, set the **Volume Options** to **Create volumes of regular meshes**, which will create a volume cell with the **Circuit element name** as the material label and then provide **Mesh size** and **Anisotropic mesh size factor** which control the anisotropic mesh of the volume cell.  
In the azimuthal direction meshing is controlled by **Mesh size**, and in radial and axial directions by the product of **Mesh size** and **Anisotropic mesh size factor**.

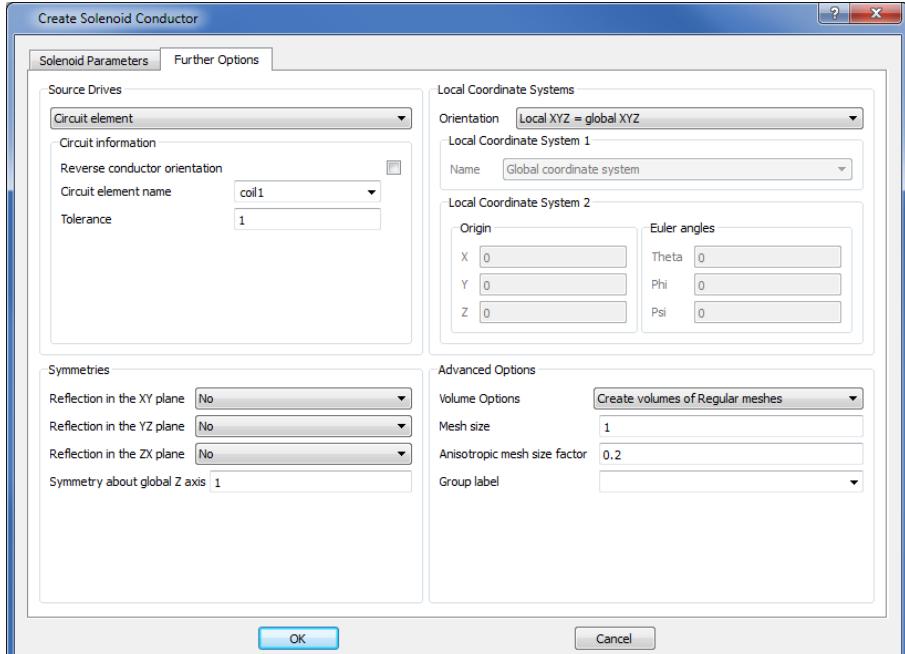
In this example, the coil is split into two sections in the radial direction, to allow the inner and outer sections to have independent protection circuits. The first section ranges from radius 6 cm to 6.84 cm and is named `coil1`. The second section ranges from radius 6.84 cm to 7.68 cm and is named `coil2`. Both sections have the same height, 6 cm, in the Y direction, in a range from -3 cm to 3 cm. To ensure the hexahedral mesh is contiguous across the two connected sections the same mesh size must be used. **Mesh size** 1 cm and **Anisotropic mesh size factor** 0.2 is therefore used for both sections. The parameters are summarized in the following table.

**Solenoid Parameters**

Section	Inside radius	Outside radius	Name	Mesh size	Size factor
1	6	6.84	<code>coil1</code>	1	0.2
2	6.84	7.68	<code>coil2</code>	1	0.2

The first section is created as follows.



<p>Click the <b>Further Options</b> tab.</p>	<p>Select <b>Circuit element</b> and input <b>Circuit element name</b> as <code>coil1</code> and set the <b>Tolerance</b> to 1, which in cgs units is 1 gauss.</p> <p>Select <b>Create volumes of Regular meshes</b>. Input <b>Mesh size</b> as 1 cm and <b>Anisotropic mesh size factor</b> as 0 . 2.</p>  <p>Click on <b>OK</b> to create <b>coil1</b>.</p>
--	--

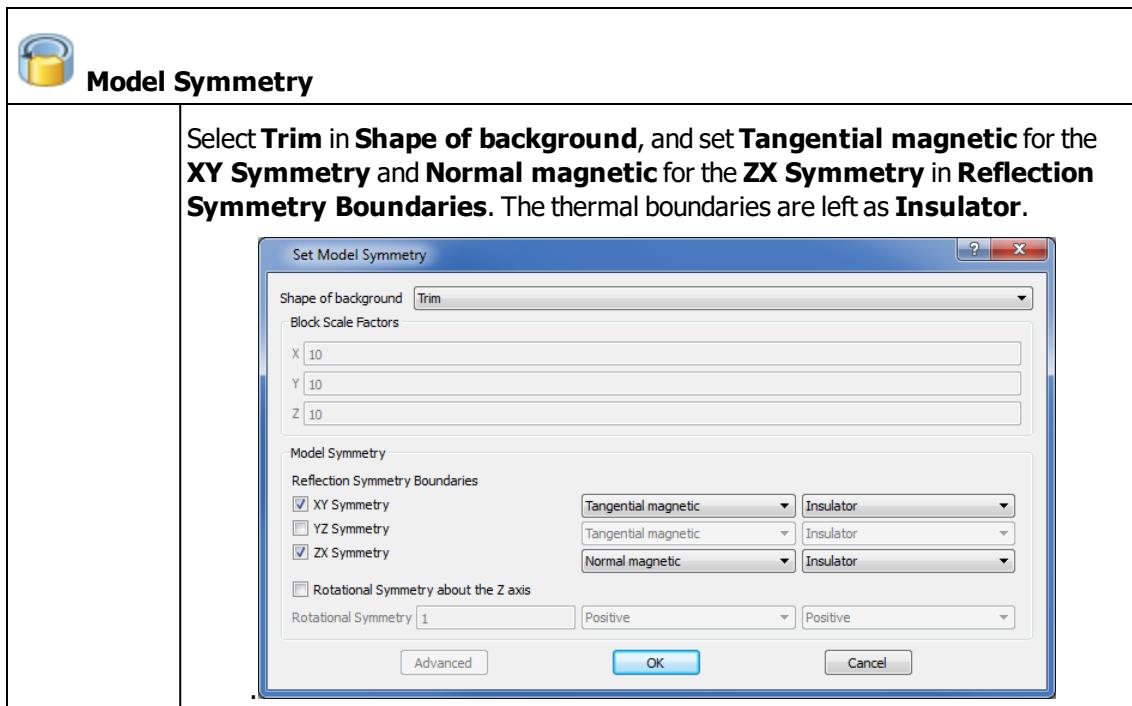
Now create **coil2** using the values in the table of [Solenoid Parameters \[page 305\]](#).

## QUENCH analysis type

Set the **Analysis Type** to be Quench Thermal. This will configure the dialogs of the **Model** menu (symmetry, material properties, boundary conditions etc.) to include the options needed for QUENCH.

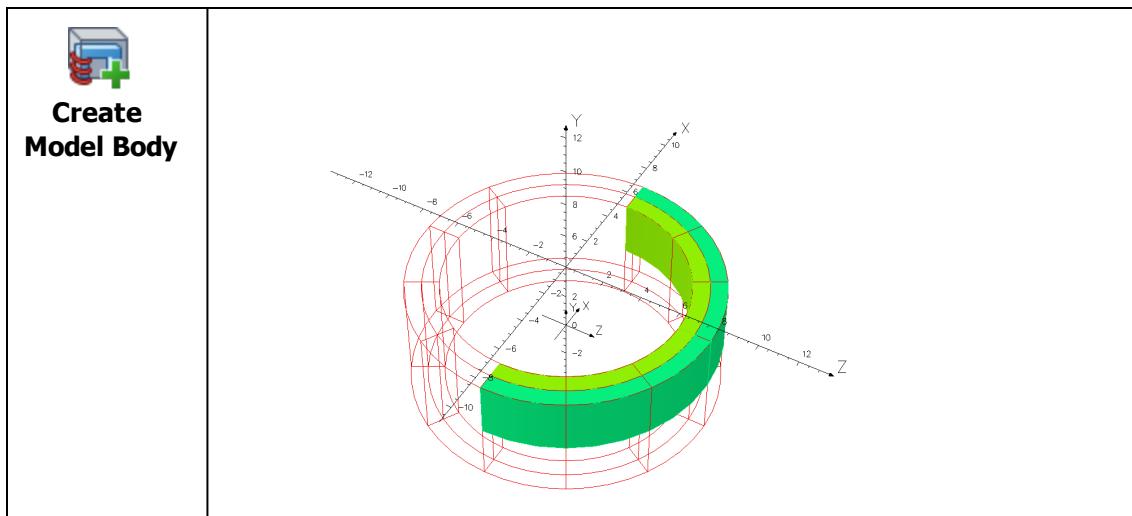
## Symmetry

Because of symmetry, only a quarter of the coil is modelled to reduce the computational costs. Reflections on **XY** and **ZX** coordinate planes are applied. No background volume is needed.



## Model body

Create the model body and view the effect of the symmetry settings. Now only a quarter of the coil remains in the model. The two coils now have material labels of **coil1** and **coil2** so that material properties can be applied to the conductors.



## Material and Volume Properties

Some thermal and electrical properties of the superconducting materials are constant, others can be expressed algebraically and others are highly nonlinear functions of temperature or in some cases temperature and flux density.

### Nonlinear Functions

The data for nonlinear material properties are supplied to the Modeller and hence to QUENCH using [Tabular Functions \[page 576\]](#).

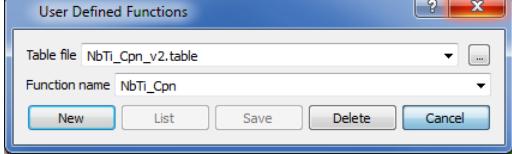
The following table shows a list of the functions required for this example. The file names are the names of table files which hold the data. (The table files are supplied in the examples folder which is installed with Opera<sup>1</sup>.) The function names are used in algebraic expressions in the material property dialogs.

File name	Function	Description
<i>NbTi_Cpn_v2</i>	<b>NbTi_Cpn (T)</b>	Heat capacity of NbTi in normal state
<i>Cu_Cp_v2</i>	<b>Cu_Cp (T)</b>	Heat capacity of copper
<i>Bulk_Kappa_r_v2</i>	<b>Bulk_Kappa_r (T)</b>	Radial thermal conductivity of bulk materials
<i>Bulk_Kappa_z_v2</i>	<b>Bulk_Kappa_z (T)</b>	Axial thermal conductivity of bulk materials
<i>Cu_Kappa_v2</i>	<b>Cu_Kappa (T)</b>	Thermal conductivity of copper
<i>Cu_Sigma_v2</i>	<b>Cu_Sigma (T)</b>	Electrical conductivity of copper
<i>NbTi_Jc_v2</i>	<b>NbTi_Jc (T;B)</b>	Critical current density of NbTi with two arguments, temperature and magnetic field.

In this example, all the necessary **\$FUNCTION** commands can be executed through a **comi** file which has been supplied with the software (see [Command file \[page 311\]](#)). However, to illustrate how to define a function, reading in the first table is shown here as an example.

---

<sup>1</sup>The examples folder can be accessed from the **File** menu of the Opera Manager.

 <b>User Defined Functions</b>	<p>Browse for the table file <code>NbTi_Cpn_v2.table</code> and enter the function name <code>NbTi_Cpn</code> and hit <b>New</b></p>  <p>Once the functions have been defined they can be listed using the <b>List</b> button.</p>
--	--

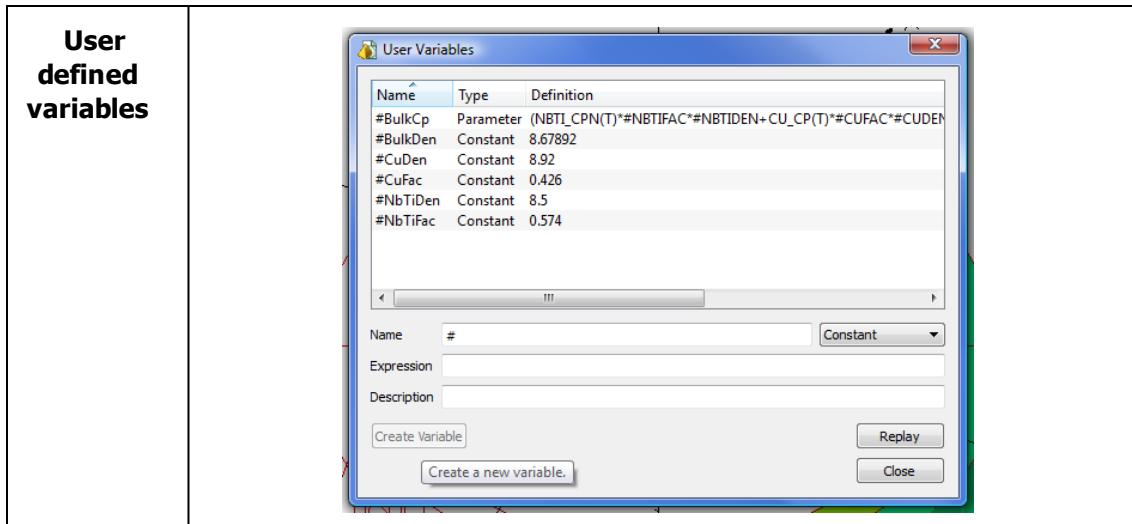
## Setting the User Defined Variables

User variables will be used to define material and volume properties which are constants or can be expressed algebraically. Constant variables are evaluated immediately and only a single value is stored. Parameter variables hold an expression that is re-evaluated every time the variable is used. Parameter variables must be used where the expression contains a varying quantity, such as **T** (temperature) or **TIME**.

The variables that are needed are:

Name	Type	Expression or Value	Description
#NbTiFac	Constant	0.574	NbTi factor
#CuFac	Constant	0.426	Copper factor
#NbTiDen	Constant	8.5	Mass density
#CuDen	Constant	8.92	Mass density
#BulkDen	Constant	#NbTiFac * #NbTiDen + #CuFac * #CuDen	Density of bulk materials
#BulkCp	Parameter	(NbTi_Cpn (T) * #NbTiFac * #NbTiDen + Cu_Cp (T) * #CuFac * #CuDen) / #BulkDen	Heat capacity of bulk materials

All the above variables can be defined using the **comifile** (See "Command file" on the facing page.) but some example definitions of variables using the GUI are shown below. In the **User Variables** dialog, variables are listed as **Model Dimensions, Constants** and **Parameters**. A new variable can be defined by completing the fields below to give the name of the variable (which must begin with a #), the type of variable, the expression for the variable and an optional description.



## Command file

All commands for reading the table files and setting the variables are saved in a command file, ***quench.comi***. Copy this file and all the table files from the **Quench** examples sub-directory to the current project folder. The commands can be executed by running the command file.

 <b>Run Comi File</b>	Browse to the project folder and select the <b>quench.comi</b> file.
---	--

## Setting the Material and Wire Properties

The constants, expressions and functions needed for the properties of the materials and superconducting wire have now been defined. They must now be applied to the materials and wires in the model.

### Thermal conductivity

Nonlinear anisotropic thermal conductivity properties are defined by three expressions based on the functions **Cu\_Kappa(T)**, **Bulk\_Kappa\_r(T)** and **Bulk\_Kappa\_z(T)**.

For the azimuthal thermal conductivity of the bulk material, the conductivity of copper, **Cu\_Kappa(T)**, will be scaled by the copper factor. It is assumed that this is significantly higher than the conductivity of NbTi and dominates the thermal conduction in this direction.

The bulk properties radially and axially are to be taken directly from the tables of measured values. (*Note that the data used in this example are fictitious, but with similar characteristics to real materials.*)

The radial, axial and azimuthal directions of the solenoid coil are automatically transformed to local X, Y and Z directions respectively by the **Volume Options** of the **SOLENOID** command.

## Mass density

The mass density of the bulk material is computed from those of copper and superconductor in the appropriate ratio. Both of them have been defined to be constants in this example, so

```
#BulkDen = #NbTiFac * #NbTiDen + #CuFac * #CuDen.
```

**#BulkDen** could be defined as a nonlinear property using tabulated functions if the density varied significantly with temperature.

## Specific heat capacity

The nonlinear heat capacity of the bulk material is specified by the user-defined variable, **#BulkCp**, which has already been defined using the two tabulated functions, **Cu\_Cpn (T)** and **NbTi\_Cp (T)**, i.e.

```
#BulkCp = (NbTi_Cpn(T) * #NbTiFac * #NbTiDen + Cu_Cp(T) * #CuFac * #CuDen) / #BulkDen
```

## Electrical conductivity

The nonlinear electrical conductivity of the wire when operating in the normal regime will be specified by the tabulated function and user-defined variable, **Cu\_Sigma (T) \* #Cufac**.

## Wire cross-section

The wire has a cross-sectional area of 0.006 cm<sup>2</sup>.

## Critical current

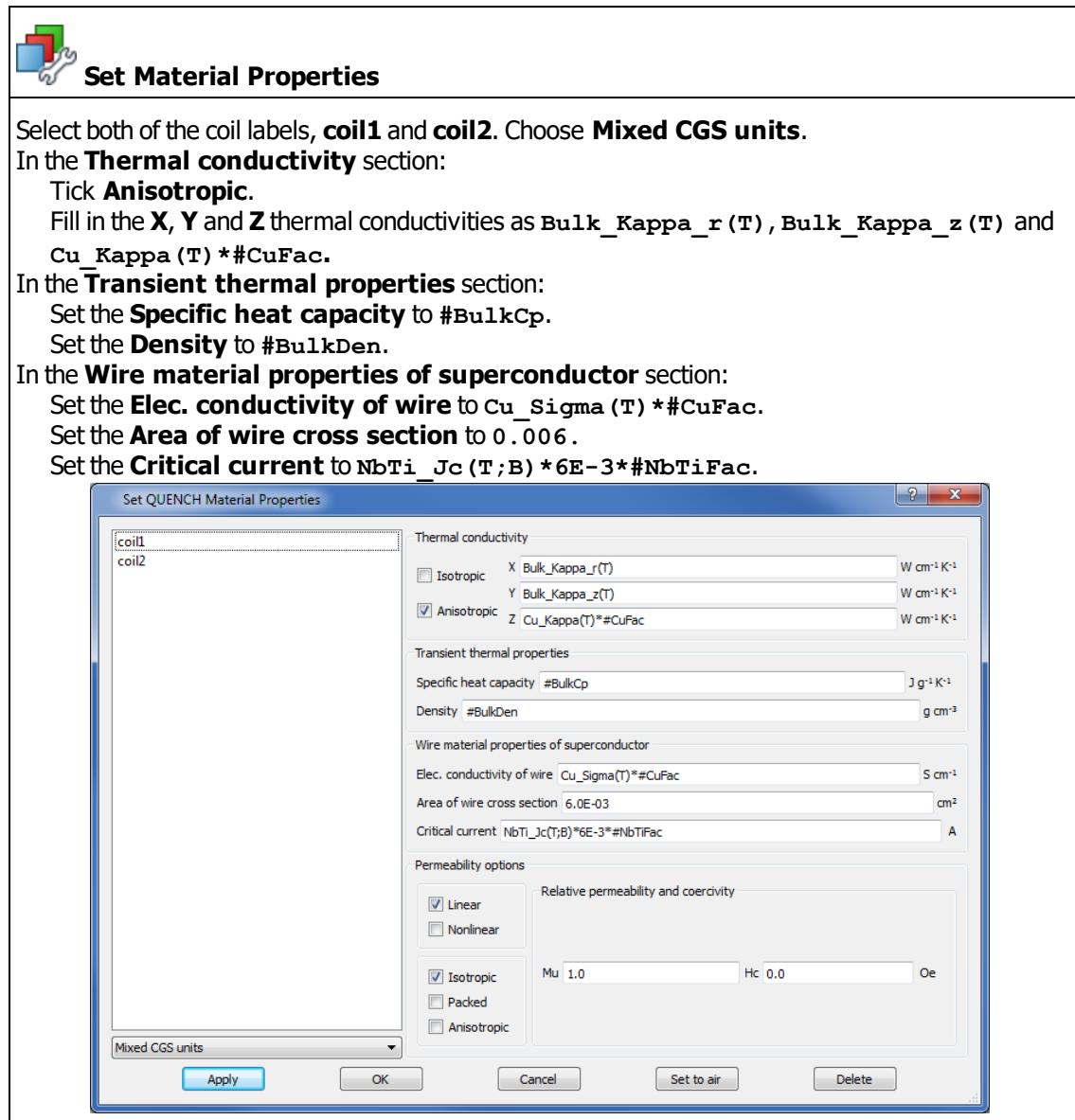
The critical current is used to decide whether the conductor material is in the superconducting or normal state. It will be specified by the two-argument tabulated function of critical current density in NbTi, the area of wire cross-section in CGS units and the user-defined superconductor factor, i.e.

```
NbTi_Jc(T;B) * 6E-3 * #NbTiFac
```

where T and B are the temperature and the flux density of the magnetic field at the evaluation points. The magnetic field is generated by the currents through the solenoid conductors.

## Set material properties

The material and wire properties will be set in one dialog.



Notes:

1. When defining material properties using expressions, those expressions must be defined in database units. The unit options in the **Material Properties** dialog only apply to numerical values.
2. The data in a table file that defines a user defined function includes unit information so that functions can be converted to the correct units during analysis.

3. The database units are only defined when the analysis database is created.

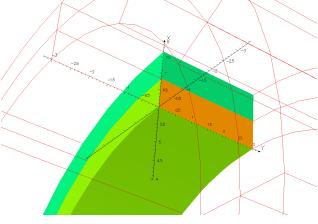
## Setting the Boundary Conditions

### Quench heat source

The quench heat source is modelled by a **Heat Flux** boundary condition. In this example the boundary condition is defined using the **RANGE (A ; B ; C)** function.

The **RANGE** function gives a value of 1 when the value of first argument is between values of the second and third arguments; otherwise **RANGE** returns zero. In this case three **RANGE** functions are multiplied together in order to limit the extent of the heat source by time (between 0 and 0.1) and position (X between 6 and 6.2; Y between 0 and 0.2).

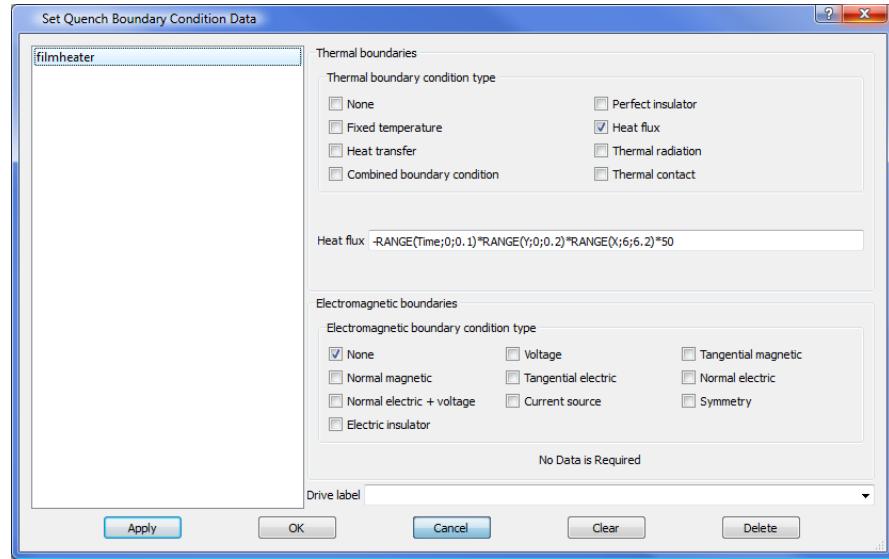
The heat source applies 2 W for 0.1 s at the start of the analysis to initiate the quench (50 W/cm<sup>2</sup> over 0.04 cm<sup>2</sup>). A negative value is used to imply heat entering the coil.

 <b>Pick Faces</b>	<p>Pick the innermost face of the symmetry surface (positive x). Using the right mouse button brings up the context menu and allows you to go straight into <b>Face Properties</b> for this face. The <b>Boundary condition label</b> is <b>filmheater</b>.</p> <p>Because this face exists only after the Model Body has been created an information box warns that this change will be lost when returning to component view.</p>	
 <b>Set Boundary Conditions</b>		

Select the **filmheater** face label, then tick **Heat flux** option.

Set the value of **Heat flux** to

**-RANGE (X ; 6 ; 6 . 2) \*RANGE (Y ; 0 ; 0 . 2) \*RANGE (Time ; 0 ; 0 . 1) \*50.**



## Defining the Circuit

---

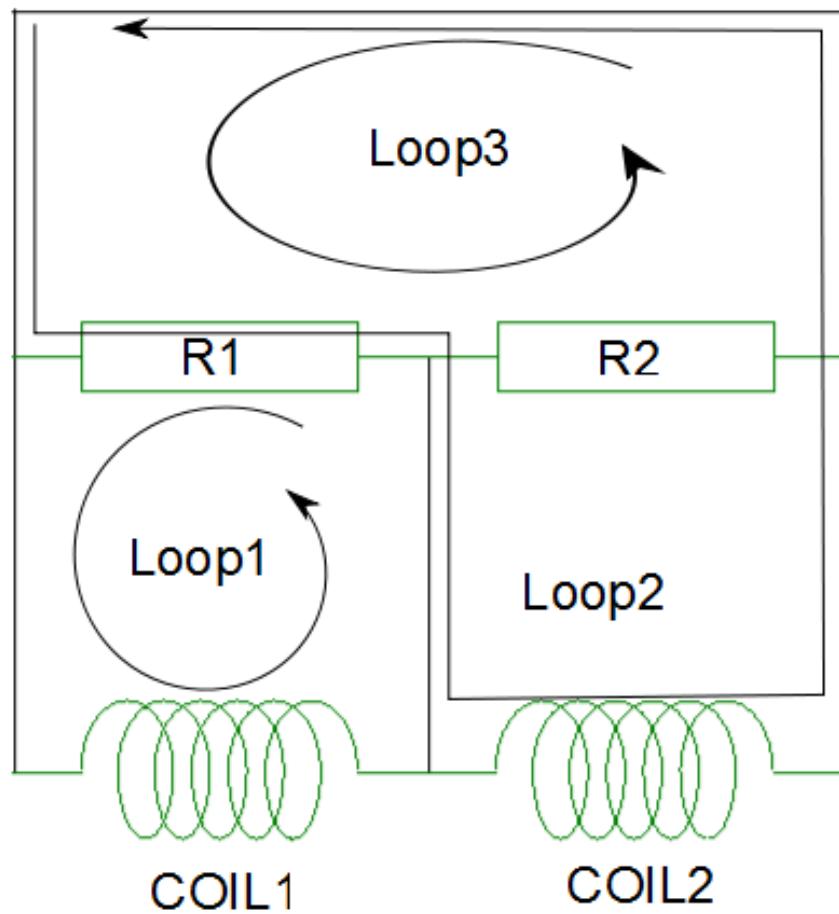
A circuit must be included to model the transient current process. The component names of the two parts of the winding (**coil1** and **coil2**) have been created by the **SOLENOID** command when building the coil. Further work is needed to complete the circuit.

This first part of the example uses a stand-alone QUENCH analysis. Therefore the stored energy in the magnetic field and the magnetic coupling between the sections of the coil must be modelled by self and mutual inductances provided by the user. Also, initial currents of the windings (prior to the quench) should be set.

During analysis the resistance of the two sections of the coil will be calculated from the resistance of the respective wire materials. Therefore the winding name and the name of the respective material must match each other. The matching has to be done when a section of the coil and its respective winding are created by the **SOLENOID** command. As only a quarter of the coil is modelled, the symmetry of the windings is adjusted, based on the model symmetry, to scale the wire material resistance to the resistance of the whole winding.

More circuit elements, used as protection resistors, are also included in the quench circuit.

The quench circuit and circuit parameters for this example are shown in the following diagram and table.

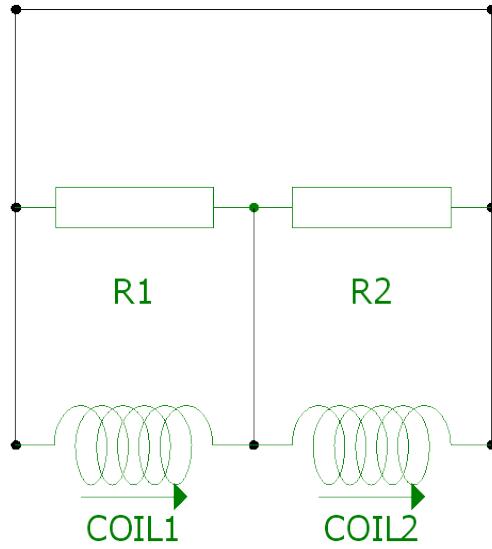


Name	Resistance	Turns	Symmetry	Inductance	Initial Current
R1	1 ohm				
R2	1 ohm				
coil1		590	4		200 A
coil2		590	4		unset
coil1:coil1				0.0443 H	
coil2:coil2				0.05356 H	
coil1:coil2				0.04206 H	

## Notes:

1. Before the quench starts, the circuit will force the current in coil1 and coil2 to be the same. It is only necessary to set the **Initial Current** in one coil.
2. **Inductance** between windings is used to represent self or mutual inductance. For example, the inductance between winding coil1 and itself represents the self inductance of coil1, while the mutual coupling between winding coil1 and coil2 represents the mutual inductance.

For more information on how to use the Circuit Editor, see [Circuits in Harmonic and Transient EM simulations \[page 651\]](#).

 <b>Define Circuit</b>	<p>Use the <b>Opera-3d Modeller Circuit Editor</b> to draw the following circuit. Components can be dragged into the circuit window from the <b>Component Explorer</b> and then values entered for each component in the <b>Component Property</b> panel. Use the values from the table above to set the protection resistor values and coil turns and currents.</p>  <p>The <b>Inductance</b> is entered by going to <b>Edit -&gt; Set Mutual Inductance</b> and entering the self and mutual inductance terms, again from the table above.</p> <p style="text-align: right;"><b>opera</b> simulation software</p>
---	--

In a complete specification, every initial current should be set. However, in preparing data for QUENCH, it is allowed that most initial currents of series windings are left unset (empty) as long as at least one of them is set. The QUENCH solver will assign the unset currents as it solves the circuit under DC conditions at the initial time using the initial currents that have been set.

## Quench Analysis

---

### Setting up the Quench Analysis

During quench analysis, nonlinear material properties are updated at every time-step. If **Nonlinear Materials** is selected, additional nonlinear iterations will be used within each time-step. In most cases the adaptive time-stepping routines are sufficiently sensitive to the changing material properties and will adapt the time-step in such a way as to remove the need for additional nonlinear updates. Although a nonlinear analysis will certainly give improved accuracy the additional computing time might not justify this. For models that do not contain nonlinear permeable materials it is usually sufficient to use **Update nonlinear properties before each time-step**.

In the **Analysis Settings** dialog add a **Quench Thermal** analysis and set the following parameters:

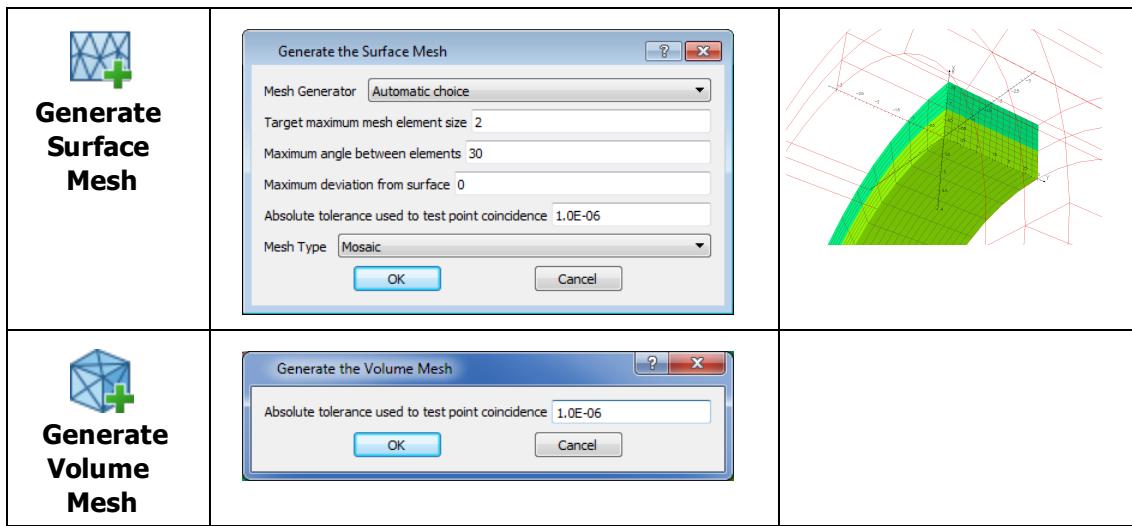
Property	Value
<b>Initial temperature</b>	4.2 [K]
<b>Material options</b>	<b>Update nonlinear properties before each time-step</b>
<b>Time-stepping option</b>	<b>Adaptive</b>
<b>Initial time-step</b>	<b>1E-04 [s]</b>
<b>Maximum error</b>	<b>0.5 [%]</b>

Add the following output points to the Output times list: 0.01, 0.02, 0.05, 0.1, 0.2 and the following Logging variables: **Tmax**, **R\_coil1**, **R\_coil2**, **coil1\_I** and **coil2\_I**.

For this non-coupled model the fields are set to come only from the **Biot-Savart Conductors**.

### Meshing

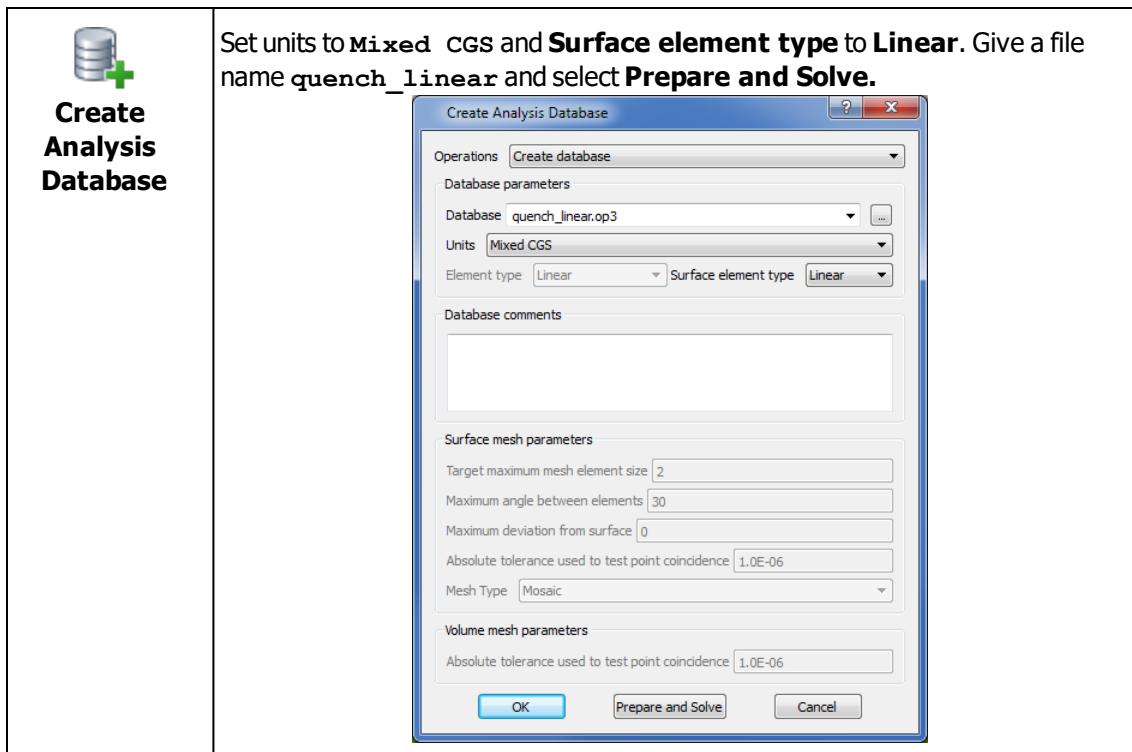
Create the surface mesh with a maximum element size of 2 and set the **Mesh Type** to **Mosaic**.



## Database

Create a database file to be solved.

In most Quench models it is only necessary to use linear elements, as the quench wave front temperature profile is so steep that the second-order function of quadratic elements is still not sufficient to match this and so the extra time to solve a model with these type of elements will not give more accurate solutions than with simple first-order linear elements.



## Saving the model

When creating the database file (**\*.op3**) the software will also save the model file (**\*.opc**) with the same name.

## Running the Analysis

Using the **Prepare and Solve** option will place the analysis into the Batch Processor in the Opera Manager and if the batch queue is set to solve immediately the analysis will start straight away.

Note that a warning will be issued when preparing the simulation, saying that the **INSULATOR** boundary condition will be ignored. This refers to the **INSULATOR** condition which was automatically created by the **Model Symmetry** operation (see [Symmetry \[page 307\]](#)) but is coincident with the user supplied **filmheater** boundary condition label and its **HEATFLUX** boundary condition (see [Quench heat source \[page 315\]](#)).

## Post-Processing the Quench Analysis

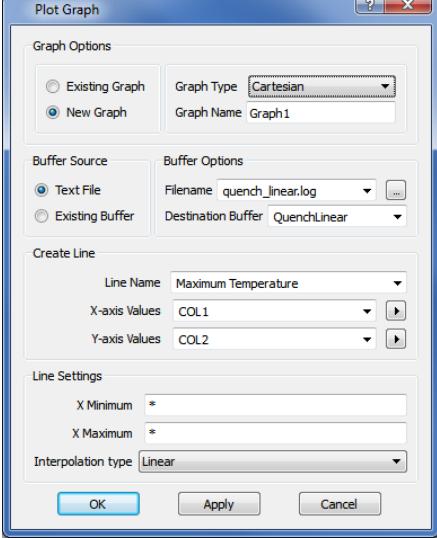
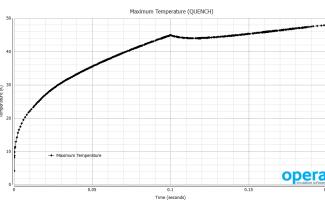
Start the Post-Processor from the Modeller or the Opera Manager or, if the analysis is complete, from the solver window.

### Launch Post-Processor

If the Post-Processor was launched from the solver window, press **Load and Refresh** to load the selected time-step.

### Log of Analysis

The variables that were set up for logging (See "Setting up the Quench Analysis" on page 320.) can be examined to gain an overview of the progress of the analysis.

 <b>Plot Graph</b>	<p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <b>Graph1</b>. In <b>Buffer Source</b>, select <b>Text File</b>.</p> <p>Set the <b>Filename</b> to be <b>quench_linear.log</b> and give a name for the Destination Buffer, for example <b>QuenchLinear</b>.</p> <p>In the <b>Create Line</b> section, set the <b>Line Name</b> to <b>Maximum Temperature</b>. For the <b>X-axis Values</b>, type <b>COL1</b>, and for the <b>Y-axis Values</b>, type <b>COL2</b>. Confirm with <b>OK</b>.</p> 	
--	--	---

The other logged variables can be graphed by changing the **Y-axis Values** to **COL3**, **COL4** etc. for column 3 (**R\_coil1**), column 4 (**R\_coil2**) etc. Axes and curves can be annotated and multiple curves drawn on the same graph (see [Figure 9.2](#) for an example).

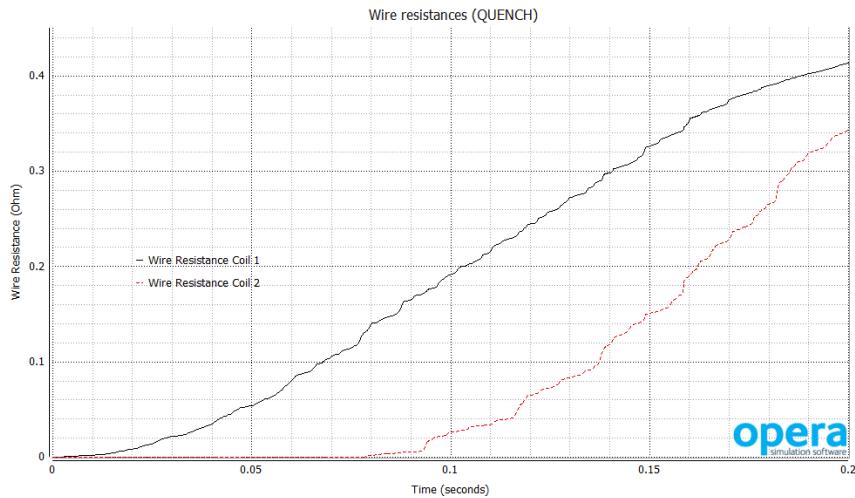


Figure 9.2 Wire resistances of Coil 1 and Coil 2

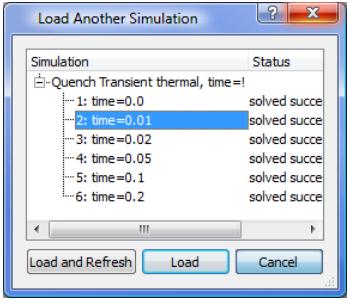
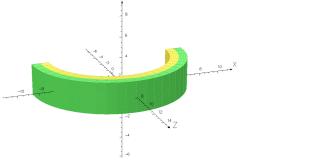
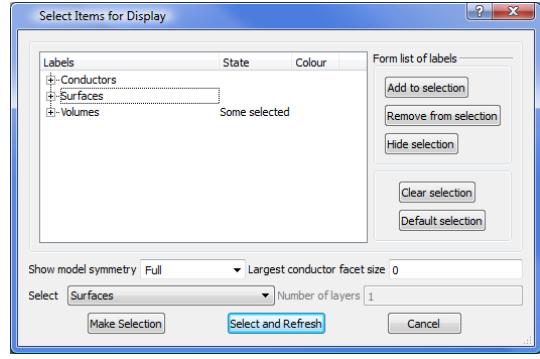
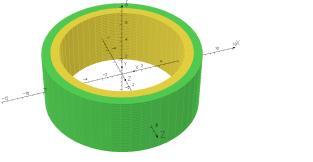
The red curve represents  $R_{\text{coil}2}$ , which is the resistance of the second coil. It stays at zero (i.e. superconducting) until  $t = 0.08\text{s}$  when coil 2 starts to quench as well.

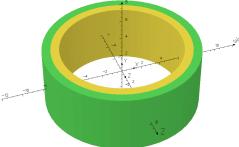
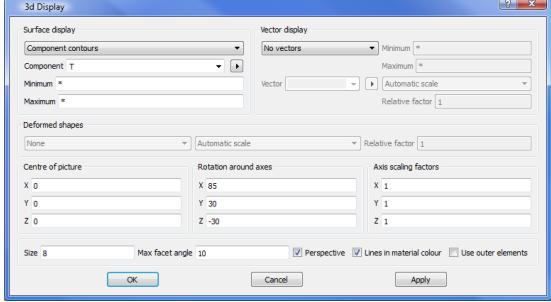
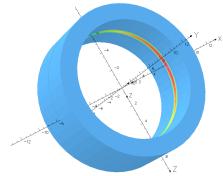
## Thermal Results

The QUENCH analysis solves for the temperature at each node in the finite element mesh. Other results available include the heat flow and the temperature gradient, both of which are vector quantities. The estimated numerical error in the heat flow is also computed. QUENCH models can be examined in the same way as other analysis **op3** files.

### Temperature

The temperature distributions at the output times are stored as individual simulations. Load a particular simulation to view the temperature at a certain time.

 <b>Open (activate and load)</b>	<p>Select the solution database, <b><i>quench_linear.op3</i></b> and then choose the second simulation from the list, the simulation at time=0.01s.</p> 	
 <b>Default Select and Refresh</b>	<p>For QUENCH analyses all materials will be displayed by default.</p>	
 <b>Select</b>	<p>To switch on the display of the complete model select <b>Full</b> model symmetry and press <b>Select &amp; Refresh</b>.</p> 	

 <b>Click Outline view of model</b>	<p>Switch off the mesh to get a clearer view.</p>	
 <b>3d Display</b>	<p><b>Change Material colours to the Component contours for the Surface display.</b> The temperature can be displayed by setting <b>Field component</b> to <b>T</b>, using the pull-right menu.</p> 	

As would be expected, the temperature rise is propagating along the azimuthal direction at this early stage.

Open other time steps using **Load the next case in the database**  to view the temperature distributions at later times.

## Advanced Quench Control

Within the Quench analysis it is possible to add more advanced controls. This is done by using an additional command file to redefine the values of user variables at every time-step based on the current solution state.

All system variables, such as **TIME**, circuit currents and voltages and the resistance of wire materials, are accessible by the command file. This mechanism allows users to set up a number of advanced controls.

As an example, switching off a fixed heat source at  $t = 0.1\text{s}$  can be achieved with the following commands:

```
$IF TIME<=0.1
  $CONSTANT #HEATSOURCE 2
$ELSE
  $CONSTANT #HEATSOURCE 0
$END IF
```

In the model, the user must apply a **Volume label** to the required cell in the **Cell Properties** dia-

log and set the **Volume label** to be a heat source with a value of **#HEATSOURCE** in the  **Volume Properties** dialog.

The command file name is related to the name of the **op3** database being solved, i.e. for a database file **filename.op3**, the related command file should be named **filename\_quench.com**.

For more information on this see the **Opera-3d Reference Manual**, Analysis Programs, "Transient Analysis Control".

## Coupled Quench Analysis

The second part of this example extends the model that has already been built and shows how to set up a coupled analysis that allows the inclusion of conducting or permeable materials. Now, rather than just calculating the temperatures and magnetic fields from the conductors, it is possible to include the fields from magnetization and eddy currents.

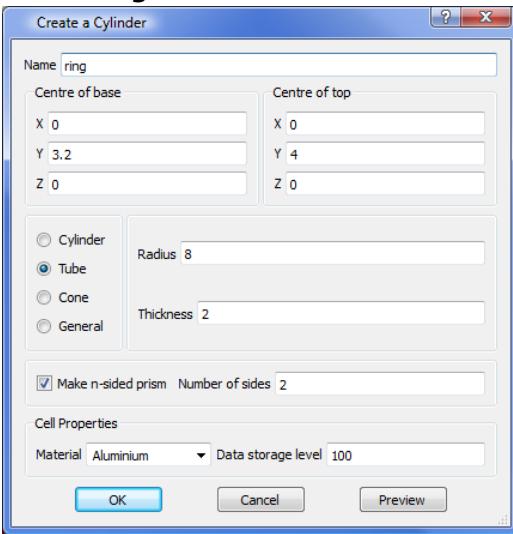
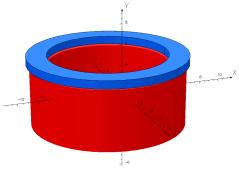
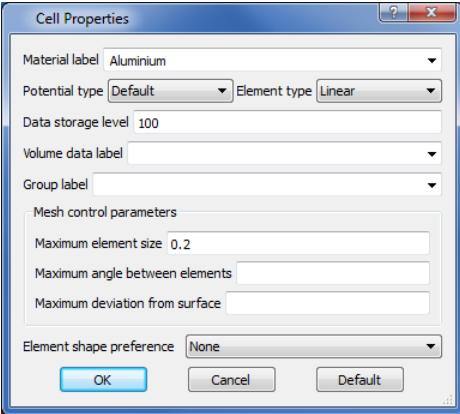
Because the model is now being solved, in part, with a transient electromagnetic solver it is not necessary to include all of the inductive coupling information as it was in the first example. The coupling between the coils and the eddy current and magnetic materials is now an integral part of the overall field solution.

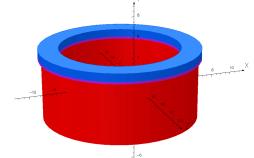
In this example a pair of end rings made from aluminium are included in the model along with a disk of insulator between the coil and the ring. Only one ring is actually included in the model as the reflection planes that are already in place will imply the existence of the second ring.

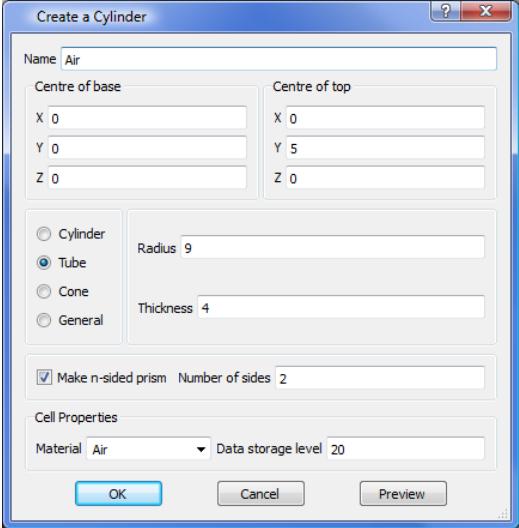
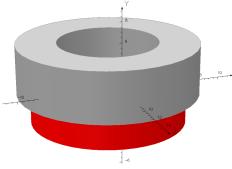
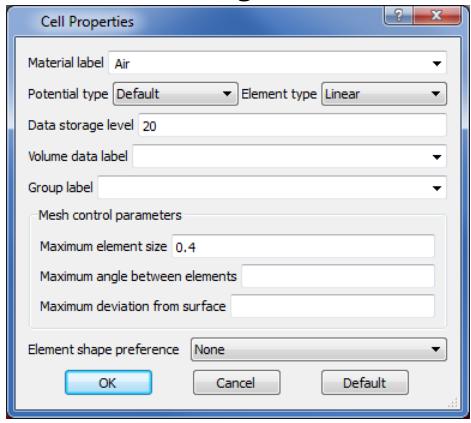
As the inclusion of additional materials requires that the field is computed not only within the coils themselves, using the Biot-Savart integral, but also in the space around this additional ring of material, it is necessary to include a volume of meshed air space around the model. This air space has to extend sufficiently far away from the system to ensure that the far field boundary conditions are correctly applied.

### Modify the Original Model

The original model is loaded back into the Modeller so that it can be modified to include the aluminium ring and the air region around the coils and ring. The air region helps to control the mesh size around the coils to improve the mesh quality and the accuracy of the results as well as grading the mesh out to the background region.

	Open the original model file <b>quench_linear.opc</b> and delete the model body using <b>Model -&gt; Delete Model Body</b>	
	<p>Set the <b>Tube</b> option with base at <math>(0, 3.2, 0)</math> and top at <math>(0, 4, 0)</math>, radius 8 and thickness 2. The first part of the ring's properties can also be set with a <b>Material</b> label of <b>Aluminium</b> and a <b>Data storage level</b> of 100.</p>  	
	Right-click on the ring and set the cell properties. Set the <b>Maximum element size</b> to 0.2. 	

 <b>Create Cylinder</b>	Again, set the <b>Tube</b> option but this time with base at $(0, 3, 0)$ top at $(0, 3.2, 0)$ radius 8, thickness 2 <b>Material</b> label of <b>Insulator</b> and <b>Data storage level</b> of 50.	
 <b>Pick Cells</b>	Right-click on the insulator and set the cell properties. Set the <b>Maximum element size</b> to 0.2.	

 <b>Create Cylinder</b>	<p>Create the air region around the coils and ring. A <b>Tube</b>, from <math>(0, 0, 0)</math> to <math>(0, 5, 0)</math> with a <b>Radius</b> of 9, a <b>Thickness</b> of 4.</p> 	
 <b>Pick Cells</b>	<p>Right-click on the cylinder of air and set the <b>Maximum element size</b> to 0 . 4 and the <b>Data storage level</b> to 20.</p> 	

## Setting up the Coupled Solution

Even in a coupled solution the settings for **Quench** and **Transient Electromagnetic** can be defined independently. For example, the **Transient Electromagnetic** analysis in this example could be run with **Use nonlinear properties** to model the nonlinear permeable materials, while **Quench** is running with **Force linear properties** to reduce the solution time.

Most of the settings remain the same as for the linear solution. The only change is to compute the magnetic field in the **Transient Electromagnetic** solver.

The **Transient Electromagnetic** settings to be used can now be set in the same dialog . For this example, since there are no nonlinear permeable materials, the default settings, using linear materials, can be accepted. This is done by default in a Quench Coupled simulation, so the option is not shown.

## Setting the Additional Material Properties

### Electrical conductivity

The nonlinear electrical conductivity of the aluminium ring will be specified using the reciprocal of the tabulated function for the resistivity, **Al6061T6\_Res (T)** .

### Specific heat capacity

The nonlinear heat capacity of the aluminium ring will be specified by the tabulated function, **Al6061T6\_Cp (T)** .

### Thermal conductivity

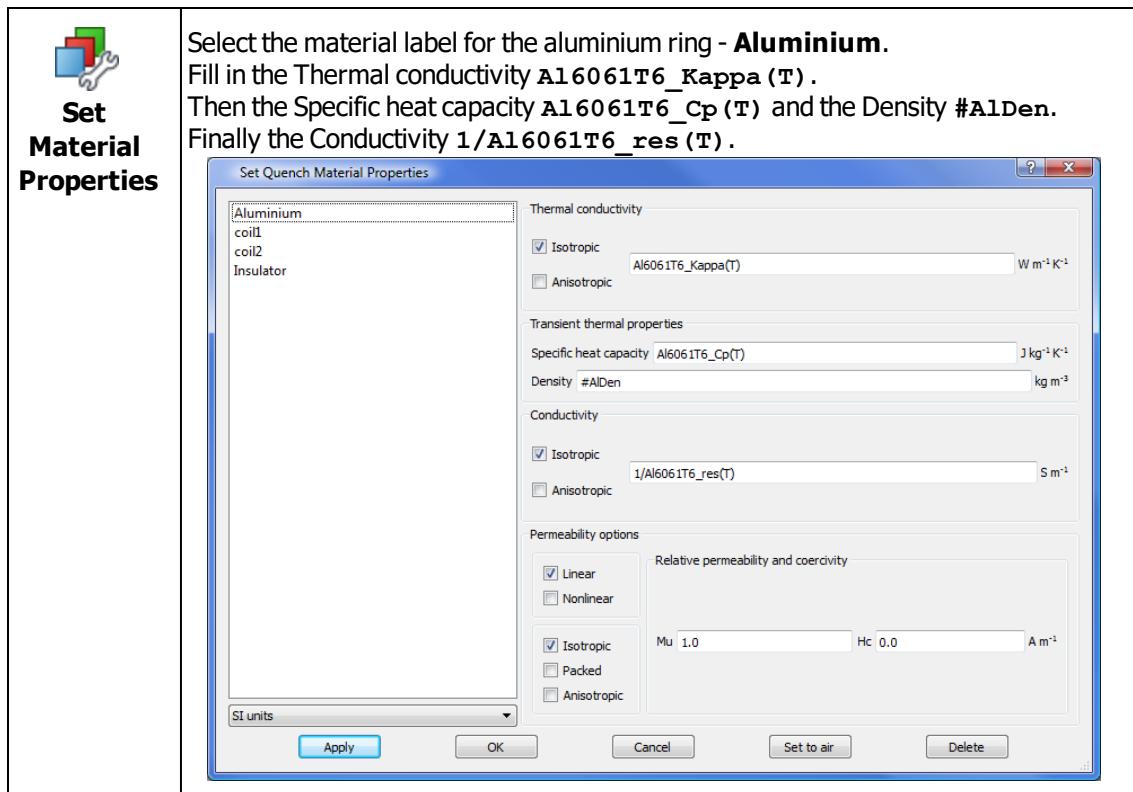
The nonlinear thermal conductivity of the aluminium ring will be specified by the tabulated function, **Al6061T6\_Kappa (T)** .

### Mass density

The aluminium ring is assumed to have a temperature independent density so a constant value will be used, **#AlDen**.

### Command file

All of these quantities, as well as the previously used functions and variables should be loaded as before using the supplied command file, **Quench2.comi**.



## Simplified Circuit Setting

With the coupled solution it is not necessary to include any inductive coupling information about the coils, as this will be taken care of as a fundamental part of the field solution. The information that has been already included from the first part of this example will therefore be ignored for the coupled solution.

## Setting up the Analysis

### Advanced time-step control

The adaptive time-stepping algorithm is a complex system that ensures that an accurate solution is computed by careful adjustment of the time-step between each solution, based upon the changing fields and material properties. With coupled solutions it is possible that the time-step may become too small due to the large changes in the extremely nonlinear material properties in a Quench solution.

The user can avoid this by setting up a special user variable `#VF_ETRTSSERV` which links the **Transient Electromagnetic** time-step to the **Quench** time-step. This can greatly speed up a coupled

solution as it avoids the **Transient Electromagnetic** solution using very small time-steps as the material properties are updated.

Setting `#VF_ETRTSSERV=1` will switch on this feature.

Some care is required in using `#VF_ETRTSSERV`: in the situation when the thermal problem has nearly reached equilibrium, but the electromagnetic model is still changing, the time-step may be increased too much and the solution accuracy may suffer. To avoid this a reasonable maximum time-step can be set up by setting `#Timestep_max`. The exact value will depend on the systems time constant, in this model a value of 0.1 s is reasonable.

## Model symmetry



Open the **Model Symmetry** dialog and change the **Shape of background** to **Block** with the following scaling factors: x=3, y=6, z=3. Keep the existing symmetry boundaries.

## Model body

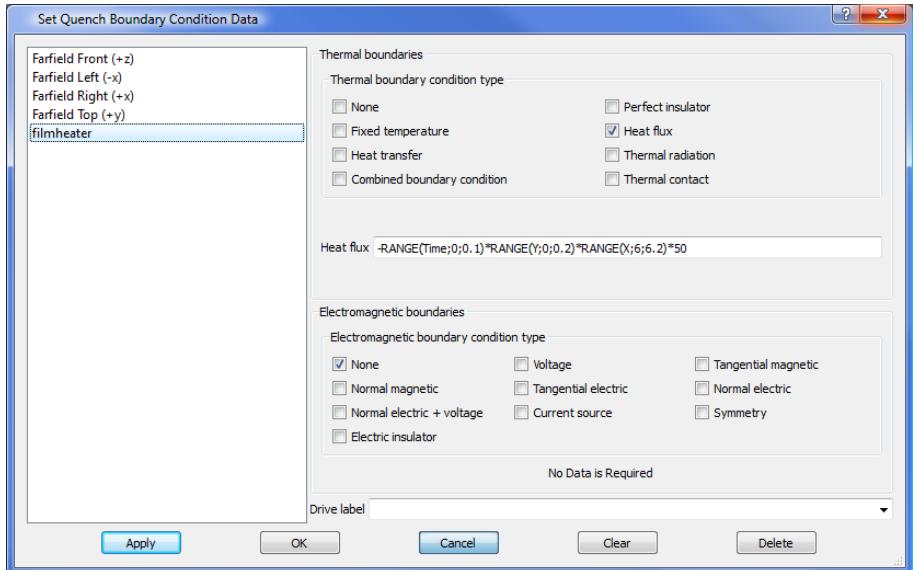


**Create Model Body**

## Quench heat source

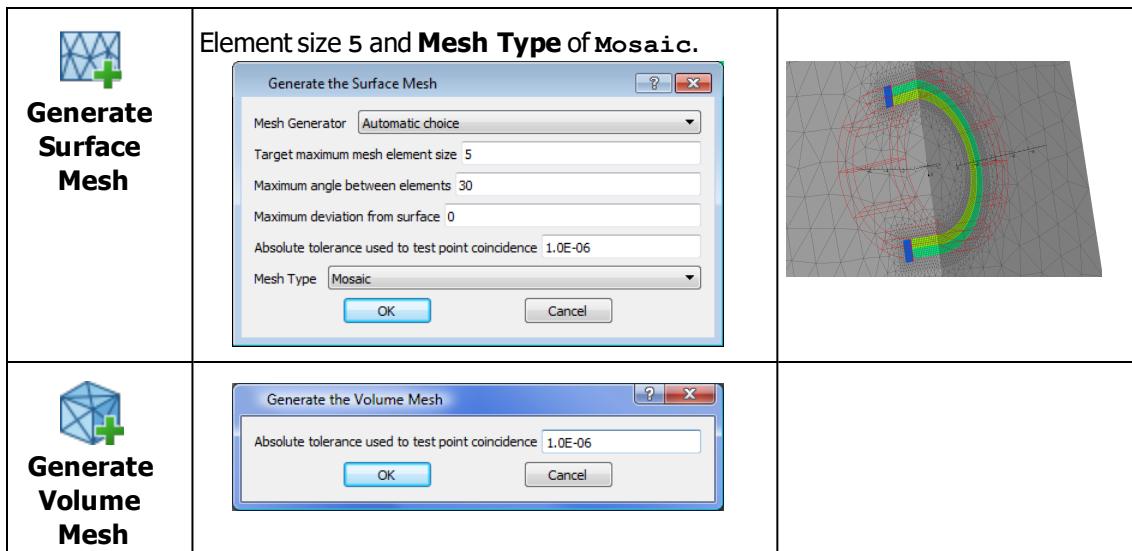
In the **Quench Thermal** example above, the quench heat source was defined after creating the Model Body. Consequently this data has been lost and must be applied again.

For more information about the heat source, see [Quench heat source \[page 315\]](#).

	<p>Pick the innermost face on the symmetry surface. Using the right mouse button brings up the context menu and allows you to go straight into <b>Face Properties</b> for this face.</p> <p>Set the <b>Boundary condition label</b> to <b>filmheater</b>.</p> <p>Because this face only exists after the Model Body has been created an Information box warns that this change will be lost when returning to component view.</p>	
 <b>Boundary Conditions</b>	<p>Select the <b>filmHeater</b> face label, then tick <b>Heat flux</b> option.</p> <p>Set the value of <b>Heat Flux</b> to  <math display="block">-\text{RANGE}(\text{Time}; 0; 0.1) * \text{RANGE}(Y; 0; 0.2) * \text{RANGE}(X; 6; 6.2) * 50.</math>  </p>	

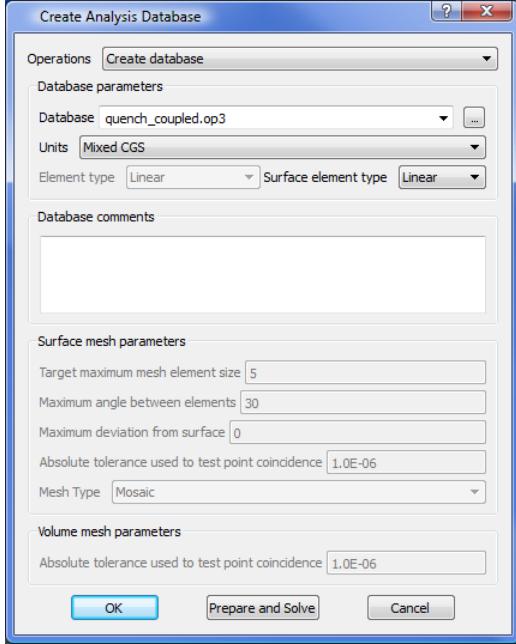
## Generate the mesh

Set a larger maximum element size which will be used for the background region; the finer meshing on the inner model regions is controlled by the individual cell properties.



## Database

When the database is created there will be two ***op3*** files created, one for the **Quench** solver, which has the name given by the user, and one for the **Transient Electromagnetic** solver, which has the name given by the user with the addition of ***\_ETR*** to identify it as the **Transient Electromagnetic** database.

 <b>Create Analysis Database</b>	<p>As before, select <b>Mixed CGS</b> units and <b>Linear</b> elements. Name the database <b>quench_coupled</b> and select <b>Prepare and Solve</b>.</p>  <p>As two databases will be created the information boxes for both will appear one after another as the two databases are built.</p>
--	---

## Running the Analysis

When the Opera Manager batch queue starts a **Quench Multiphysics** analysis, it will start both analyses. While the analyses are running the information about fields and material properties is passed between the solvers at each time-step and a control file is used to keep the two solutions synchronized.

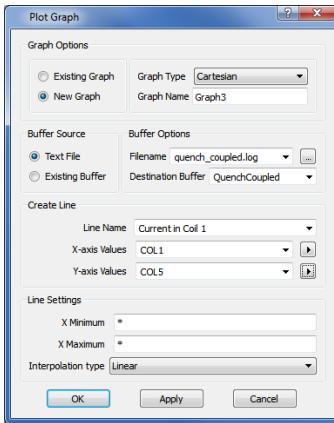
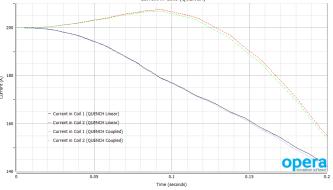
As both solutions need to run concurrently, it is advisable to run them on a machine with more than one processor or multiple processor cores and with sufficient physical memory (RAM) to ensure that both solutions are kept in RAM and do not swap to disk (virtual memory). It might be necessary to adjust the Opera Manager Batch Processor Options to allow more than one solver at the same time.

## Post-Processing the Coupled QUENCH Solution

Start the Post-Processor from the Modeller or the Opera Manager or, if the analysis is complete, the solver window.

### Log of Analysis

The variables that were set up for logging (see [Setting up the Quench Analysis \[page 320\]](#)) can be examined to gain an overview of the progress of the analysis. By overlaying the results from the coupled and the non-coupled models the influence of the aluminium rings can be investigated.

 <b>Plot Graph</b>	<p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <b>Graph3</b>. In <b>Buffer Source</b>, select <b>Text File</b>.</p> <p>Set the <b>Filename</b> to be <b>quench_coupled.log</b> and give a name for the Destination Buffer, for example <b>QuenchCoupled</b>.</p> <p>In the <b>Create Line</b> section, set the <b>Line Name</b> to <b>Current in Coil 1</b>. For the <b>X-axis Values</b>, type <b>COL1</b>, and for the <b>Y-axis Values</b>, type <b>COL5</b>. Click on <b>OK</b> to plot the current in coil1 against time.</p> 	
---	---	---

Repeat with **COL6** and select **Existing Graph** to add the current from coil2.

Compare the currents in the non-coupled coils as well by changing the log file to `quench_linear.log` and adding the current curves from this file as well. Figure 9.3 shows a graph with all 4 lines.

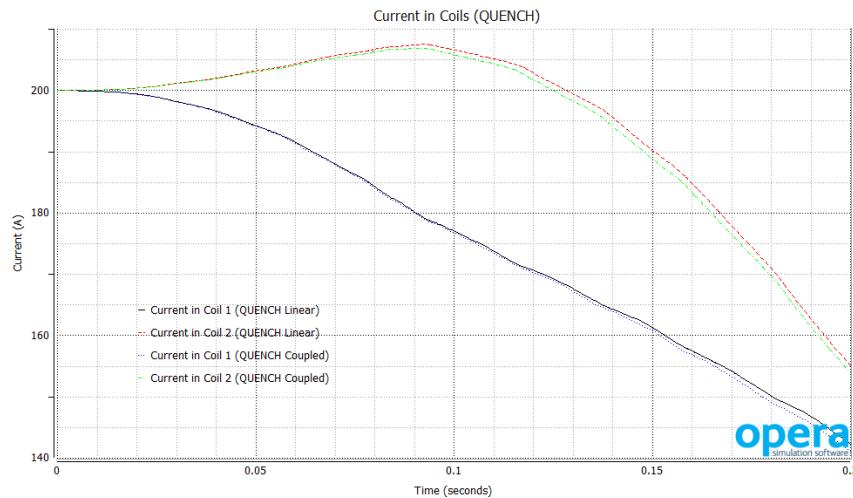
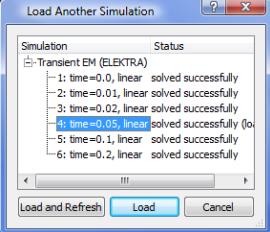
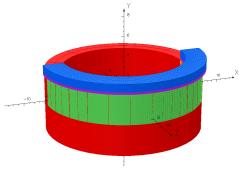
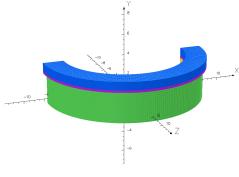


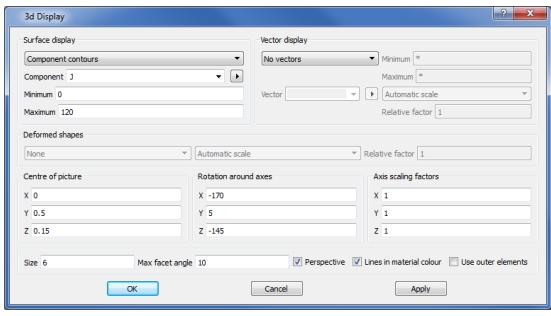
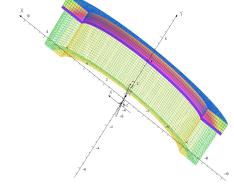
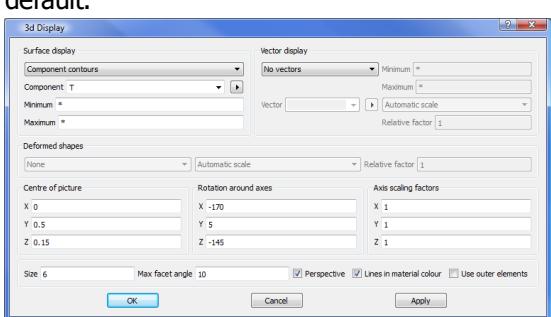
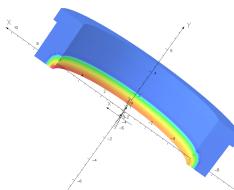
Figure 9.3 Current in Coil 1 and Coil 2

## Thermal and Eddy Current Results

The thermal and eddy current results are both stored in the ELEKTRA/TR database file. For easy of viewing, only the quarter model will be displayed. The rest of the model can be switched on, as

before, from the **Select** dialog, .

 <b>Open (activate and load)</b>	<p>Select the solution database, <b><i>quench_coupled_ETR.op3</i></b> and then choose the fourth simulation from the list, the simulation at time=0.05s.</p> 	
 <b>Default Select and Refresh</b>	<p>For QUENCH analyses all materials will be displayed by default.</p>	
 <b>Hide the Conductors</b>	<p>Hide the conductors so that only the materials of the coils and the aluminium ring are displayed.</p>	

 <b>3d Display</b>	<p>Display the surface contours of <math>\mathbf{J}</math> to display the eddy currents in the aluminium ring. Set the maximum limit to 120.</p> 	
 <b>3d Display</b>	<p>Temperatures can be viewed by setting the <b>Component</b> to <math>\mathbf{T}</math>. Set the display limits back to default.</p> 	

By changing the selection of which materials are being displayed the temperature rise, due to eddy currents, in the aluminium ring can also be investigated. As the insulator between the coil and the ring has no thermal conductivity there will be no thermal conduction between the coil and the ring. Coupling can be modelled by entering a constant or functional thermal conductivity for the insulator.

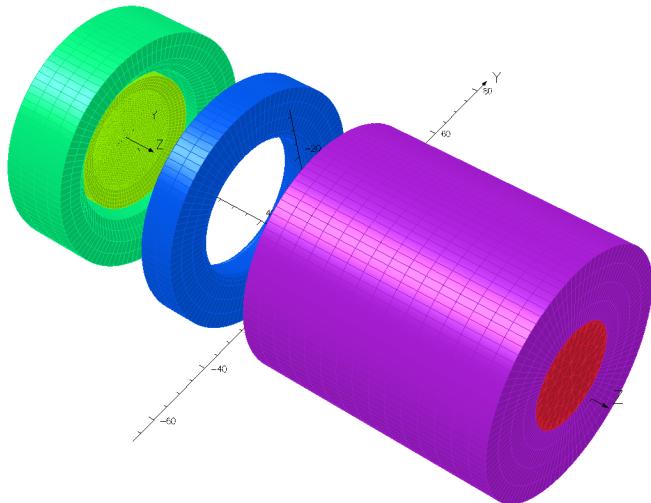
# **Chapter 10**

## **A Charged Particle Example**

### **Introduction**

---

An electron gun with a curved cathode is described in this example. The complete gun is shown in Figure 10.1; the symmetric one quarter model is shown in Figure 10.2. The features demonstrated in this example include: exploiting symmetry and the specification of emitters using the menu facilities in the Modeller.



*Figure 10.1 The complete 3d model of the electron gun*

The device is modelled using Mosaic meshing with tetrahedral, hexahedral, prismatic and pyramidal elements. Calculation of the emitted electron current depends on having an accurate solution close to the cathode surface. The electric field varies rapidly in a direction normal to the cathode surface when space charge effects are involved. These space charge effects limit the current that can be extracted. An additional cell is defined to enable mesh size control just outside the surface of the cathode.

A sequence of solutions is calculated for this model. Initially only the thermionic cathode is included; a secondary particle emitter is then added to the anode; finally the effects of dielectric charging are estimated by coupling the space charge solution to a lossy dielectric simulation.

When the primary electron beam collides with the anode surface, secondary particles are produced. These will include back-scattered incident particles, true secondary electrons and a few ionized atoms from the anode surface. In this example true secondary production is not important because the electrons with low energy are swept back into the anode.

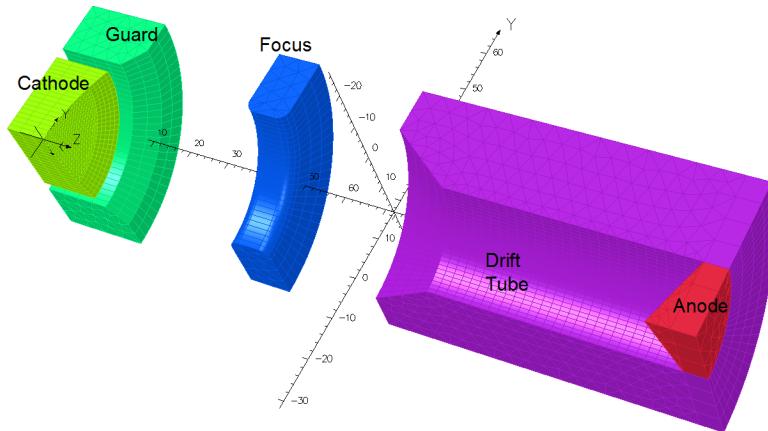


Figure 10.2 The symmetric one quarter model of the electron gun

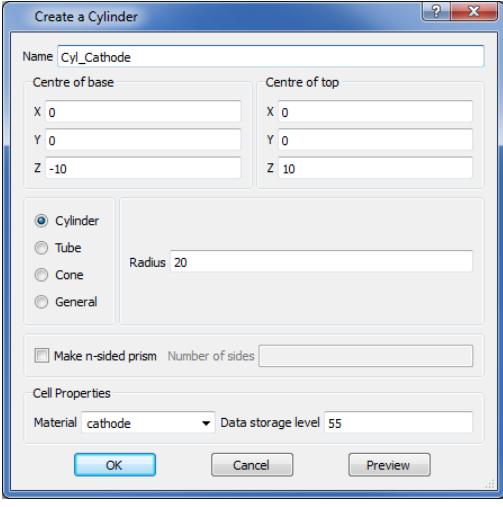
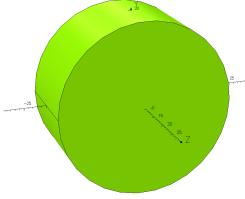
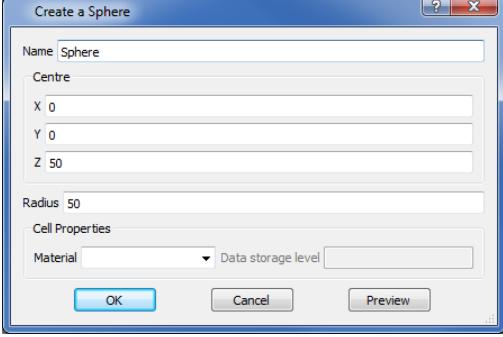
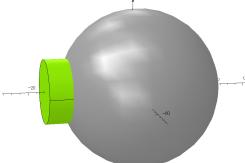
The lossy dielectric simulation calculates the currents of particles arriving on material surfaces. It uses this information to predict a modified voltage distribution caused by these currents; the beam trajectories may also be modified by this change in the voltage distribution, hence an iterative solution is required.

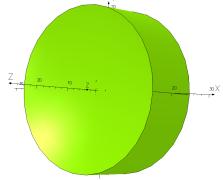
## Constructing the Model

### Building the Geometric Model

#### Cathode

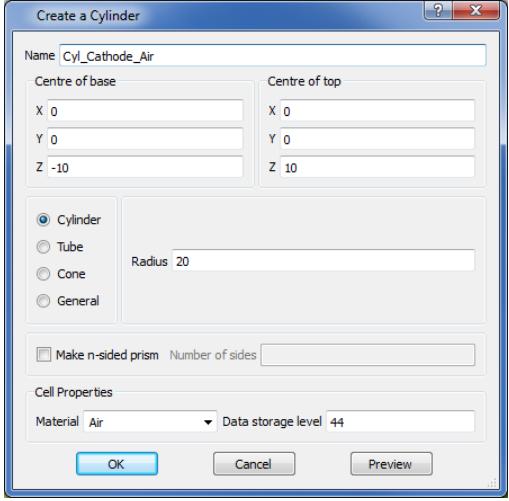
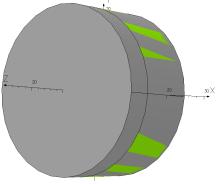
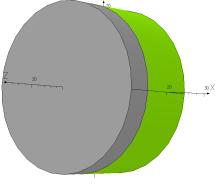
Construct the cathode from a cylinder, base  $(0, 0, -10)$ , top  $(0, 0, 10)$ , radius 20, material name **cathode** and a data storage level of 55.

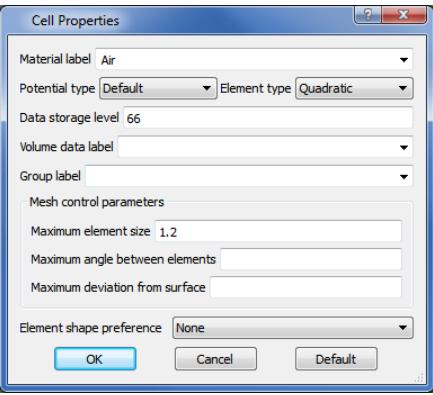
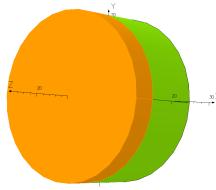
 <b>Create Cylinder</b>	 <p><b>Create a Cylinder</b></p> <p>Name: Cyl_Cathode</p> <p>Centre of base:</p> <ul style="list-style-type: none"><li>X: 0</li><li>Y: 0</li><li>Z: -10</li></ul> <p>Centre of top:</p> <ul style="list-style-type: none"><li>X: 0</li><li>Y: 0</li><li>Z: 10</li></ul> <p><input checked="" type="radio"/> Cylinder    <input type="radio"/> Tube    <input type="radio"/> Cone    <input type="radio"/> General</p> <p>Radius: 20</p> <p><input type="checkbox"/> Make n-sided prism   Number of sides: [ ]</p> <p>Cell Properties:</p> <p>Material: cathode   Data storage level: 55</p> <p>OK   Cancel   Preview</p>	
 <b>Create Sphere</b>	<p>Create a sphere, centred at <math>(0, 0, 50)</math>, radius 50. Leave the material name empty.</p>  <p><b>Create a Sphere</b></p> <p>Name: Sphere</p> <p>Centre:</p> <ul style="list-style-type: none"><li>X: 0</li><li>Y: 0</li><li>Z: 50</li></ul> <p>Radius: 50</p> <p>Cell Properties:</p> <p>Material: [ ]   Data storage level: [ ]</p> <p>OK   Cancel   Preview</p>	

 <b>Pick Bodies</b>	Select the cylinder followed by the sphere as bodies. Right-click on the bodies and perform a <b>Combine bodies -&gt; Subtraction, with regularization</b> operation. This sequence cuts away a spherical cap from the cylinder that will form the cathode.	
---	---	---

### Mesh control volume

Small quadratic elements are required close to the emitter surface. Therefore a second cylinder is added so that the mesh size can be specified.

 <b>Create Cylinder</b>	<p>Create a cylinder, base <math>(0, 0, -10)</math>, top <math>(0, 0, 10)</math>, radius 20, material name <b>air</b> and a data storage level of <b>44</b>.</p>  <p>Note that the cathode and cylinder are overlapping, shown by the indistinct colours.</p>	
 <b>Pick Bodies</b>  <b>Pick Entity</b>	<p>Select the cylinder and then the cathode, as bodies. (Pressing the space bar will allow you to select the cathode while it is hidden by the air.) Right-click and select <b>Combine bodies -&gt; Trim Overlap, with regularization</b> from the context menu. This sequence will uniquely resolve the overlap, forming two bodies and two cells: the cathode and a free-space cell for mesh size control.</p>	

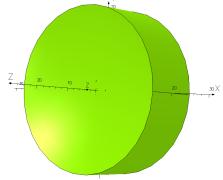
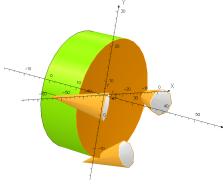
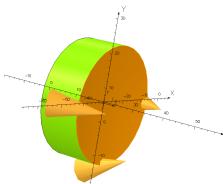
 <b>Pick Cells</b>	<p>Right-click on the air and select <b>Cell properties</b> from the context menu. Specify a mesh size of 1.2, a <b>Data storage level</b> of 66, and set the <b>Element type</b> to be <b>Quadratic</b>.</p> 	
--	---	---

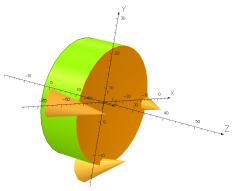
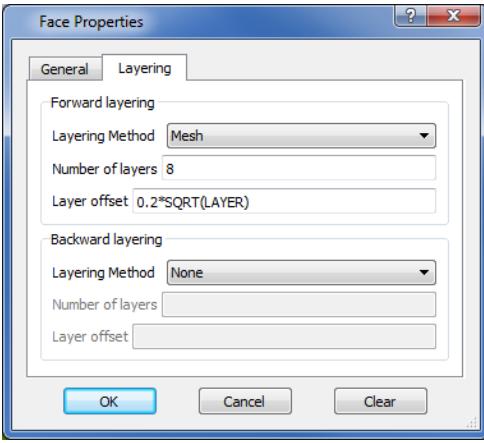
## Mesh layering

In addition to the measures above (defining an air volume with quadratic elements and a mesh size of 1.2 length units), a layered mesh in front of the emitter will be implemented.

To define layering, a face has to be picked, and the number of layers and the direction of the layering has to be specified. Here it is important to understand the concept of surface directions.

Faces have an orientation (normal direction); in general the face normal points outwards from a cell. However this is not guaranteed when a face is created by a boolean operation such as subtraction or a trim operation as in this example. If in doubt, the face direction can be displayed and changed if needed.

 <b>Pick Bodies</b>  <b>Hide Entity</b>	<p>Double-click on the air in front of the emitter - this will make the emitter surface accessible for the following steps.</p>													
 <b>Pick Faces</b>  <b>Pick Entity</b>	<p>Pick the curved surface of the cathode by pointing at it and double-clicking. Then use  <b>Vectors</b>.</p> <p><b>Vectors</b> dialog box:</p> <table border="1"> <tr> <td><b>Vector component</b></td> </tr> <tr> <td><input checked="" type="radio"/> Face normal (picked faces only) <input type="radio"/> Volume orientation</td> </tr> <tr> <td><input type="radio"/> Conductor current direction</td> </tr> <tr> <td><input type="radio"/> Current density</td> </tr> <tr> <td><input type="radio"/> Body force</td> </tr> <tr> <td><input type="radio"/> Vector potential</td> </tr> <tr> <td><input type="radio"/> Electric field</td> </tr> <tr> <td><input type="radio"/> Displacement</td> </tr> <tr> <td><input type="radio"/> None</td> </tr> <tr> <td>Scale factor for the vectors: 5</td> </tr> <tr> <td><input checked="" type="checkbox"/> Display vector magnitude</td> </tr> <tr> <td><b>OK</b>   <b>Cancel</b>   <b>Apply</b></td> </tr> </table> <p>Select the <b>Face normal</b> option and a <b>Scale Factor</b> of 5, then press <b>OK</b> to display the normal vectors.</p>	<b>Vector component</b>	<input checked="" type="radio"/> Face normal (picked faces only) <input type="radio"/> Volume orientation	<input type="radio"/> Conductor current direction	<input type="radio"/> Current density	<input type="radio"/> Body force	<input type="radio"/> Vector potential	<input type="radio"/> Electric field	<input type="radio"/> Displacement	<input type="radio"/> None	Scale factor for the vectors: 5	<input checked="" type="checkbox"/> Display vector magnitude	<b>OK</b> <b>Cancel</b> <b>Apply</b>	
<b>Vector component</b>														
<input checked="" type="radio"/> Face normal (picked faces only) <input type="radio"/> Volume orientation														
<input type="radio"/> Conductor current direction														
<input type="radio"/> Current density														
<input type="radio"/> Body force														
<input type="radio"/> Vector potential														
<input type="radio"/> Electric field														
<input type="radio"/> Displacement														
<input type="radio"/> None														
Scale factor for the vectors: 5														
<input checked="" type="checkbox"/> Display vector magnitude														
<b>OK</b> <b>Cancel</b> <b>Apply</b>														
	<p>If the face normals are pointing into the cathode cell as shown above right, point at one of the vectors and double-click using the left mouse button. This will reverse the normal direction.</p>													

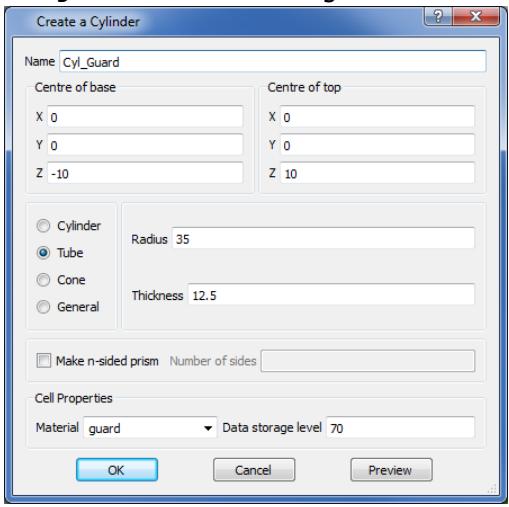
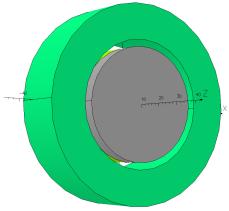
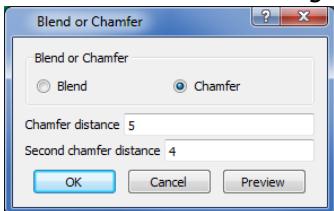
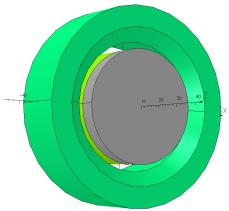
	<p>Right-click on the highlighted face, select <b>Face Properties</b> then choose the <b>Layering</b> tab.</p>	
	<p>Referring to the directions defined by the face normals, set <b>Forward Layering</b> to <b>Mesh</b>, the <b>Number of Layers</b> to 8, and the <b>Layer Offset</b> to <math>0.2 * \text{SQRT}(\text{LAYER})</math>.</p> 	

For more information on layering, see "The FACEDATA Command" in the *Opera-3d Reference Manual*.

After the face properties have been set, the default display can be restored by pressing  **Default Selection**.

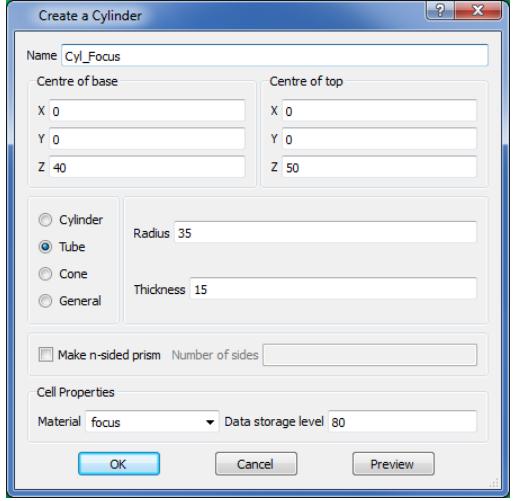
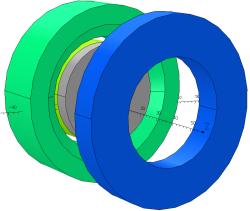
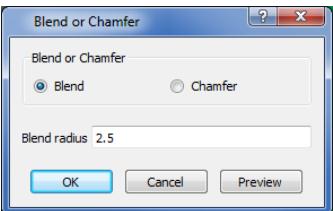
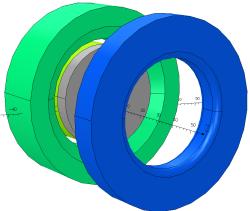
## Guard ring

Construct a guard ring around the cathode as follows:

 <b>Create Cylinder</b>	<p>Create a tube, base <math>(0, 0, -10)</math>, top <math>(0, 0, 10)</math>, radius 35, thickness 12.5, material name <b>guard</b> and a data storage level of 70.</p>  <p>The dialog box shows the following settings:  Name: Cyl_Guard  Centre of base: X: 0, Y: 0, Z: -10  Centre of top: X: 0, Y: 0, Z: 10  Type: Tube (selected)  Radius: 35  Thickness: 12.5  Cell Properties: Material: guard, Data storage level: 70</p>	
 <b>Pick Edges</b>	<p>Pick the two edges that form a circle on the inside radius of the tube at Z=10. Use  <b>Blend/Chamfer</b> and specify the first chamfer distance to 5 and the second to 4 length units.</p>  <p>The dialog box shows the following settings:  Blade or Chamfer: Chamfer (selected)  Chamfer distance: 5  Second chamfer distance: 4</p>	

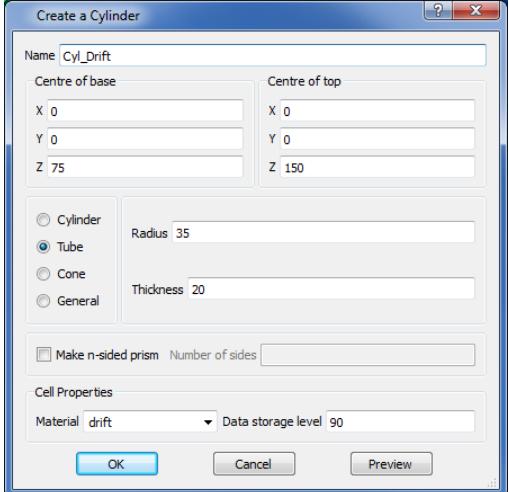
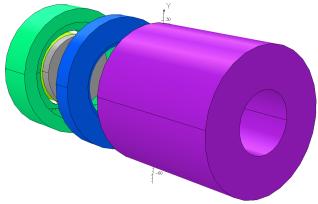
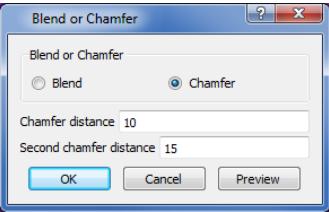
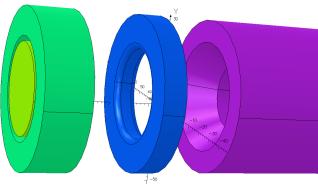
## Focus ring

Add a focus ring next to the cathode as follows:

 <b>Create Cylinder</b>	<p>Create a tube, base <math>(0, 0, 40)</math>, top <math>(0, 0, 50)</math>, radius 35, thickness 15, material name <b>focus</b> and a data storage level of 80.</p> 	
 <b>Pick Edges</b>	<p>Pick the four edges that form the two circles on the inside radius of the tube. Use  <b>Blend/Chamfer</b> to create the blends with a radius of 2.5.</p> 	

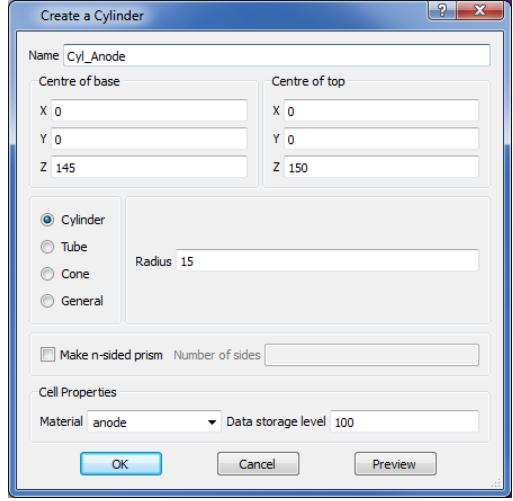
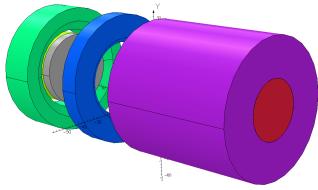
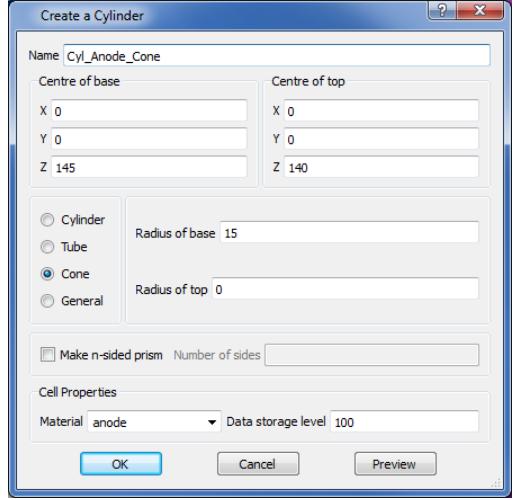
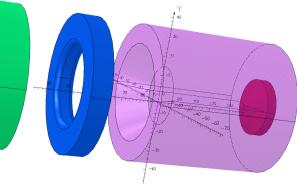
## Drift tube

Add a drift tube to the model as follows:

 <b>Create Cylinder</b>	<p>Create a tube, base <math>(0, 0, 75)</math>, top <math>(0, 0, 150)</math>, radius 35, thickness 20, material name <b>drift</b> and a data storage level of 90.</p> 	
 <b>Pick Edges</b>	<p>Pick the two edges that form a circle on the inside radius of the tube at <math>Z=75</math>. Use  <b>Blend/Chamfer</b> to create the chamfer shown on the right, with chamfer distances of 10 and 15.</p> 	

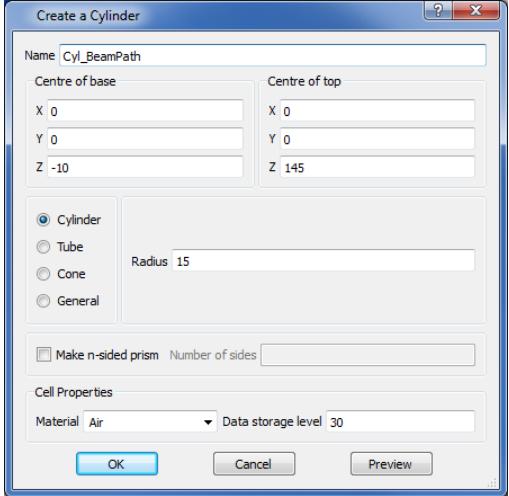
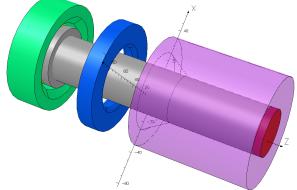
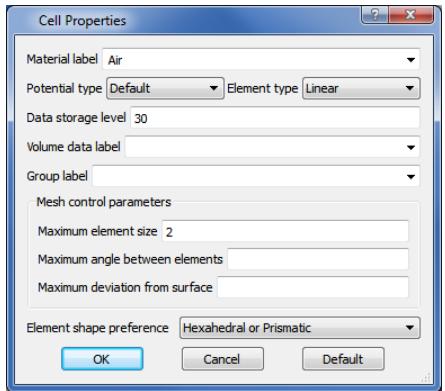
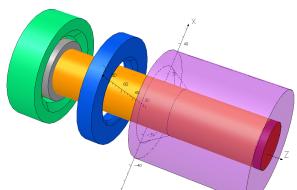
## Anode

The anode is inserted at the end of the drift tube, formed from a short cylinder capped with a cone, as follows:

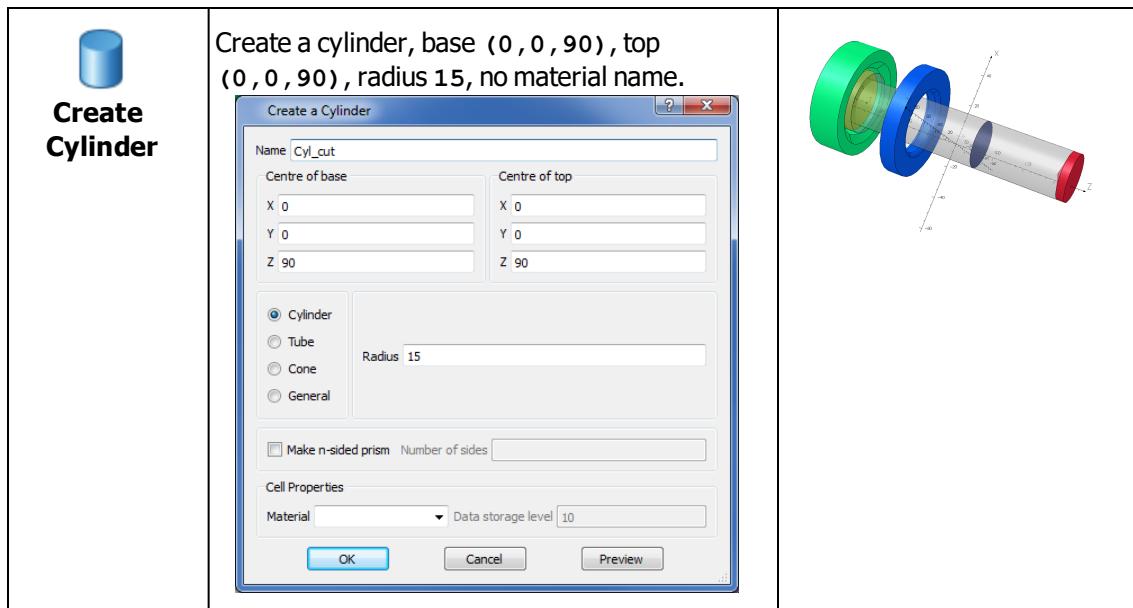
 <b>Create Cylinder</b>	<p>Create a cylinder, base <math>(0, 0, 145)</math>, top <math>(0, 0, 150)</math>, radius 15, material name <b>anode</b> and a data storage level of 100.</p> 	
 <b>Create Cylinder</b>	<p>Create a cone, base <math>(0, 0, 145)</math>, top <math>(0, 0, 140)</math>, radius of base 15, radius of top 0, material name <b>anode</b> and a data storage level of 100.</p> 	

### Mesh size along the beam path

To control the mesh size along the path of the beam, it is convenient to introduce a cylindrical free-space volume.

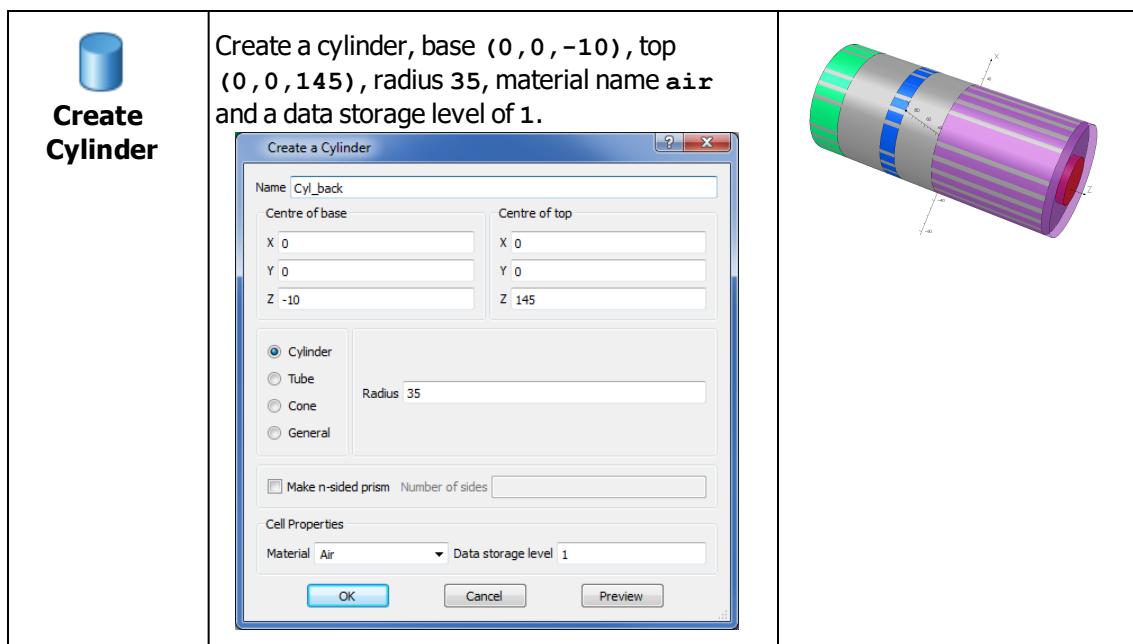
 <b>Create Cylinder</b>	<p>Create a cylinder, base <math>(0, 0, -10)</math>, top <math>(0, 0, 145)</math>, radius 15, material name <b>air</b> and a data storage level of 30.</p> 	
 <b>Pick Cells</b>	<p>Point at the air cylinder and right-click to pick it, select <b>Cell Properties</b> and set the maximum element size to 2 and the element shape preference to <b>Hexahedral or Prismatic</b>.</p> 	

At present, the beam path cannot be meshed with prism elements because extra vertices and edges exist where the top of the drift tube chamfer touches the beam path. In order to generate a mesh of prism elements along the beam path, we need to divide the cylinder into two cells at this point. This is done by introducing a [Sheet face \[page 76\]](#), in this case a zero-length cylinder of radius 15 at  $z=90$ .



## Background

The analysis volume needs to be completed by filling in the spaces between the electrodes. The easiest way to do this is to define a cylinder enclosing the entire geometry.



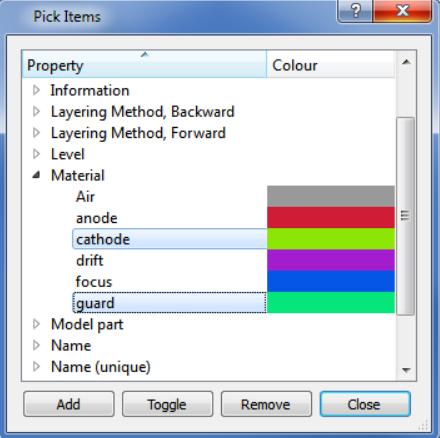
## Charged Particle Simulation

Before starting to specify electrode voltages, materials and emission characteristics the analysis type should be set to **Charged Particle**. This will appropriately tailor the dialogs that now need to be populated with options specific to this type of analysis. In the Analysis Settings window replace the default analysis type with a **Charged Particle** simulation.

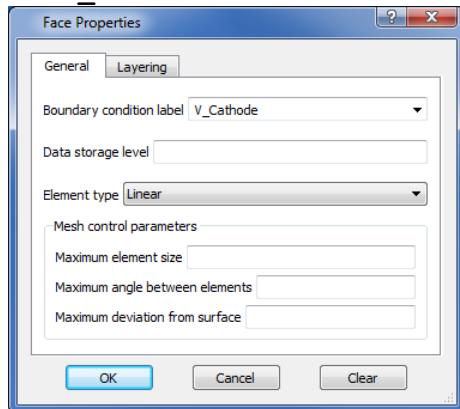
In the configuration pane of the analysis most parameters are populated with defaults; the only action necessary is to set the **Maximum trajectory step** size to 10.

## Boundary Conditions

The metal surfaces of the cathode, guard ring, focus electrode, drift tube and anode need to be set to specified voltages. The Modeller has a useful facility to help with this process: a cell can be picked and the selection can be transferred to all the faces (or edges, or vertices) of the cell. Start by picking the **cathode** and **guard ring**:

 <b>Pick Entities by Property</b>	<p>Open the materials list and select the <b>cathode</b> and <b>guard ring</b> from the list, then press <b>Add</b>.</p>  <p><b>CLOSE</b> the dialog.</p>	
 <b>Pick Faces</b> <b>Change Type of Picked Entities</b>	<p>Select the <b>Pick Faces</b> toolbutton and then press the <b>Change Type of Picked Entities</b> toolbutton to move the selection to the surfaces of the picked cells. Note the entity counts at the bottom right of the window border; the number picked is shown in brackets.</p>	

Right-click and select **Face Properties**. Define the **Boundary condition label** for these faces as **v\_Cathode**.

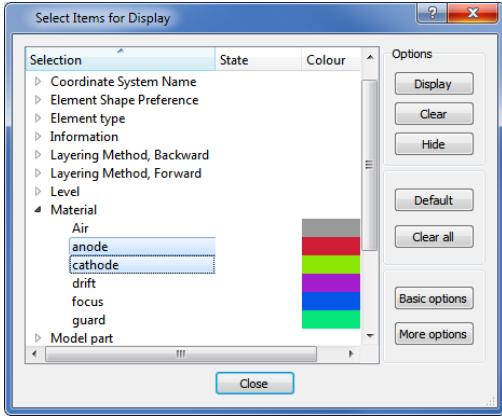
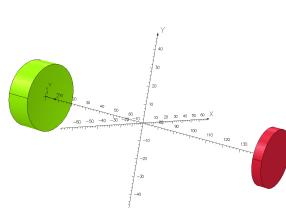


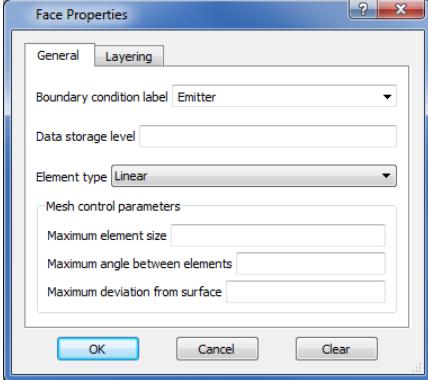
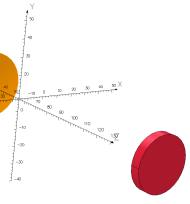
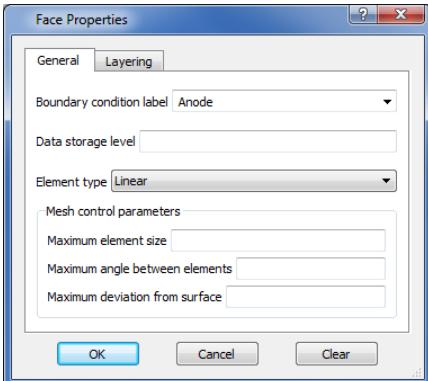
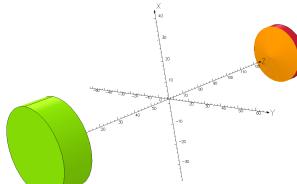
Press **OK** to complete the action.

Using the same sequence of operations, select the **focus** ring and set the boundary condition label to **v\_focus**. Then select the **drift** tube and set the boundary condition label to **v\_drift**.

## Emitter labels

Two special sets of boundary condition labels need to be set on the emitting surface of the cathode and the anode, because emission properties are also attached to boundary condition labels. The curved front surface of the cathode should be selected and its boundary condition label changed to **Emitter**. The conical surface of the anode should then be selected and its boundary condition label should be set to **Anode**. In order to select these faces it will be necessary to hide other parts of the model, or modify the View Selection so that only the cathode and anode are visible, for example:

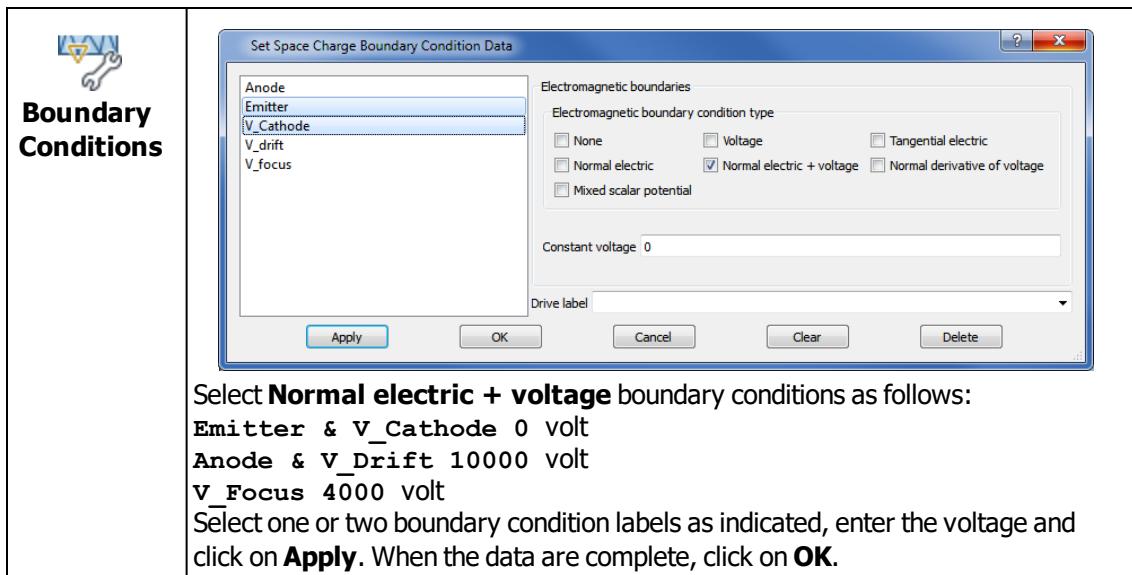
 <b>Select</b>	<p>In the selection dialog press the <b>Clear All</b> button and then select <b>Anode</b> and <b>Cathode</b> from the material list.</p> <p></p> <p>Press the <b>Display</b> button to activate the display of these items and then press <b>Close</b> to exit.</p>	
--	--	---

 <b>Pick Faces</b>  <b>Pick Entity</b>	<p>Pick the curved surface of the cathode by pointing at it and double-clicking.</p> <p>Right-click and select <b>Properties</b>. Define the <b>Boundary condition label</b> for this face as <b>Emitter</b>.</p> 	
 <b>Pick Faces</b>  <b>Pick Entity</b>	<p>Pick the conical surface of the anode and set the <b>Boundary condition label</b> to <b>Anode</b>, as above.</p> 	

After the **Emitter** and **Anode** boundary condition labels have been added, the default display can be reset by pressing the  **Default Selection** toolbutton.

## Assigning Voltage Boundary Conditions

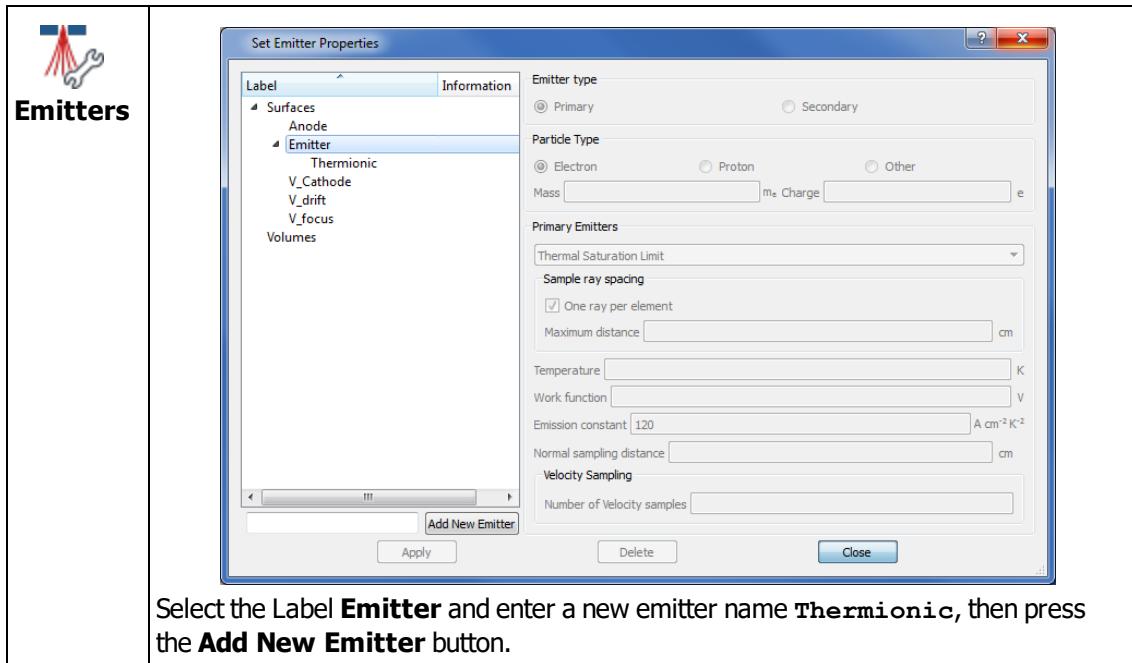
Once the boundary condition labels have been defined the voltages can be set on the surfaces of the electrodes, anode and cathode.



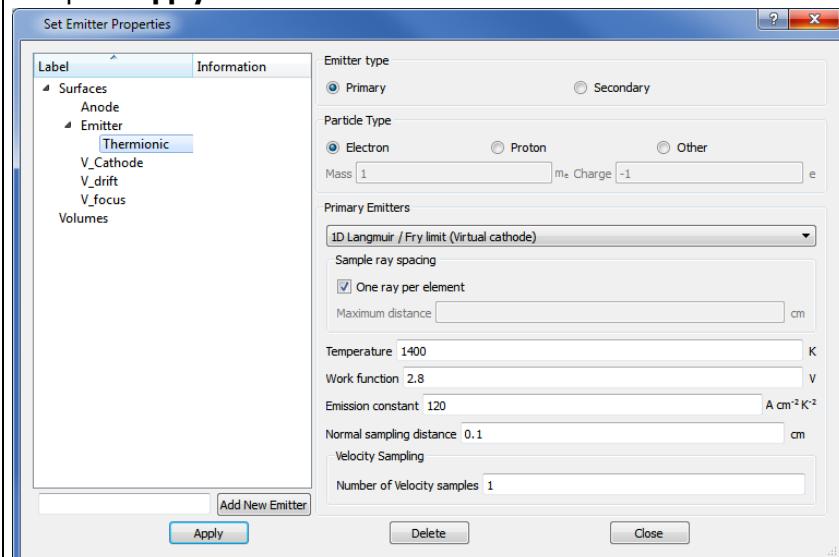
## Specification of Emitters

Thermionic, field effect and plasma primary emission options are available in a Charged Particle simulation. Secondary emission options are also supported. The Charged Particle Analysis section in the Analysis Programs chapter of the ***Opera-3d Reference Manual*** discusses the theory of the different emission models.

In this example a Langmuir/Fry thermionic emission model will be attached to the Emitter surface of the cathode.



Select the **Thermionic** emitter, enter the following properties:  
**Primary Emitter 1D Langmuir/Fry limit (virtual cathode)**,  
One ray per element,  
Temperature 1400 K,  
Work function 2.8,  
Emission constant 120,  
Normal sampling distance 0.1,  
Number of velocity samples 1,  
then press **Apply** to store the data.

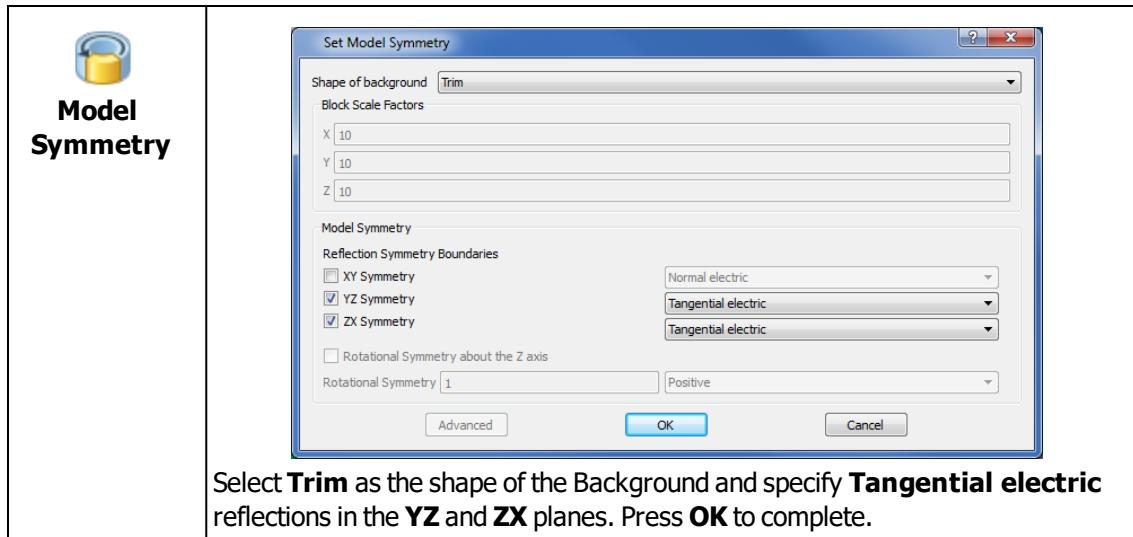


**Close** the dialog.

For the first analysis, only this primary emitter will be defined. Having obtained suitable results, the example will be extended to include the effects of [Secondary Emission \[page 376\]](#).

## Model Symmetry

In this example a complete model has been created and the space charge effects could now be simulated. However the model is rotationally symmetric and it can be reduced in size by exploiting symmetry. The model could be reduced to a thin wedge with an appropriate order of rotational symmetry. However, using a thin wedges would reduce the quality of the element shapes near the axis so for the purpose of this example the model will be reduced to a 90 degree segment.



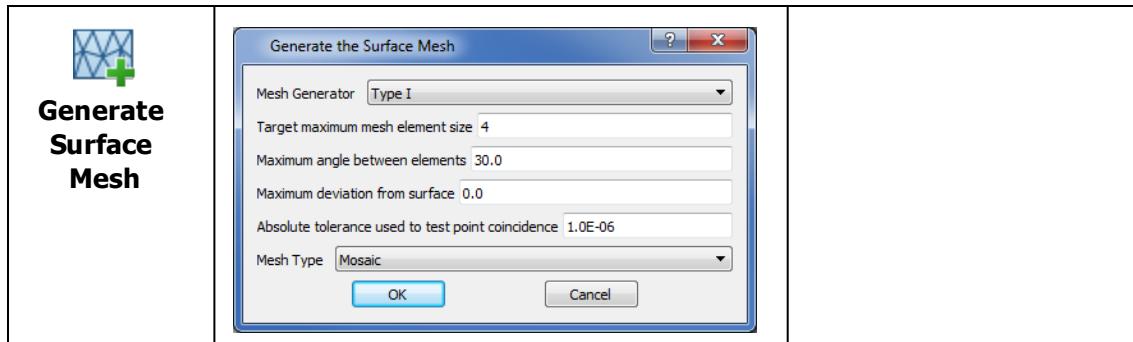
The components of the model need to be joined together before the finite element mesh is created; the symmetry defined above is implemented as part of this process.

Select  **Create Model Body** to join the components and apply symmetry.

## Mesh Generation

The accuracy of a finite element solution is related to element size. To obtain reliable beam current calculations, accurate solutions are required close to thermionic, field effect and free surface emitters. Appropriate element sizes have been defined for the cells in this example model.

Opera produces a surface mesh guided by the element size specified in the model, together with a maximum for the model as a whole. Element shapes are determined by the topology of each cell, the shape preference for the cell and the requested mesh type. By the appropriate construction of our geometry, we have created some cells which can be meshed using prismatic elements and hexahedral elements. The surface mesh can now be created with a mesh type of **Mosaic** and a target maximum mesh size of 5. Smaller mesh sizes specified for particular cells during the construction of the model will apply to those regions.

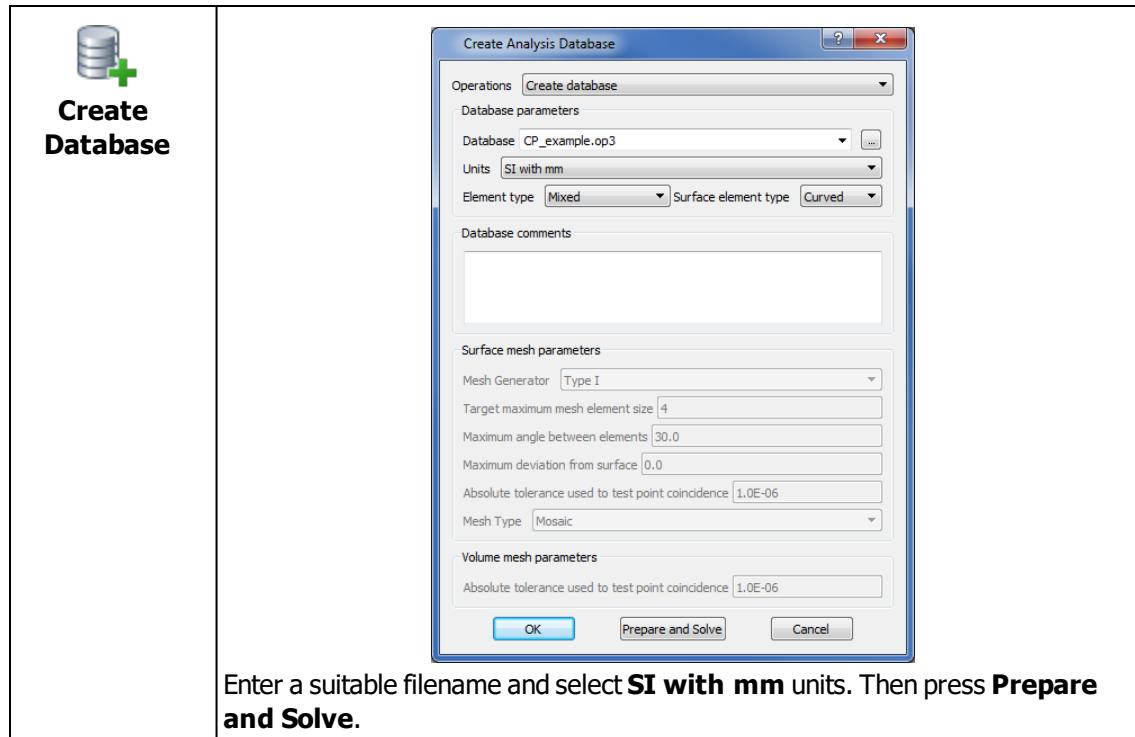


Once the surface mesh has been created a volume may be constructed using  **Generate Volume Mesh.**

## Running the Charged Particle Simulation

In the Analysis Settings dialog a nonlinear simulation with a maximum of 21 iterations and a convergence tolerance of 0.001 was selected. The relaxation parameter should normally be set to 1 because the **Charged Particle** solver determines an appropriate relaxation value automatically.

An Opera database can now be prepared and immediately solved.



The Charged Particle solver uses a simple update method to include the effect of space charge and so the solution may be difficult to calculate, particularly if the beam current is strongly dependent on the space charge (for example with a strongly space charge limited solution close to cut off). More than 21 iterations might be required to achieve convergence in such cases.

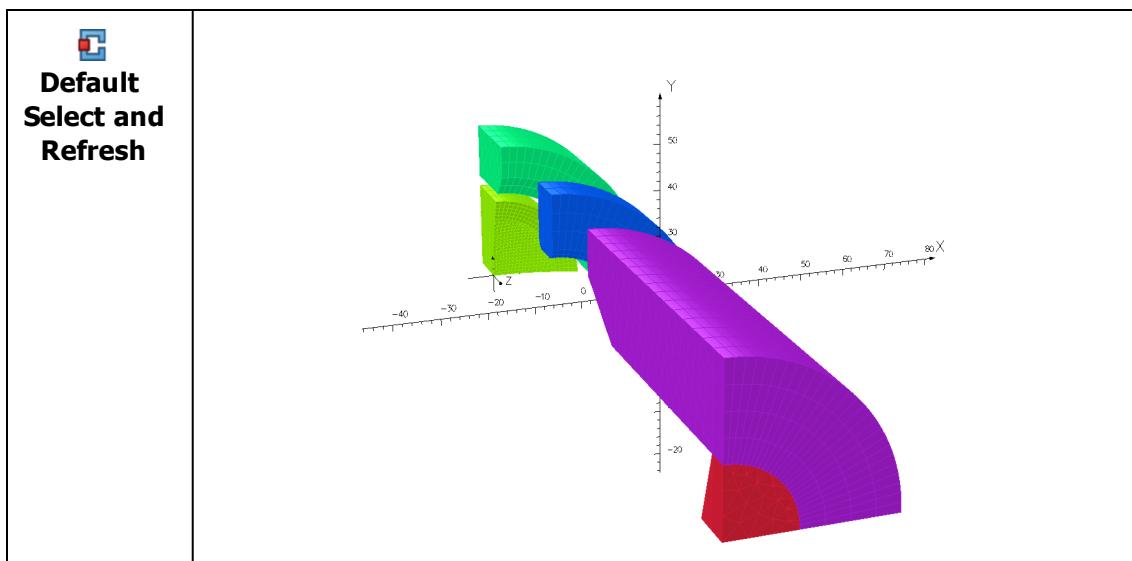
## Post-Processing, Analyzing the Results

### Loading and Displaying the Model in the Post-Processor

When the Charged Particle simulator completes the analysis, the results can be displayed using the Post-Processor. The Post-Processor automatically loads and displays the most recent model if it is run from the Modeller or from the Solver window. In other cases the model may be opened from the

Post-Processor using  **Open (Activate and Load) (Ctrl+O)**.

The default display can be viewed by selecting the  **Default Select and Refresh** toolbutton.

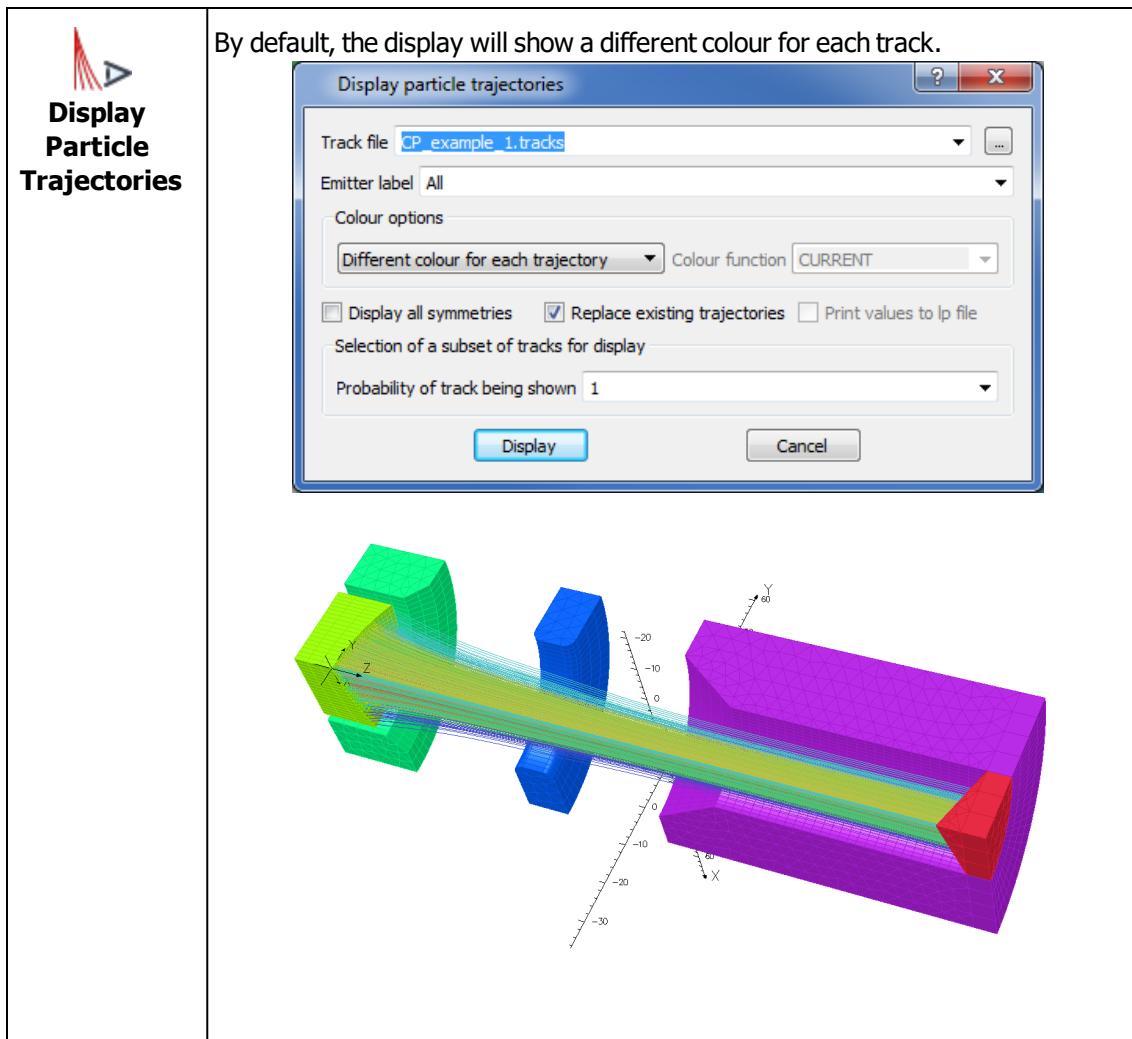


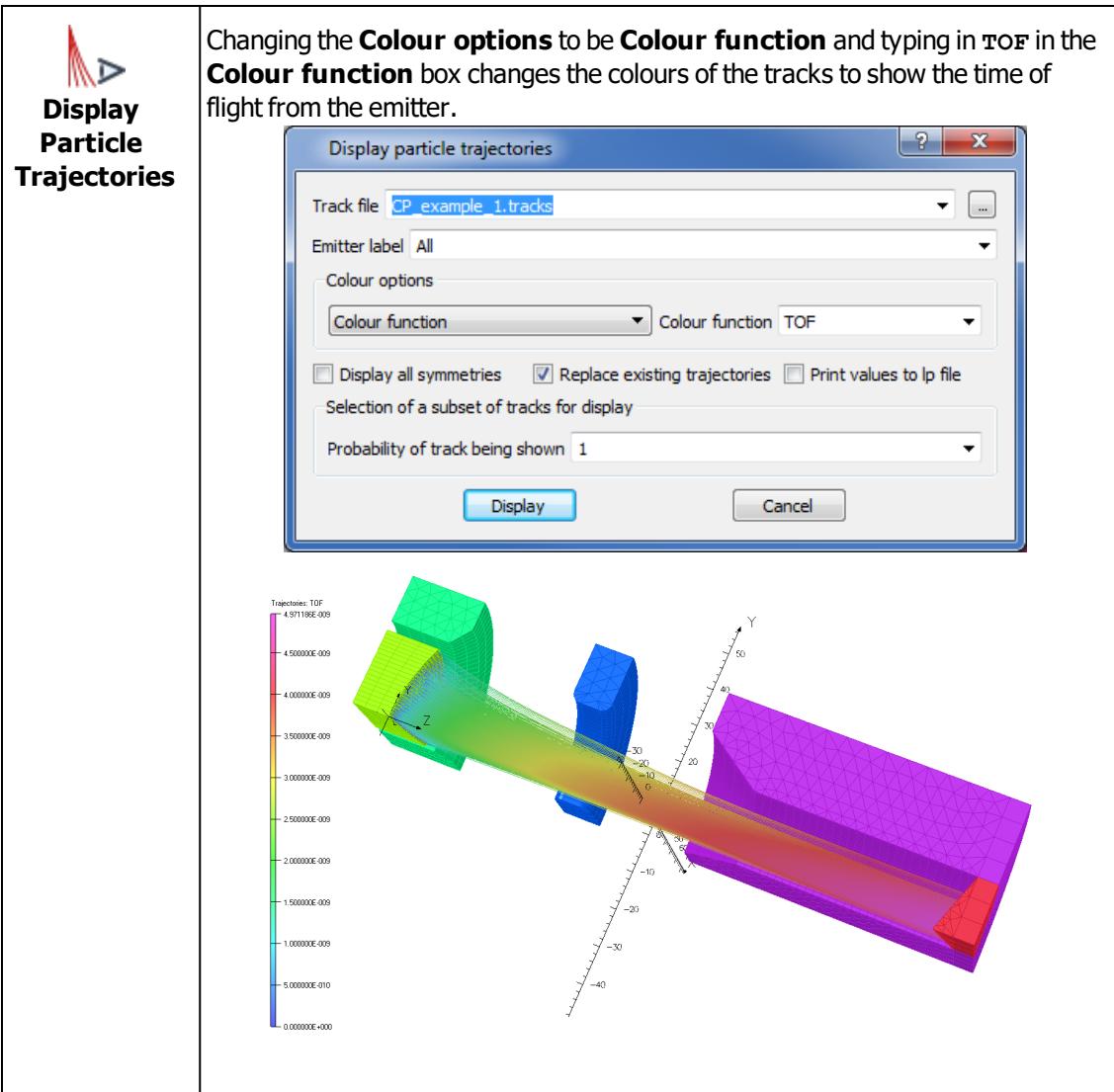
The default display does not show volumes labelled Air, but they can be selected and added by using  the **Volumes -> Materials** section in the **Select** dialog.

All of the standard Post-Processor commands are available for displaying the field and potential results (values, graphs, contour maps, integrals etc.).

### Displaying Trajectories

The trajectories results are stored during the simulation for display in the Post-Processor:





The value given for **Probability of track being shown** has the following meaning:

$0 < n < 1$  : probability of a track being displayed

$n \geq 1$  : display every  $n^{\text{th}}$  track

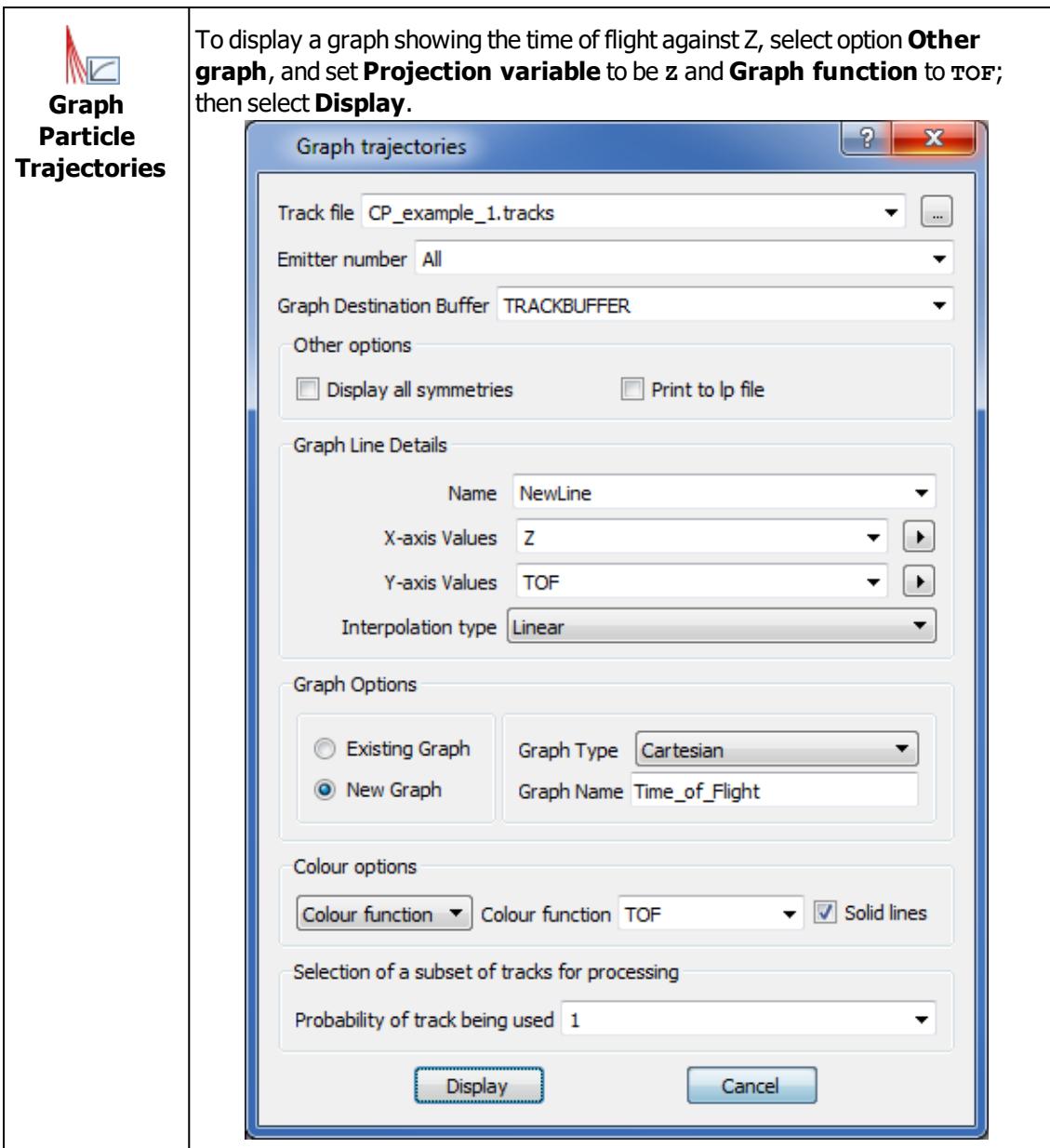
Because the model was created using reflection or rotational symmetry, the tracks display may be seen in all the reflections by clicking on **Display all symmetries** in the **Display Particle Trajectories** dialog. The model can also be displayed with all or some of its symmetrical components

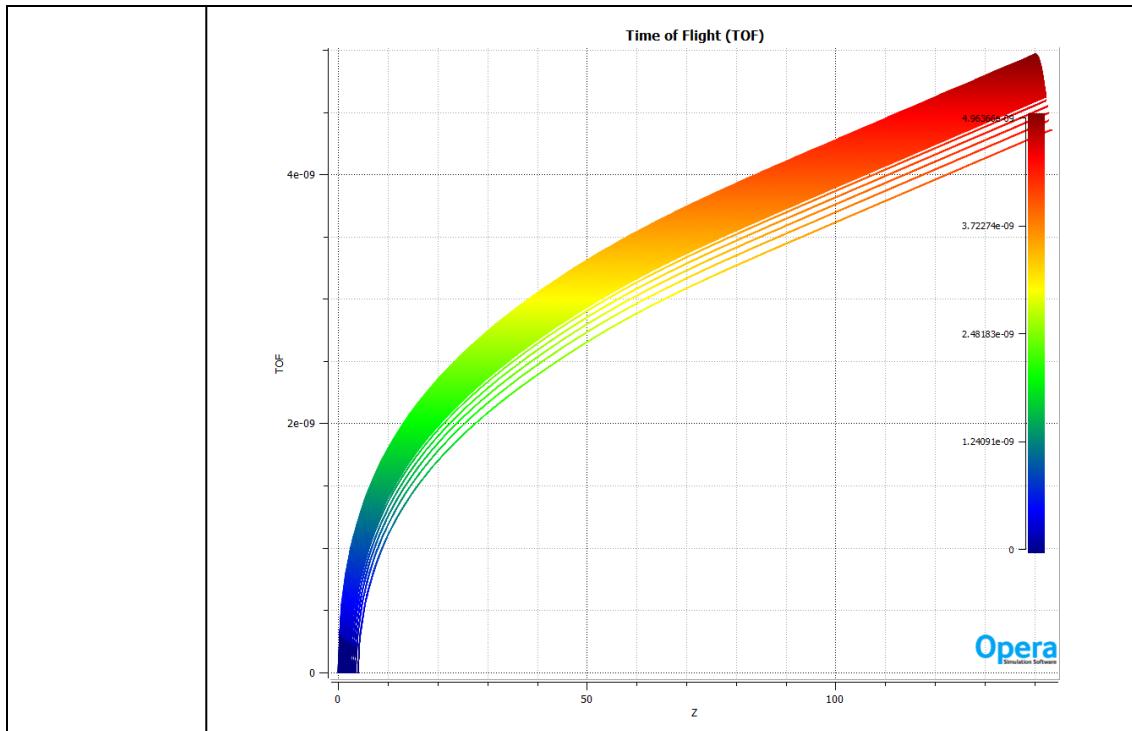
(see the  **Select** dialog). The tracks can be hidden using the  **Toggle visibility of trajectories** toolbutton.

## Graph of trajectories

There are other functions available for processing the particle trajectories. In addition to displaying

the trajectories on the 3d model, they can also be plotted on 2d graphs using the  **Graph Trajectories** toolbutton.



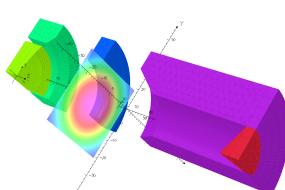


A number of trajectory-specific system variables are available for calculation and plotting. They are described in the [VIEW](#) command section in the ***Opera-3d Reference Manual***.

The system variables available depend on how the results are being plotted. For example, with the graphing option they include the **X**, **Y** and **Z** coordinates of the particle, its velocity components (**VELX**, **VELY** and **VELZ**) and its starting position (**XSTART**, **YSTART** and **ZSTART**).

## Trajectory intersections

The Post-Processor provides facilities to define 2d surfaces and plot the fields on these surfaces. The surfaces can be quadrilaterals in Cartesian, cylindrical or spherical coordinates. Intersections of the trajectories with these surfaces can be calculated and plotted.

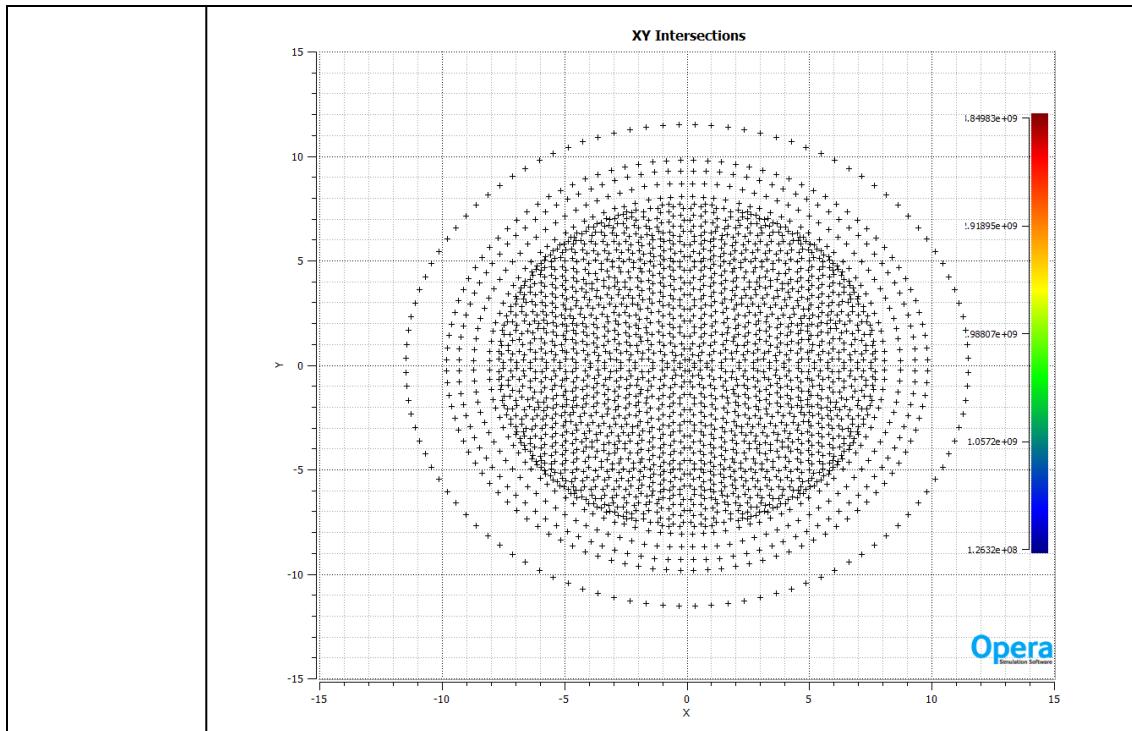
 <b>Cartesian Patch</b>	<p>Define a patch in the XY plane, at <math>z=60</math>, with <math>x=-20</math> to <math>x=+20</math> and <math>y=-20</math> to <math>y=+20</math>. Use 32 points on each side. Display the z-component of the electric field (<math>E_z</math>). Note that the numbers of points are given as a power of 2. This is essential when calculating the beam current density map later.</p> <p><b>Field on a Cartesian Patch</b></p> <p>Set field point local coordinate system Buffer: Cartesian</p> <p>On XY Plane   On YZ Plane   On ZX Plane   Any Direction</p> <p>First corner ... X: -20      Y: -20</p> <p>Opposite corner... X: 20      Y: 20</p> <p>XY plane ... Z coordinate: 60</p> <p>Sides 1 and 3 in X direction Sides 2 and 4 in Y direction</p> <p>Number of points ... on sides 1 and 3: 32      on sides 2 and 4: 32</p> <p>Component for map: <math>E_z</math></p> <p><b>Evaluate fields</b>   <b>Evaluate and Map</b>   <b>Cancel</b></p> <p><b>Press Evaluate and Map.</b></p>	
---	---	---

The intersections of the trajectories with the patch can now be displayed.

**Intersect Trajectories with Patch**

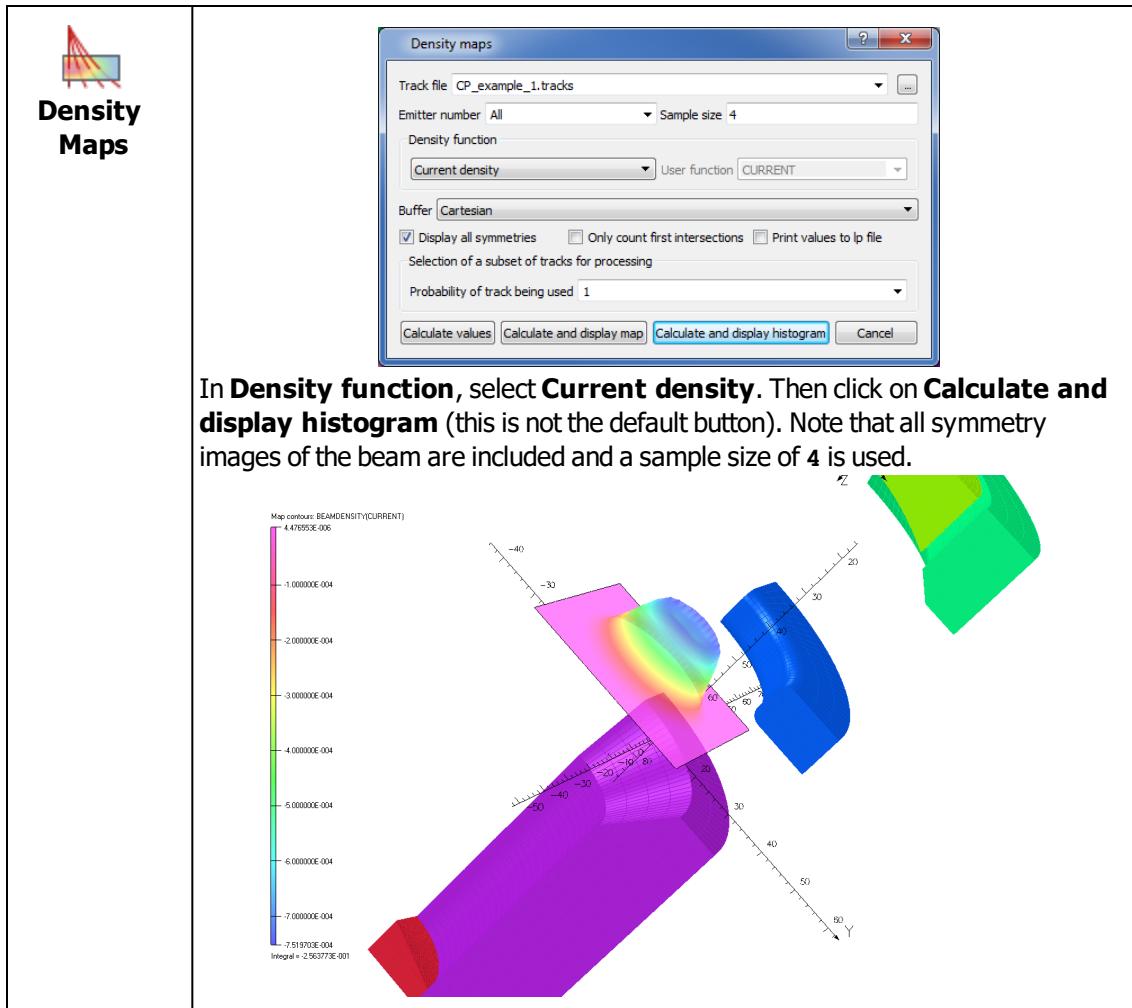
Choose the **Other** graph option and plot the XY coordinates of the intersections with the square.

Note the use of a **Colour function** to display information about the particle velocity  $\text{SQRT}(\text{Velx}^2 + \text{Vely}^2)$  and also that **Solid lines** joining the points were deselected.



### Beam current density display

The current density distribution in a beam of charged particles, consisting of many trajectories, can also be plotted using the intersection facility. The surface patch used for calculating fields and displaying the intersections is defined by a grid of points. The program overlays a regular lattice over the grid of points and integrates the trajectory currents in each cell of this lattice. The cell values are then transferred to the points. A Fourier filter can be applied to remove noise caused by using a small number of trajectories, but this can be done only if the number of points in each direction of the Cartesian patch is a power of 2. The sample size controls the number of adjacent points in the patch which are included in this smoothing algorithm.



In **Density function**, select **Current density**. Then click on **Calculate and display histogram** (this is not the default button). Note that all symmetry images of the beam are included and a sample size of **4** is used.

The beam current density data can be displayed as a contour, line or 3d histogram plot.

The **Sample size** in the dialog determines the degree of filtering that will be applied to reduce noise caused by using a small number of trajectories. It should typically be one tenth of the number of points in each direction of the grid; the value can be increased if the current density contours are obviously affected by noise but this will quickly degrade the quality of the results. Increasing the number of trajectories used to model the beam is the best method of reducing the error in the current density calculation.

## Secondary Emission

The **Charged Particle** solver will stop a trajectory when its path hits a material cell, unless the cell has been given the label **BEAMPASS** to avoid this. Trajectories will also stop if they intersect a surface that has secondary emitters attached to it. The input particle type is checked against all secondary

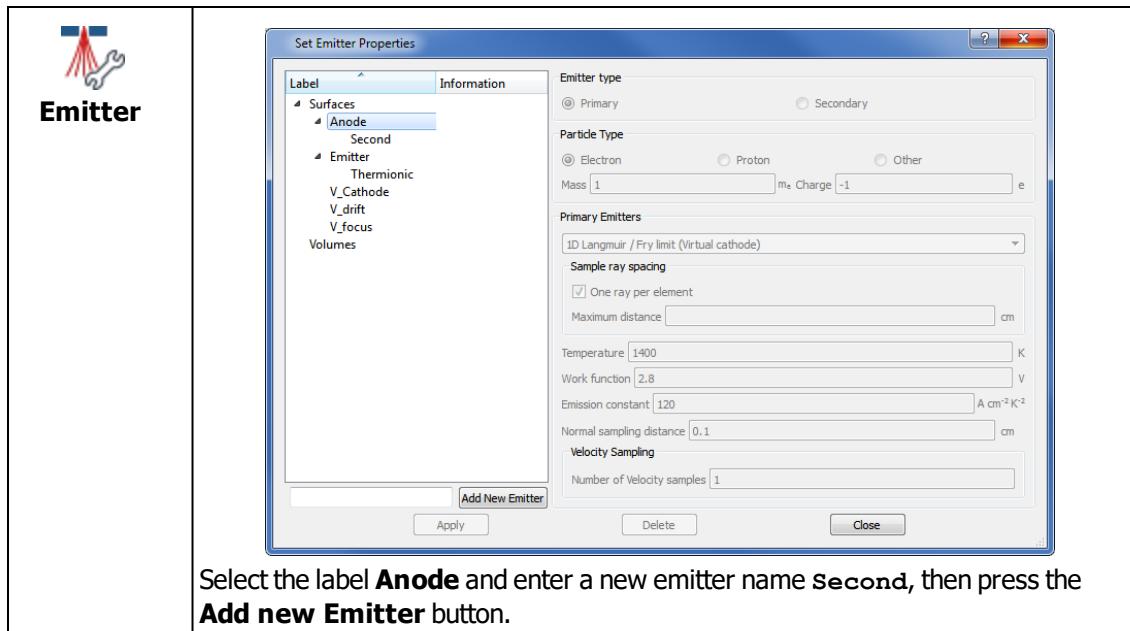
emitters that have been attached to the surface and for those that match, secondary particles will be generated.

The creation of secondary particles is limited by the number of **Generations** specified in the Charged Particle analysis parameters. The generation number of a secondary particle is one higher than its parent and primary beams have a generation number of zero. This means that if **Generations** is set to 1, secondary particles will be created only by the primary beams.

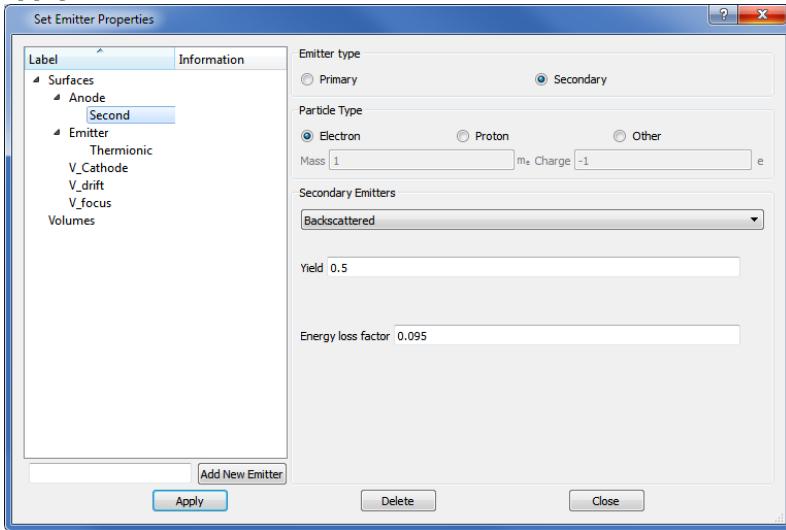
The basic types of secondary available in the **Charged Particle** solver include back scattered (almost elastic) particles and true secondaries with a cosine scattering distribution. More advanced scattering models can be added by defining the secondary scattering fraction as a function of the direction and energy of the input particles and output particles. For more information see [Particle Collision and Scattering in the Charged Particle solver \[page 762\]](#).

In this example, secondary emission characteristics will now be added to the anode, to produce basic back scattered secondary particles.

Return to the Opera-3d Modeller or restart the Modeller and open the **opc** file that was stored when the model was saved for analysis.



Select the **Second** emitter, and set the **Emitter type** to **Secondary** and the **Particle type** to **Electron**. The **Secondary Emitter** properties should be **Backscattered** with a **Yield** of **0 . 5** and an **Energy loss factor** of **0 . 095**. Press **Apply** to store the data.



When a **Backscattered** secondary is specified, the output particle type (in this case an electron) must be the same as the incoming particle type. If a **true secondary** type had been used the input and output particle types can be different and both must be specified.

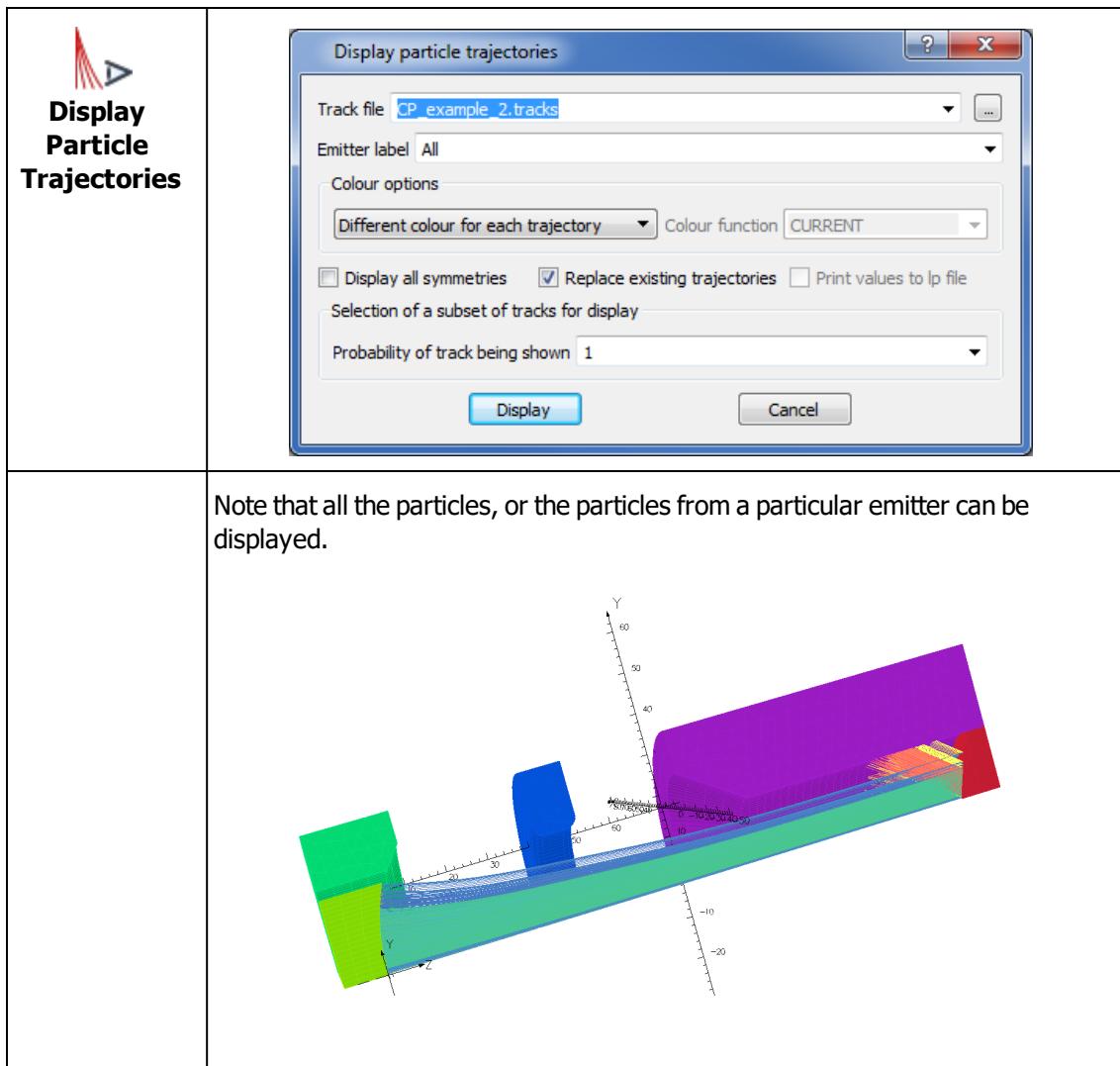
## Charged Particle solution with secondaries

The analysis parameters have now been updated to request 1 generation of secondary particles, and a new<sup>1</sup> **Charged Particle** analysis database can be prepared and solved.

Once the **Charged Particle** analysis has been completed the results can be post-processed. Load the model, select the default display and then display the trajectories.

---

<sup>1</sup>If the Modeller window is still open and the volume mesh is still present, a second case can be added to the first database. In the Post-Processor, make sure to close the loaded database first. It is important that the mesh is exactly the same; otherwise create a new Charged Particle analysis database.



The space charge effects caused by the secondary particles are included in the solution.

## Lossy Dielectrics

In this example, the secondary particles produced from the anode collide with the drift tube wall. If the drift tube was constructed from a material with a very low value of electrical conductivity (a lossy dielectric), the current flow resulting from the secondary particles would change the voltage distribution in the gun.

This effect can be modelled using Opera's Lossy Dielectric simulator<sup>1</sup>. The static field Lossy Dielectric simulator determines the steady-state voltage distribution, assuming that the currents have been flowing for a sufficiently long time for transient effects to decay away.

In order to access this facility for the **Charged Particle** solver, it is only necessary to specify the conductivity of the drift tube material, and update the **Analysis Settings**. Check the **Lossy dielectrics** option, limit the number of iterations to 3 and accept the default tolerance of  $1.0\text{E-}03$ .

The lossy dielectric simulation is integrated with the Charged Particle solver's nonlinear analysis. The drift tube material could have a nonlinear electrical conductivity, in which case the number of nonlinear iterations allowed for the lossy dielectric solution should be set (typically in the range 3 to 5). If the drift tube material has a constant electrical conductivity, only one iteration is required for the lossy dielectric solution.

Note that for this example, if a lossy dielectric simulation is required the boundary conditions on the drift tube are not appropriate. The poorly conducting drift tube might be connected to a good electrical conductor at some points. In this case, delete all voltage boundary conditions from the drift tube. Then, for example, pick the face at the end of the drift tube at  $Z=150$  and apply the voltage  $v_{\text{drift}}$  here.

The steady-state lossy dielectric simulation can also be used with any electrostatic calculation. It involves a two-step solution: the voltage distribution on conductors is calculated first and this information is then used to calculate the electrostatic fields in non-conducting materials.

## Beam Magnetic Fields

The self magnetic field produced by a particle beam often has no effect on the beam trajectory. However for high energy and current electron beams the self magnetic field may be important. The **Charged Particle** solver has the option of calculating the magnetic field produced by the beam (selectable in the **Charged Particle** analysis settings dialog). Selecting this option will increase the solution time by approximately a factor of two, and therefore it should be used only when necessary. It is essential that the symmetry of the model is correctly defined when the self magnetic field option is used, because the magnetic fields from the symmetry copies of the beam must be included as well.

Note that the **Opera-3d Post-Processor** can be used to estimate the importance of the self magnetic field.

## External Magnetic Fields

External magnetic fields can be applied to a Charged Particle simulation either by adding Biot-Savart coils to the model, or by importing a specified field. For an overview of the options, please see "The Charged Particle Algorithm" in the *Opera-3d Reference Manual*.

---

<sup>1</sup>This feature requires a licence for Opera-3d/LD.

## Symmetry

By default the external magnetic field is assumed to have the same symmetry as the electric field, e.g. a normal electric field boundary condition will imply normal magnetic fields as well. A user defined variable (`#magsign`) can be used to set the applied magnetic field symmetry. If it exists with a negative value then the applied magnetic fields have inverse symmetry reflections compared with the electric fields, e.g. a normal electric field boundary condition will imply tangential magnetic fields.

Note that the beam self magnetic field always has inverse symmetry reflection compared with the electric fields (i.e. if the electric fields are tangential to a symmetry plane, the beam magnetic field will be normal to that plane). This assumed symmetry cannot be changed.

# **Chapter 11**

## **RF Cavity and Waveguide Modal and Harmonic HF Example**

### **Introduction**

---

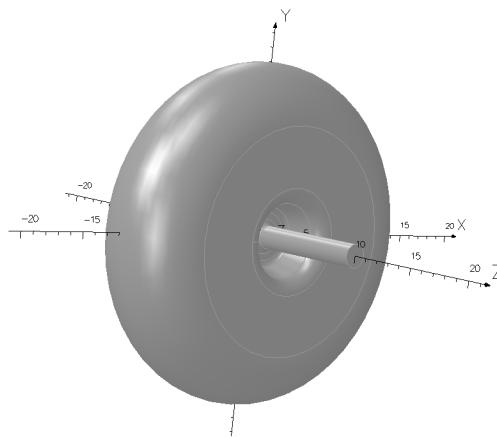
Opera-3d includes a solver for high frequency analysis. The solver incorporates displacement current in the field equations so that it can perform full-wave solutions, and allows simulations of devices that may be multiple wavelengths in size.

Two versions of the High Frequency solver are available: **Modal High Frequency**, which solves for eigenvalues in closed conducting cavities, and **Harmonic High Frequency**, which performs a harmonic solution for general geometries.

This chapter presents demonstration examples of the use of both versions of the solver. The first is based on determining the resonant frequencies and modes in a cavity typical of those used in linear particle accelerators. The second simulates propagation in a section of a loaded rectangular waveguide.

### **Cavity Modelling Using the Modal HF Solver**

A typical cavity from a linear particle accelerator is shown in [Figure 11.1](#). The geometry will be created, the analysis options set up and the simulation run. Use of the Opera-3d Post-Processor to view the results of the simulation will be described.



*Figure 11.1 Geometry of the Cavity*

Creating the geometry of the model is dealt with only briefly, but some features that have not been used in other examples are demonstrated, in particular:

- edge blending is used to create a smooth transition between planes;
- boundary conditions are used to impose the symmetry of the field for the modes required (and, consequently, reduce the problem size).

[Figure 11.1](#) shows the geometry of the interior of a cavity and a short section of beam line to which it is attached. This is really the geometry of the space enclosed by the walls of the cavity, which only exist in the model as boundary conditions. Additional boundary conditions are used to impose the symmetry of the geometry and fields so that, in this example, a solution is needed on only an eighth of the complete geometry.

## Waveguide Modelling Using the Harmonic HF Solver

The second example is a length of loaded waveguide, as shown in [Figure 11.2](#); The geometry is typical of a simple waveguide load, frequently used in practice as a matched termination.

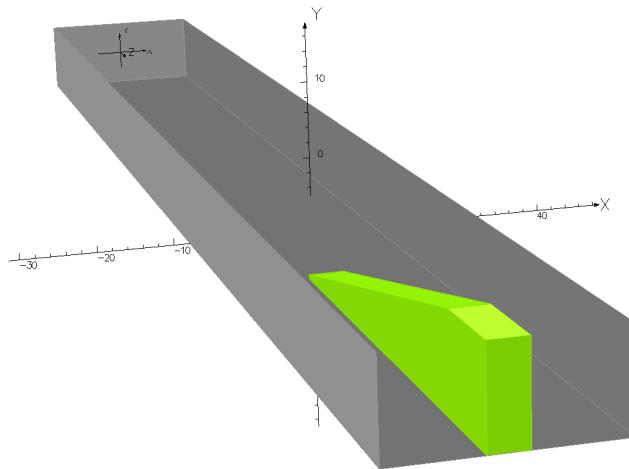


Figure 11.2 Geometry of Waveguide

Two variants of the model will be used. Initially, the geometry will comprise only a dielectric wedge embedded in a block of air, which represents the waveguide. As in the modal example, the walls of the waveguide will be represented by a normal electric (i.e. perfect conductor) boundary condition.

A second variant will then be constructed, in which the walls are represented by a conducting material. This will illustrate the use of the surface impedance boundary condition (SIBC) to avoid the fine mesh that would otherwise be needed to resolve the skin depth in the waveguide walls.

These examples demonstrate how to:

- apply a boundary condition to define a particular mode at an input port;
- set cell and surface mesh options to give different mesh element types;
- display fields in the waveguide and extract the reflection coefficient of the device;
- use the SIBC to avoid fine meshing in highly conducting materials.

## Cavity Modelling Using the Modal HF Solver

---

### Building the Model Geometry

The cavity is constructed from three bodies; two of these are cylinders, and one is a torus. Use the following tables for the dimensions, which are given in inches.

The table below defines the two cylinders that are used; note that c2 is actually a truncated cone, as it has different radii at top and base.

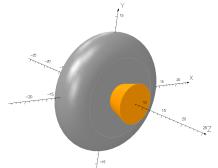
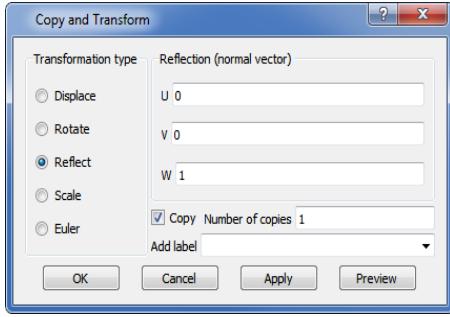
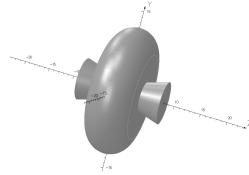
Name	C1	C2
<b>Base Centre</b>	(0, 0, -4)	(0, 0, 1.2)
<b>Top Centre</b>	(0, 0, 4)	(0, 0, 8)
<b>Shape</b>	<b>Cylinder</b>	<b>Cone</b>
<b>Radius</b>	8	
<b>Radius of Base</b>		2
<b>Radius of Top</b>		4

The torus is defined by the following details.

Name	T1
<b>Centre of Torus</b>	(0, 0, 0)
<b>Major Radius</b>	8
<b>Minor Radius</b>	4

### Copying and subtracting

Any existing body can be replicated. In this example, a copy of the truncated cone c2 is made by a reflection in the XY plane.

 <b>Pick Bodies</b>	<p>Note the high-lighting of the body edges when the cursor is over a body; this shows which body will be picked. Move the cursor over the truncated cone and pick it by clicking the right mouse button.</p> <p>Select  <b>Copy and Transform</b> from the context menu.</p>	
	<p>Set the <b>Transformation type</b> to be <b>Reflect</b>, and the <b>Reflection vector</b> to be <math>(0, 0, 1)</math>. Select the <b>Copy</b> option and set the <b>Number of copies</b> to 1.</p>  <p>Click on <b>OK</b> to complete the copy operation and to close the dialog.</p>	

Using double clicks of the mouse, first pick the cylinder c1, followed by one of the truncated cones. Right click and select **Combine bodies > Subtraction, with regularization** to cut the truncated cone from the cylinder c1. Repeat this operation using the cylinder c1 and the other truncated cone.

This should leave the geometry as shown in [Figure 11.3](#), with a similar view from the negative Z direction.

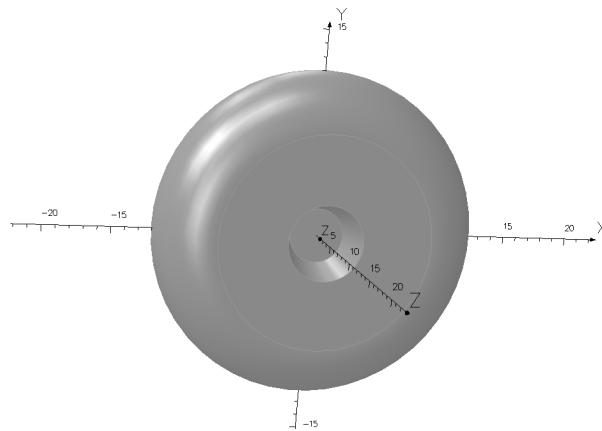
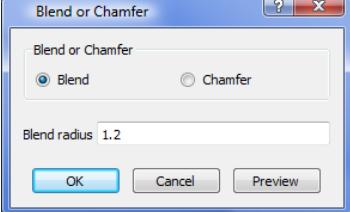
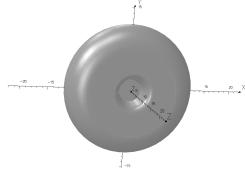


Figure 11.3 Truncated cones cut from the cylinder

### Blending edges

The two sharp outside edges of the conical holes, at Z coordinates -4 and 4, formed by the subtraction operation, should now be smoothed using a blend operation.

 <b>Pick Entity</b>  <b>Pick Edge</b>	<p>Pick the 2 edges on the positive Z side (as shown) and the equivalent 2 edges on the negative Z side. Use a right mouse button click to pick the fourth of these edges, since this picks and then displays a context menu. Select <b>Blend or Chamfer</b> from the context menu.</p>	
	<p>Choose <b>Blend</b> and set the <b>Blend radius</b> to <b>1.2</b></p> <div style="text-align: center;">  </div>	

Now repeat the same procedure with the 4 edges at  $Z = \pm 1.2$ , but this time set a **Blend radius** of 0.4.

## Beam line

The beam line cylinder should now be added, with the following dimensions.

<b>Name</b>	c3
<b>Base Centre</b>	(0,0,-10)
<b>Top Centre</b>	(0,0,10)
<b>Radius</b>	0.8

Figure 11.4 shows a close-up of the two blends on the negative Z side of the cavity together with the beam line cylinder.

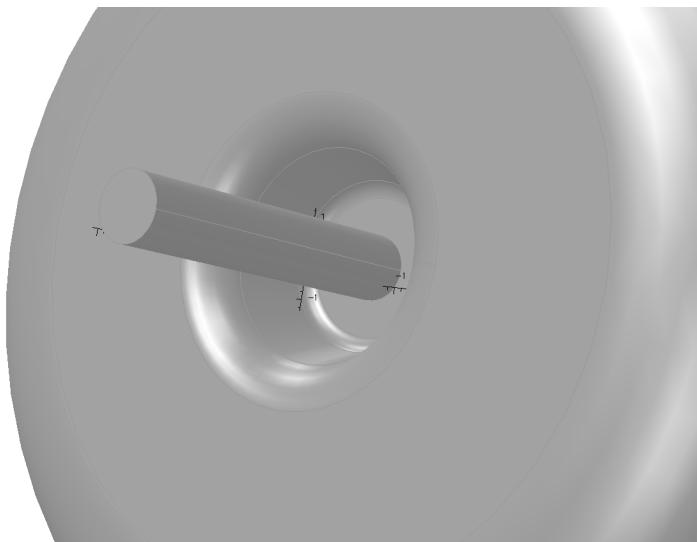


Figure 11.4 Blended edges

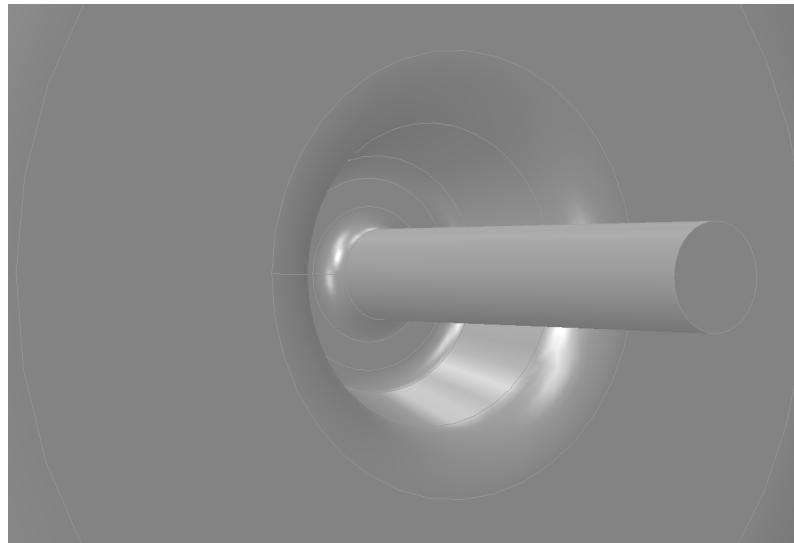
## Union of cavity and beam line

A blend with **Blend radius** of 0.4 should be applied between the beam line and the cavity. However, the edges between them do not exist yet. They are created using the **Union** operation.

 <b>Pick Body</b> <b>Combine Bodies</b>	 <b>Pick All</b> toolbutton.  Complete the union operation by selecting <b>Combine bodies &gt; Union, with regularization</b> from the context menu.	
---	--	--

### Blend new edges

This has created 4 edges between the beam line and the cavity, and a blend can now be applied as before. The resulting geometry is shown in [Figure 11.5](#).



*Figure 11.5 Beam line blended to cavity*

The model is now complete.

## Setting Options for the Modal HF Analysis

The next step is to select the analysis module, **Modal High Frequency**, required for the eigenvalue calculation. The options for **Modal HF** can then be specified, and information about boundary conditions and the symmetry of the model can be added to complete the description.

### Analysis options

In the Analysis Settings window replace the default analysis type with **Modal High Frequency**. The analysis configuration dialog will be shown, allowing the user to set the number of eigenvalues for which to search, and the lower and upper frequency limits. The **Modal High Frequency** solver searches for eigenvalues in ascending frequency order, starting at the specified lower frequency limit. It will stop searching when it finds the required number of eigenvalues. If the upper frequency limit is not the same as the lower limit, fields will be calculated for those eigenvalues that lie in the frequency range; only these modes will be available for examination in the Post-Processor. If the lower and upper frequency limits are the same, fields will be calculated for all eigenvalues that have been found.

In the configuration pane enter the following analysis data for **Modal High Frequency** (frequency range is entered in Hz):

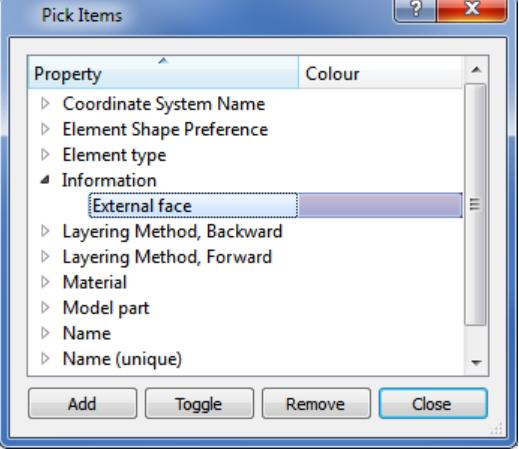
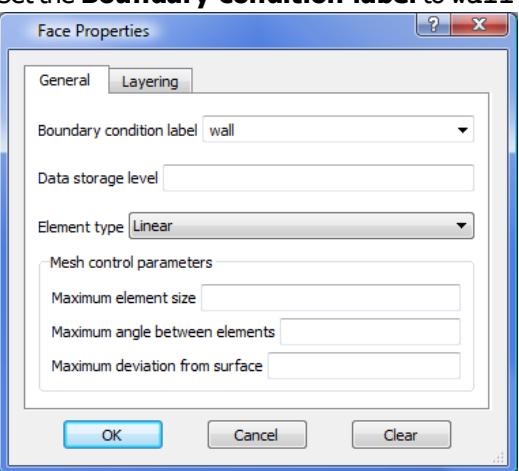
- **Lower frequency:** 3 . 0E+08
- **Upper frequency:** 8 . 0E+08
- **Number of eigenvalues:** 3

A finite element mesh will be generated to perform the eigenvalue calculation; the accuracy of the calculated eigenvalues and their field distributions depend on the finite element mesh size that is used.

## Boundary conditions

The model that has been created describes the space inside a metal cavity, and the exterior surface of the model is therefore the inner surface of the enclosing metal. In the **Modal High Frequency** solver, metals are modelled by applying a perfect electric conductor boundary condition.

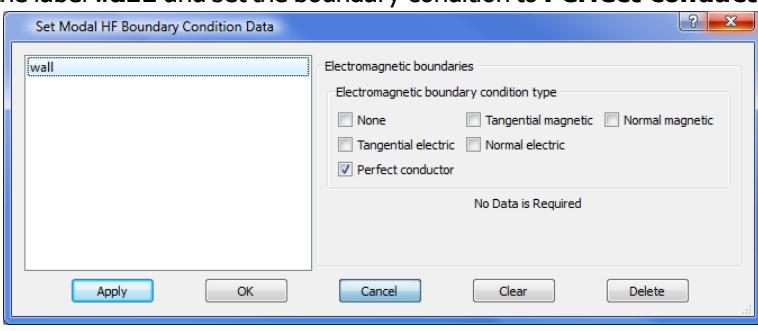
The easiest way to select the relevant surfaces for this boundary condition is to use the property that they are external faces of the model.

 <b>Pick by Property</b>	<p>Open the <b>Information</b> item in the list view and select <b>External face</b>.</p>  <p>Pick the external faces by pressing <b>Add</b> and then exit by pressing <b>Close</b>. The boundary condition can then be defined by using the context menu (right mouse button) and selecting <b>Face Properties</b>.</p>	
	<p>Set the <b>Boundary condition label</b> to <b>wall</b></p>  <p>Press <b>OK</b>.</p>	

The properties of the boundary condition may now be defined.

**Boundary Conditions**

Select the label **wall** and set the boundary condition to **Perfect conductor**.



Select **OK**.

## Model symmetry

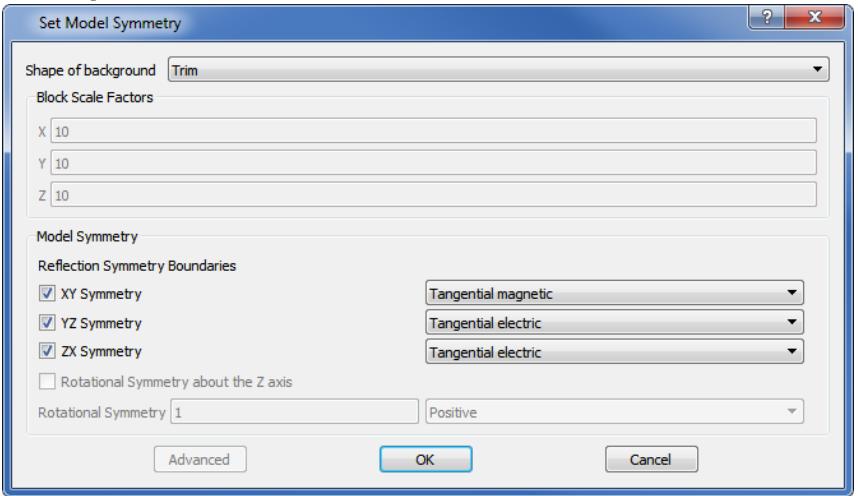
The model that has been build represents the complete cavity. However, in most cases only the low order symmetrical modes are required and the symmetry of the geometry and of the required modes can be used to reduce the model size. Using a minimum symmetry section will make the calculation much faster.

The Modeller's **Model Symmetry** option provides the most convenient method of reducing the model to this state. The reflection and rotational symmetries of the required modes are defined, and the Modeller will then automatically trim the model to the minimum symmetry section. In this example, appropriate reflection boundaries for low order symmetrical modes are tangential magnetic in the XY plane and tangential electric in both the YZ and ZX planes.

This choice of boundary conditions permits the existance of transverse magnetic (TM) modes in the full cavity. The  $TM_{010}$  mode in particular is often required in accelerator cavities owing to its uniform axial electric field.

  
**Model symmetry**

Select the **Trim** option and specify the required mode symmetry by setting the **Reflection Symmetry Boundaries** to **Tangential magnetic** for the **XY Symmetry**, and **Tangential electric** for both **YZ Symmetry** and **ZX Symmetry**.



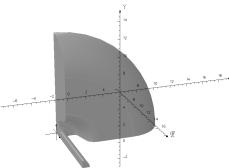
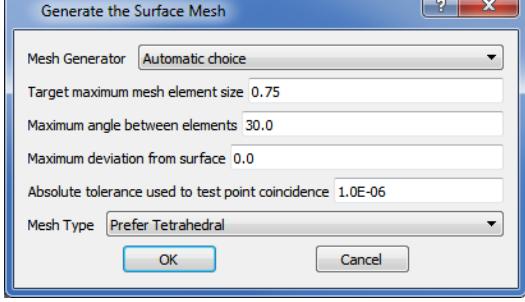
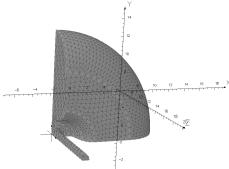
Press **OK** to apply these conditions.

## Model Body Creation and Meshing

Having set up the model, the Model Body can be created and the finite element mesh generated. Creating the Model Body is an important operation that forms the union of all the components, cuts away all but the minimum symmetric part and applies the symmetry conditions that have been specified.

After the Model Body has been created, the mesh can be generated. This is usually performed as a two-step process. In the first step, a surface mesh of two-dimensional elements is formed. In the second, a volume filling mesh of three-dimensional elements is created.

It is important that the mesh adequately represents not only the geometry but also the field structure. In this example, the maximum frequency specified is 800 MHz, which corresponds to a wavelength of 375 mm in free space. A minimum of 20 elements per wavelength will be used, so the maximum element size should be set to 18.75 mm (approximately 0.75 inches).

 <b>Create Model body</b>	After the model body has been created, only the minimum symmetric part of the model geometry is displayed.	
 <b>Generate Surface Mesh</b>	Set the <b>Target maximum mesh element size</b> to 0 . 75.  Select <b>OK</b> . The volume mesh may now be generated using  <b>Generate Volume Mesh</b> .	

Select **OK** to accept the default tolerance and create the volume mesh.

A tooltip displaying the number of elements in the mesh will be displayed if the mouse pointer is hovered over **Volume Mesh** in the status bar at the bottom of the Modeller window.

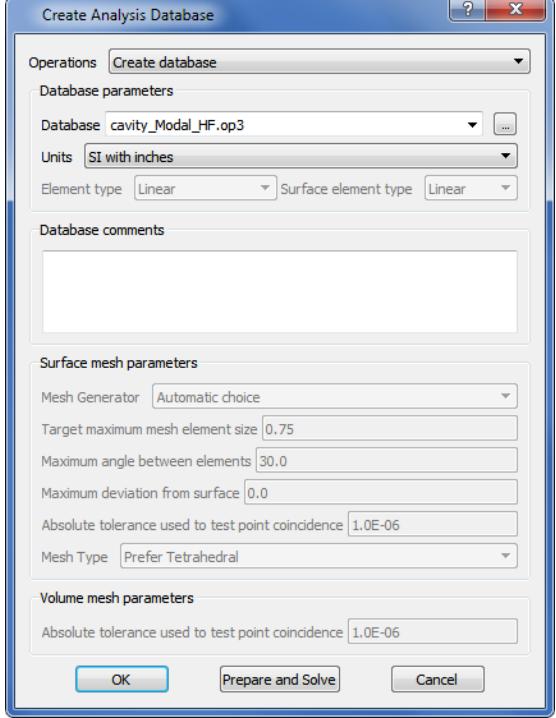
## Preparing and Running the Modal HF Analysis

At this point, the model is ready for analysis, and the **op3** database may now be prepared and solved. This contains all the data required by the Modal HF solver, and once calculated, the eigenmodes will be stored as separate solutions in this database.

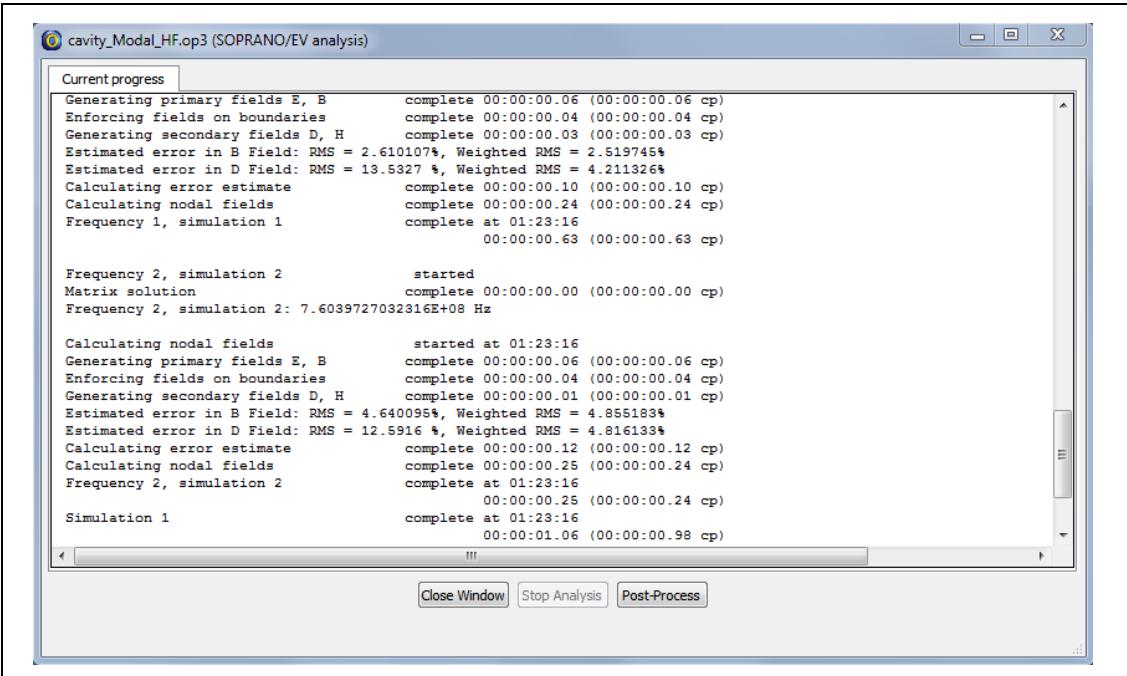


### Create Analysis Database

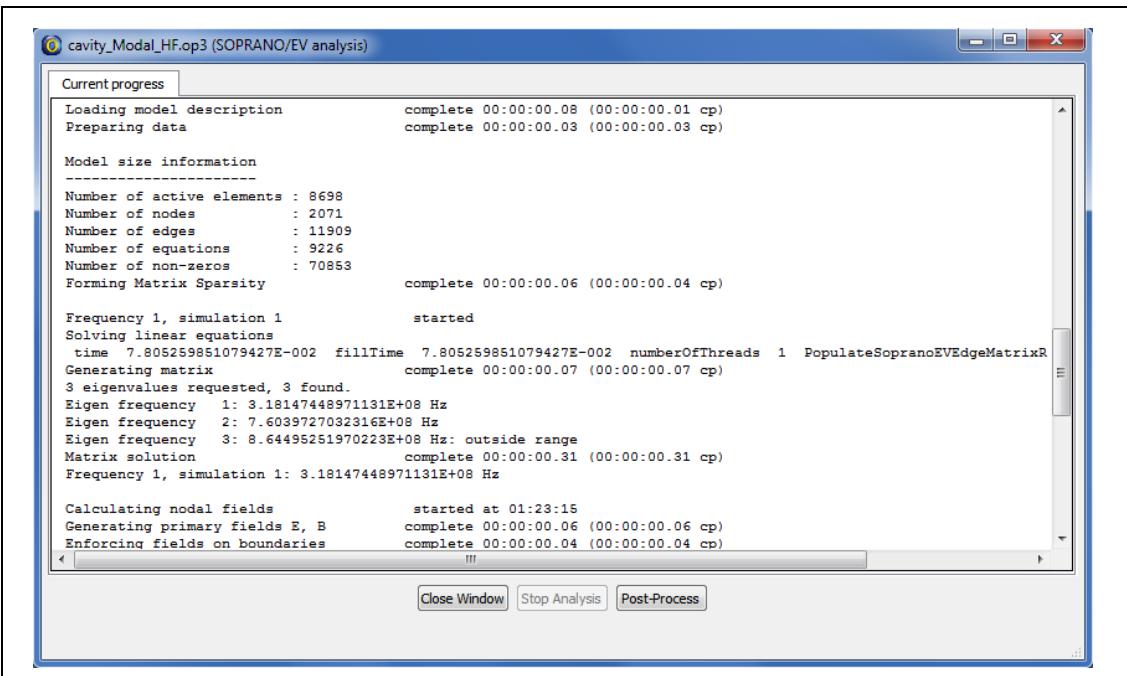
Enter a filename in the **Database** field and set the units to **SI with inches**. The database may then be prepared and the solver launched by pressing **Prepare and Solve**. Note that selecting **OK** at this point would create the database, but would not start the analysis.



A solver window will appear displaying the progress of the calculation. This should be complete in a few seconds, leaving the solver window as shown below. (Note that the display of the solver window depends on the Opera Manager **Batch Options** setting, **Show Solver Window**. The output from the analysis can also be seen in the file **cavity\_Modal\_HF.res**.)



The list of eigenvalues found may be examined by scrolling the Solver window display.



This shows three values, as requested. However, the third value is above the upper frequency limit of 800 MHz, and hence the field values associated with this mode will not have been calculated.

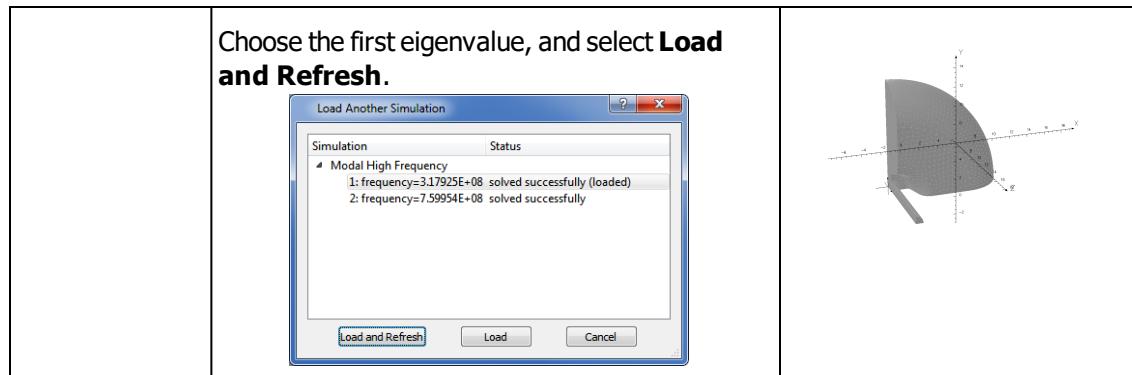
Note that the frequencies may not be exactly as shown in above, since the precise details of the finite element mesh generated by the Modeller is, to some extent, system dependent.

## Analysing the Results in the Post-Processor

The Opera-3d Post-Processor can now be used to display the field distribution of the eigenmodes and to calculate any functions dependent on the solution, such as the Q-factor of the cavity.

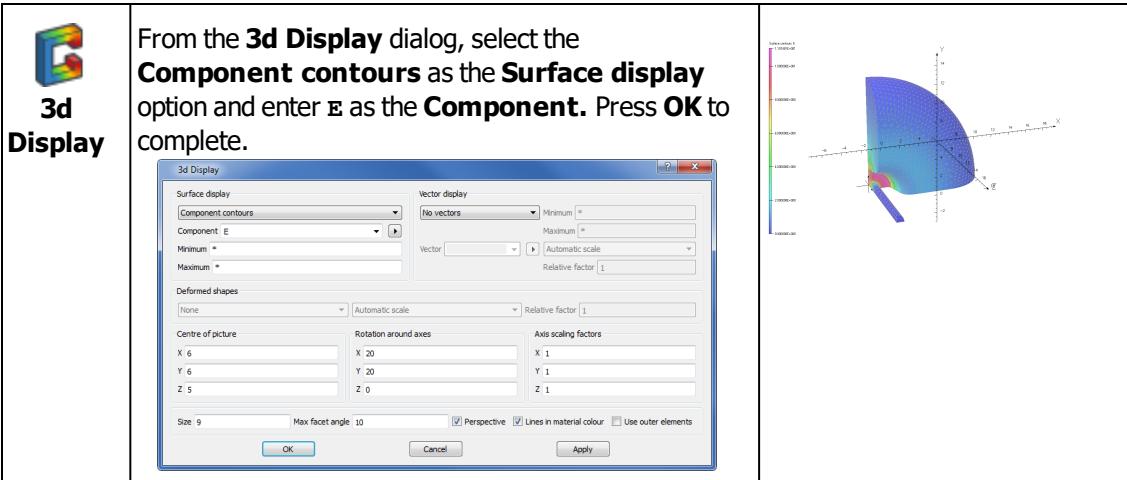
Press the **Post-Process** button on the solver window, or alternatively, double-click on the solved **cavity\_Modal\_HF.op3** database in the Manager file pane.

The Post-Processor starts in its own window and activates the model just analysed. The eigenvalues that have been found within the specified frequency range are shown. The user may select any of the listed solutions to load into the Post-Processor. It is assumed in the following discussion that the first eigenvalue has been selected initially.



Typical post-processing operations are shown in the following pages.

## Surface field distribution

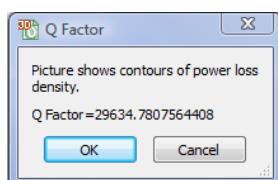
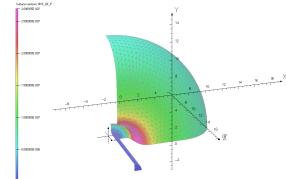


## Q-factor

Although the walls of the cavity have been represented with PEC boundary conditions (i.e. infinite conductivity), an approximation to the Q-factor of the cavity can be calculated using a more realistic conductivity value. This database was created with units of **SI with inches**. However, unit conversion may be performed in the Post-Processor. For example, if preferred, the unit of conductivity can be changed from S in<sup>-1</sup> to S m<sup>-1</sup> in the **Set Units** dialog, which is activated by selecting the **Units** tool-



button .

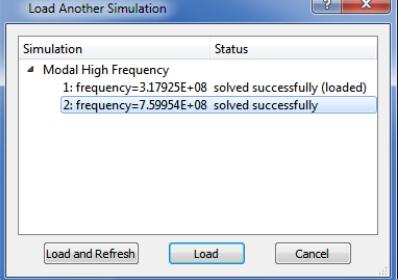
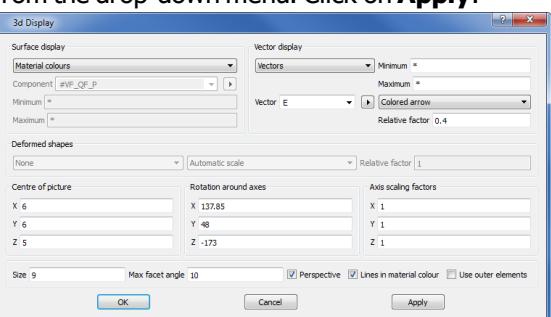
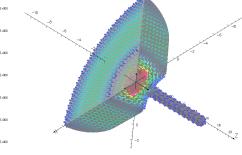
 <b>Cavity Q-factor</b>	<p>When the <b>Cavity Q-factor</b> toolbutton is selected, the user is prompted to supply the conductivity of the cavity walls; assuming that the conductivity units have been changed to S m<sup>-1</sup>, enter a value of 5.0E7.</p>  <p>On pressing <b>Calculate</b>, the program calculates the Q factor and displays the power loss on the walls.</p> 	
--	--	---

A Q factor of around 29600 can be expected from this model, though altering the mesh will have some effect on this.

## Field Vectors

When the Post-Processor was launched, the first, lowest frequency, eigenvalue solution was loaded.

Other simulations may be loaded from the database at any time, either by selecting the  **Select Case** toolbutton, or by using the  and  toolbuttons, which load the previous and next cases respectively. The first of these is assumed below.

 <b>Select Case</b>	<p>Choose the second solution and press <b>Load and Refresh</b>.</p> 	
 <b>Default Select and Refresh</b>	<p>During the Q-factor calculation, the PEC surface was automatically selected and the field component was changed. Pressing the  <b>Default Select and Refresh</b> toolbutton selects and re-displays all surfaces of the model geometry.</p>	
 <b>3d Display</b>	<p>Set the <b>Surface display</b> back to <b>Material colours</b>. Select <b>Vectors</b> from the <b>Vector display</b> drop-down list. In the <b>Vector</b> entry field select the electric field <b>E</b>, and <b>Coloured arrow</b> from the drop-down menu. Click on <b>Apply</b>.</p>  <p>Automatic scaling of the vectors is enabled by default. However, their displayed size may be modified if required, for example to investigate regions of high or low field, by adjusting the value of <b>Relative factor</b>. In this case, a factor of about 0.4 allows the peak field to be viewed clearly.</p> <p>When the display is as needed, select <b>OK</b> to close the dialog.</p>	

This completes the **Modal High Frequency** example.

The remaining part of this chapter describes the **Harmonic High Frequency** waveguide modelling example.

## Waveguide Modelling with Harmonic HF

---

### Building the Model Geometry

In the first variant of the model, the waveguide comprises only the interior of the waveguide - that is, the boundary of the model defines the interior walls of the guide and, in this model, the input port. The walls of the guide are assumed to be perfect conductors. The input port is modelled using an incident electric field boundary condition, which allows the required excitation to be imposed.

This model will be constructed initially for tetrahedral meshing. A modified version will then be made, in which parts of the geometry can be meshed with hexahedral or prismatic elements. After this model has been set up for analysis, a second variant will be described that adds finite conductivity metal walls to the waveguide.

The waveguide used in this example has the internal cross-section of a standard WR90 (WG16) waveguide, i.e. 22.86 x 10.16 mm, and is 200 mm long. At one end of the guide is a wedge of lossy dielectric, which is tapered to change the impedance slowly, and so absorb energy while giving very little reflection. This type of structure is used in practical devices to produce a matched termination.

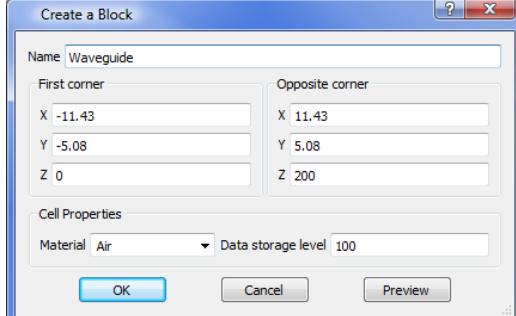
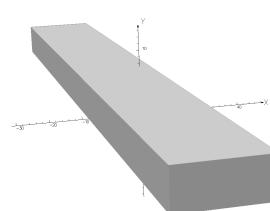
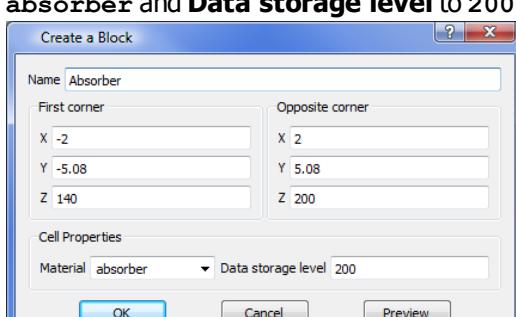
The recommended operating band of this guide is 8.2 to 12.4 GHz; the example here is excited by the dominant TE<sub>10</sub> mode at a frequency of 10 GHz.

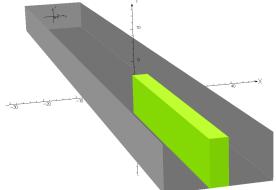
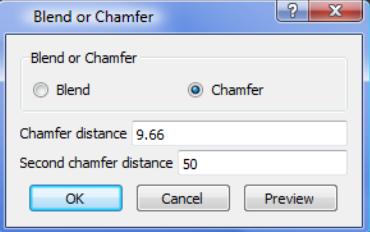
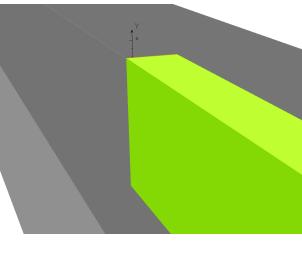
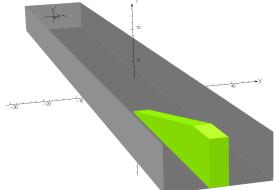
In this example, the intention is to discretize the model with a minimum of 15 finite elements per wavelength. Assuming free-space conditions, this would require a maximum element size of approximately 2 mm in air, and 1.4 mm in the dielectric wedge, which has a permittivity of 2. Note that the wavelength in the guide is longer than in free space - approximately 40 mm compared with 30 mm. So using the free-space wavelength criterion, modified by the permittivity where appropriate, results in greater than the minimum required element density. Tetrahedral meshing will be used initially. A second variant will then be considered which allows a mixture of tetrahedral and hexahedral meshing (a Mosaic mesh).

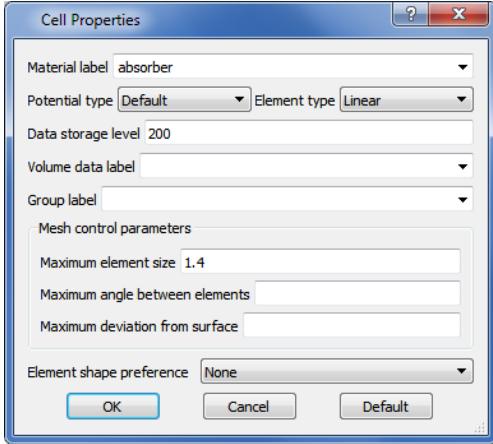
In the third variant of the mode, the waveguide wall is included as a metallic material of high conductivity ( $2.5 \times 10^7$  S m<sup>-1</sup>).

The lossy dielectric included in the model has a conductivity of 1 S/m. At 10 GHz, the skin depth in the absorber is approximately 5 mm. Hence the discretization of 1.4 mm adequately resolves the skin depth in the absorber. However, the skin depth in the highly conducting wall material, used in the second variant of the model, is less than a micron, and is impracticable to mesh. The use of the surface impedance boundary condition in this variant avoids this problem, and allows the simulation of such highly conductive materials.

The model is constructed from two blocks.

 <b>Create Block</b>	<p>First create the block representing the interior of the waveguide. Set the <b>Name</b> to <b>Waveguide</b>, and enter the coordinates of two opposite corners, at <math>(-11.43, -5.08, 0)</math> and <math>(11.43, 5.08, 200)</math>. Set the <b>Material</b> to <b>Air</b>, and <b>Data storage level</b> to 100.</p>  <p>Note that explicitly setting a <b>Material</b> label of <b>Air</b> is not strictly necessary. Air is the default material, and if the material of a cell is not set, Air will be assumed. Click on <b>OK</b>.</p>	
 <b>Create Block</b>	<p>Then create the block representing the lossy dielectric absorber. Set the <b>Name</b> to <b>Absorber</b>, and enter the coordinates of two opposite corners, at <math>(-2, -5.08, 140)</math> and <math>(2, 5.08, 200)</math>. Set the <b>Material</b> to <b>absorber</b> and <b>Data storage level</b> to 200.</p>  <p>Click on <b>OK</b>.</p>	

 <b>Pick Faces</b>  <b>Hide Entity</b>	<p>The absorber needs to be tapered, which may be achieved simply by adding a chamfer to one edge.</p> <p>First, it is useful to hide two faces of the waveguide to make the absorber visible. With <b>Hide Entity</b> and <b>Pick Face</b> selected, double-click on the waveguide top face, and the end face at z=200 mm.</p>	
 <b>Pick Edges</b>  <b>Pick Entity</b>	<p>With the picking options as shown to the left, right-click on the top edge of the absorber at z=140 mm and select <b>Blend or Chamfer</b>. On the dialog that appears, select the <b>Chamfer</b> radio button and enter the chamfer distances shown.</p> <p></p> <p>Click on <b>OK</b>.</p>	 

 <b>Pick Cells</b>	<p>Before unhiding the waveguide faces, the mesh size for the absorber should be set. Change the pick option to <b>Pick Cells</b>, right-click on the absorber and select <b>Cell Properties</b>. Set a <b>Maximum element size</b> of 1.4 mm. Leave the <b>Element shape preference</b> set to <b>None</b> for the moment. We will return to the options for the choice of mesh element types later in this example.</p>  <p>Press <b>OK</b> to close the dialog.</p>
 <b>Unhide Entities</b>	<p>This makes the previously hidden faces of the waveguide visible.</p>

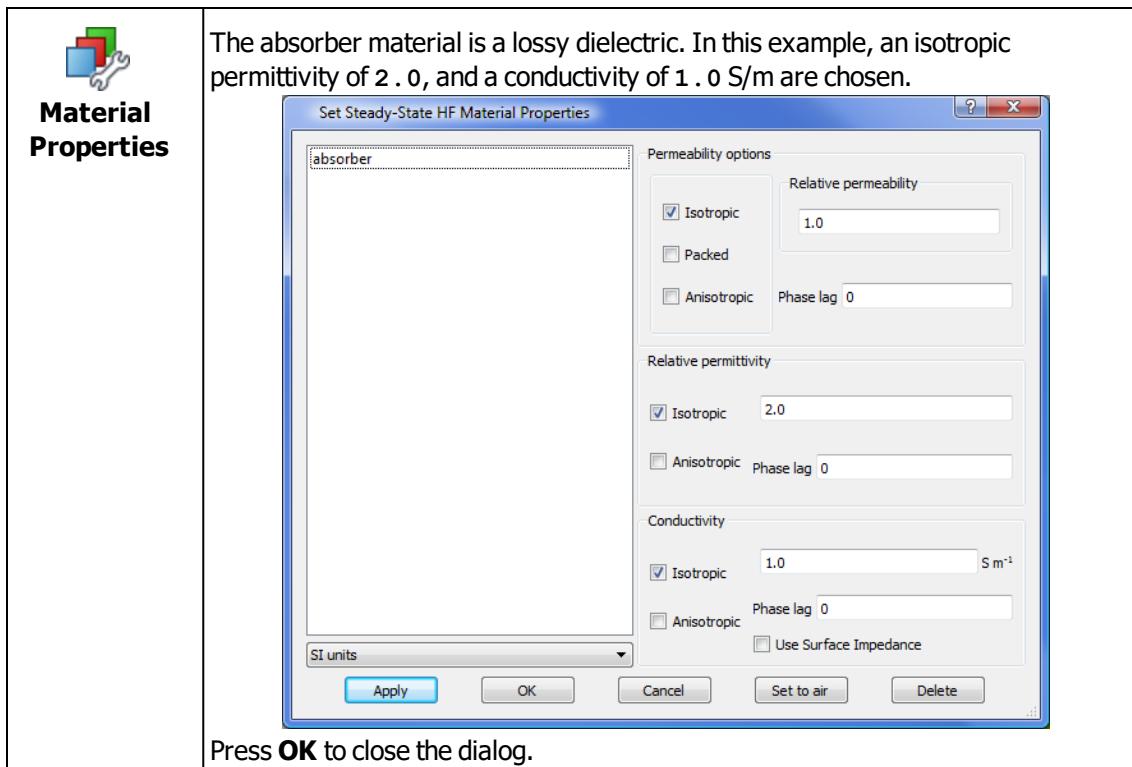
This completes the construction of the model geometry.

## Setting Options for Harmonic HF Analysis

The next step is to select the analysis module required, after which the appropriate analysis options and boundary conditions may be set. Set the analysis type to **Harmonic High Frequency**. In the configuration pane change the analysis frequency from the default value of 1 GHz ( $1.0\text{E}+09$  Hz) to 10 GHz ( $1.0\text{E}+10$  Hz).

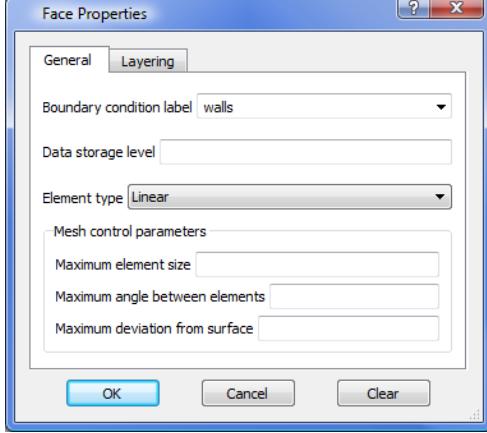
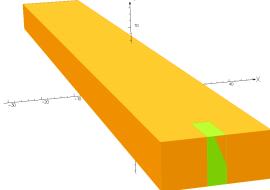
### Defining material properties

Once the type of analysis to be performed has been defined, the absorber may be given appropriate material properties.



## Boundary conditions

Before the boundary conditions can be applied on the faces of the waveguide, each face must be given an appropriate boundary condition label.

 <b>Pick Faces</b>	<p>Pick all faces of the waveguide block except the one at Z=0. Right-click to bring up the context menu and select <b>Face Properties</b>. In the <b>Boundary condition label</b> field, enter the label <b>walls</b>.</p>  <p>Click on <b>OK</b>. Then pick the face at Z=0 and set the <b>Boundary condition label</b> to <b>input</b>.</p>	
--	---	---

## Input port

In order to excite the waveguide, the boundary condition on its input face may be given in terms of the magnetic vector potential or the electric field. The latter will be used in this example. However, it should be noted that the electric field ( $E$ ) and magnet vector potential ( $A$ ) are simply related, i.e.

$$E = -j\omega A.$$

For the TE<sub>10</sub> mode, with the guide orientated as defined in the model, the electric field in the guide is given by

$$E_y = \operatorname{Re} \left\{ E_{y0} \cos \left( \left( \frac{x\pi}{l} \right) e^{j\omega t} \right) \right\} \quad (11.1)$$

where  $E_{y0}$  is the amplitude of  $E_y$  at  $x = 0$  and  $l$  is the width of the guide (22.86 mm in the present case). For this mode,  $E_x = E_z = 0$ .

The fields in a waveguide, or in any transmission line, may in general be described in terms of a forward and a backward propagating wave, the latter typically being caused by impedance changes in the line. The total field at any point on the line is the sum of the complex field amplitudes of these waves.

Steady-State HF calculates total field, but the electric field on the input face may be specified as either the total field or the incident (forward) field. If the total field is used, the effective amplitudes of the forward and backward waves in the guide are not known *a priori*. This means, for example, that in the general case, a simulation cannot be excited by a known input power.

If the incident field is used, the forward wave will have the user specified amplitude. However, when defining an incident electric field boundary condition, consideration must be given to the impedance conditions of the face on which the boundary condition is applied. In the present example, we want, in effect, the backward wave to be absorbed, not reflected, at the boundary. This simulates the case of a matched source, or, equivalently, an infinitely long length of guide behind the input face. This is achieved by placing an absorbing boundary condition on the input face, co-located with the incident electric field boundary.

In this example, an incident electric field of 1 V/m RMS will be used. This corresponds to a value of  $\sqrt{2}$  V/m for  $E_{y0}$  on the input face.

The absorbing boundary is characterized by an impedance scaling factor relative to free space. The wave impedance in the waveguide,  $Z_g$ , is given by

$$Z_g = \alpha Z_0 \quad (11.2)$$

where  $Z_0$  is the free-space impedance. For a TE mode,  $\alpha$  is given by the ratio of the free-space wavelength  $\lambda_0$  and the waveguide wavelength,  $\lambda_g$ , i.e.

$$\alpha = \frac{\lambda_g}{\lambda_0} \quad (11.3)$$

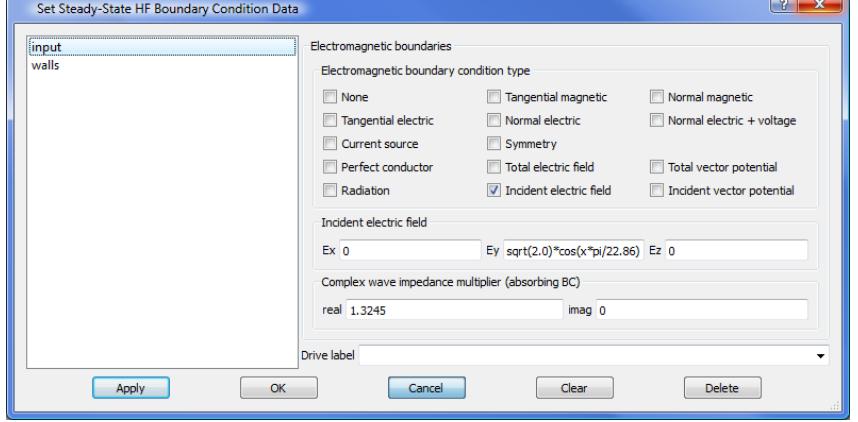
In a WR90 waveguide at 10GHz,  $\alpha$  is approximately 1.3245. Note that this is a real number because the waveguide has been assumed to be lossless. Practical waveguides usually have very small losses, and so the use of real impedance above cut-off is a good approximation.

The parameters indicated above are now used to define the excitation on the input face. Select **Boundary Conditions**.




**Boundary Conditions**

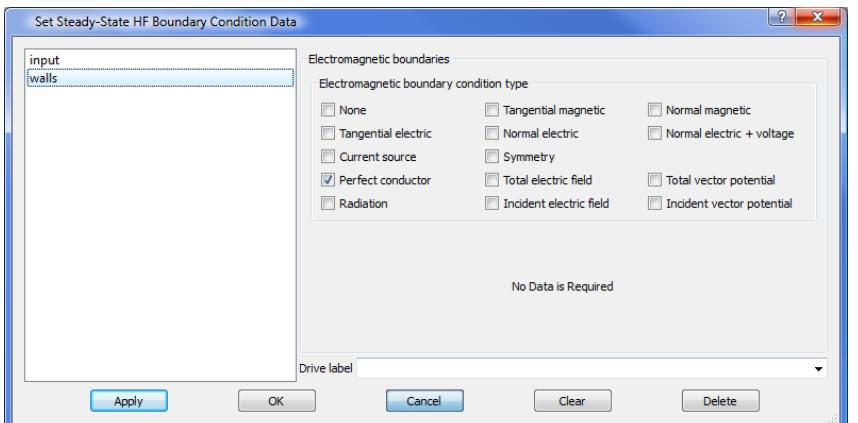
Select **input**, click on **Incident electric field** and set the components of the **Incident electric field** to be  $Ex = 0$ ,  $Ey = \sqrt{2} * \cos(x * \pi / 22.86)$  and  $Ez = 0$ .  
 Set the **real** and **imag** parts of the **Complex wave impedance multiplier**,  $\alpha$ , to be **1 . 3245** and **0** respectively.



Click on **Apply** to set these values and keep the dialog open.

## Wall boundary conditions


**Select **walls** and click on **Perfect conductor**.**



Click on **OK** to apply this and to dismiss the dialog.

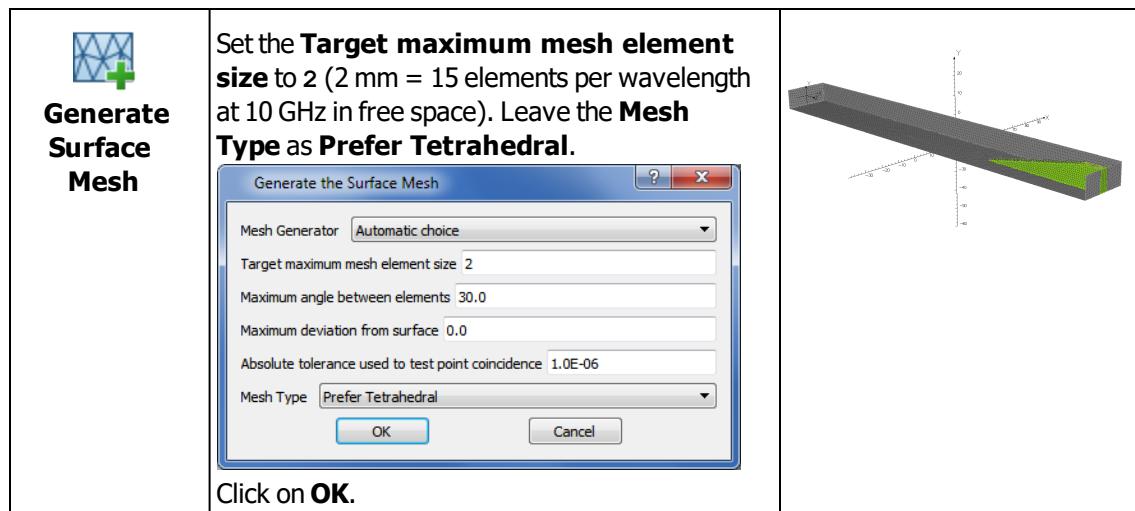
The model body should now be created using  **Create Model Body**.

## Meshing and Creating the Analysis Database

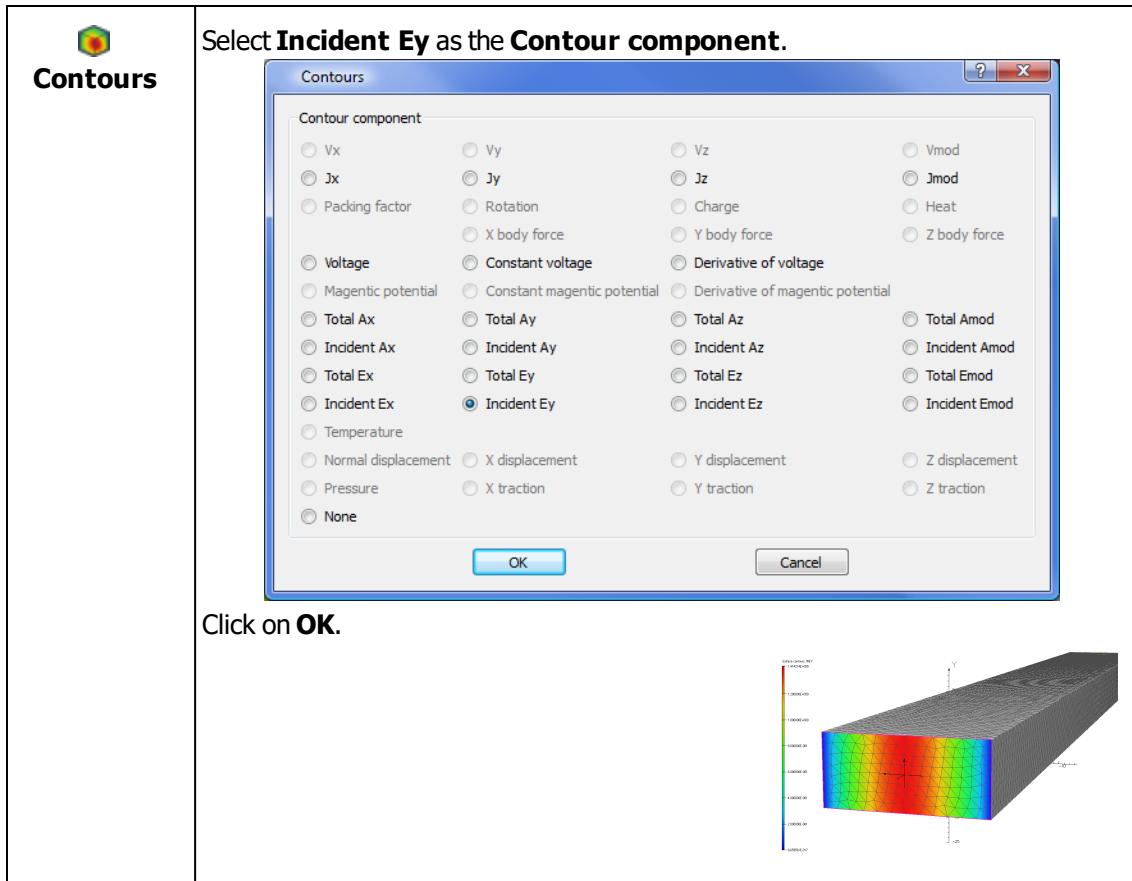
The Modeller offers a number of options for the type of finite element mesh, including tetrahedral, hexahedral and prismatic elements. The default is tetrahedral mesh elements, and this will be used initially.

### Tetrahedral meshing

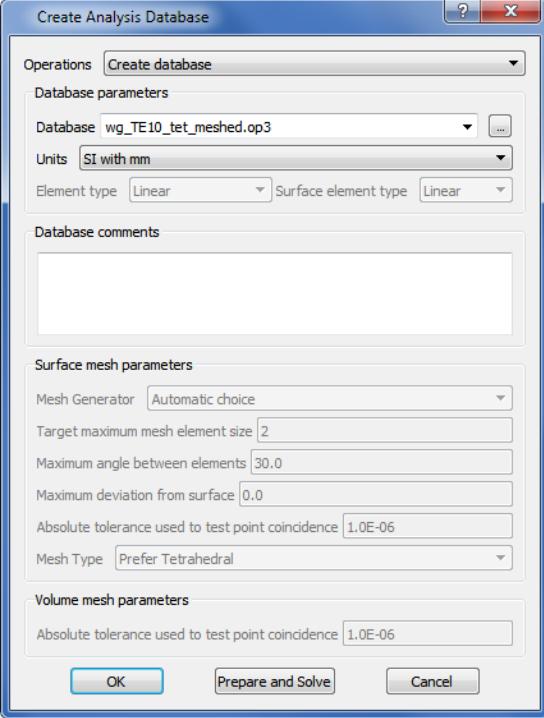
The finite element mesh must be fine enough to resolve the geometry of the model and to capture field variations that depend on the wavelength. For this model, a mesh size of 2mm is adequate for the waveguide, whereas a smaller mesh of 1.4mm, already defined, is used in the absorber to account for the effect of its permittivity.



Once the surface mesh has been generated, the contours of incident electric field on the input port can be displayed.



Generate the volume mesh using  **Generate Volume Mesh**. Accept the default settings and then create the database.

 <b>Create Database</b>	<p>Select a suitable <b>Database</b> file name, set the <b>Units</b> to <b>SI with mm</b>. Note that the <b>Element type</b> and <b>Surface element type</b> are both set to <b>Linear</b> by default. (Harmonic HF analysis supports only 1st order, i.e. linear, edge elements.)</p>  <p><b>Click on Prepare and Solve.</b></p>
---	--

When the analysis is complete, the solved database may be loaded into the Post-Processor. To do this, click on the **Post-process** button on the Solver window<sup>1</sup>. Alternatively, launch the Post-Processor from the Modeller, using **Post-Processor**, or from the menu or icon bar in the Manager.

Post-processing options will be described later in the example. Before this, a second variant of the model will be considered.

## Mosaic meshing

The model as configured above is suitable only for tetrahedral meshing. However, simple modifications to the way the model is built would allow at least part of the geometry to be meshed with either hexahedral or prismatic mesh elements.

A mesh that contains element types other than tetrahedra is known in Opera as a Mosaic mesh. A mosaic mesh may contain only hexahedra, or a mixture of hexahedral, prismatic, pyramidal and

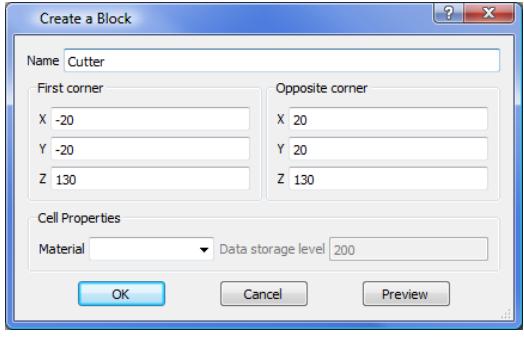
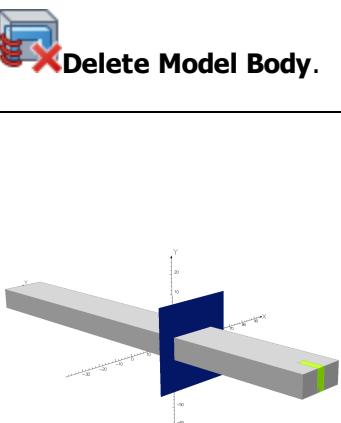
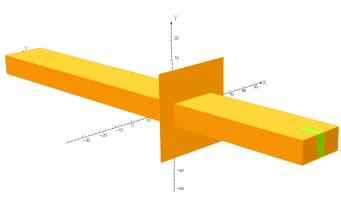
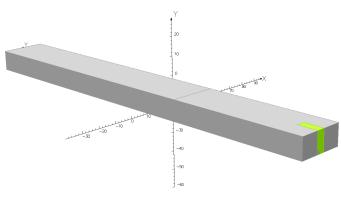
---

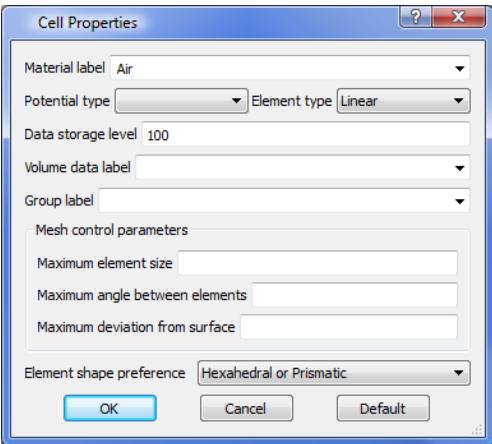
<sup>1</sup>Note that the display of the solver window depends on the Opera Manager Batch Options setting, Show Solver Window.

tetrahedral mesh element types. There are a number of strategies for preparing a model for mosaic meshing, and a very simple example is shown below. More information on selecting meshing options is given in [Meshing in the Modeller \[page 541\]](#), and in the Modeller **MESH** command section of the [\*\*Opera-3d Reference Manual\*\*](#).

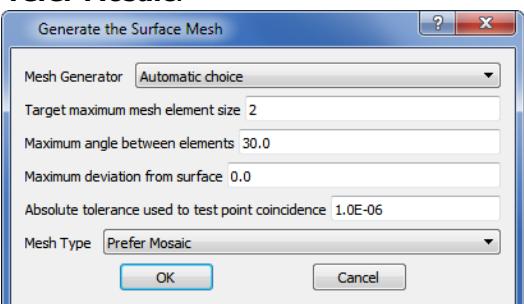
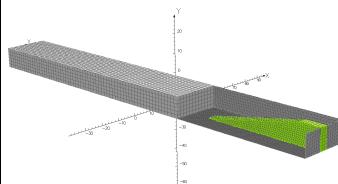
In this particular model, the structure of the free-space part of the waveguide is relatively simple. This region would require fewer hexahedral than tetrahedral mesh elements. To make this region hex-meshable, it must be a separate cell from the region containing the load. This can be achieved by cutting the waveguide into two cells.

To allow the model to be edited, return to component mode by using  **Delete Model Body**.

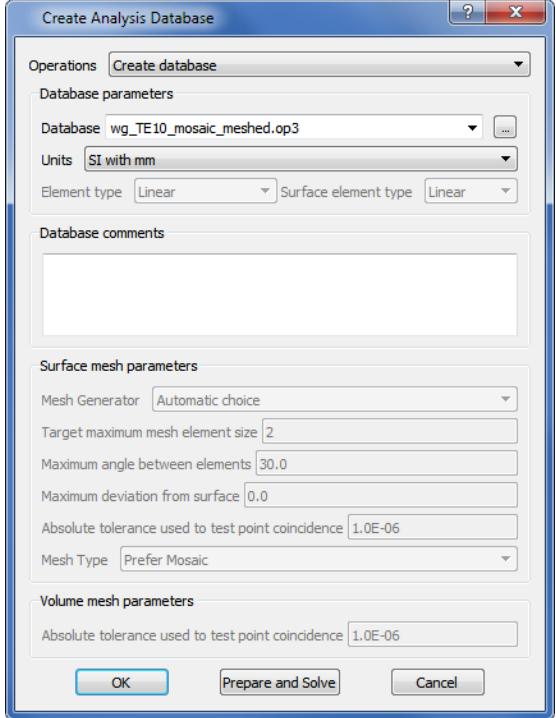
 <b>Create Block</b>	<p>Create a sheet at <math>z=130</math> by defining a block that goes from <math>(-20, -20, 130)</math> to <math>(20, 20, 130)</math>.</p>  <p>Click on <b>OK</b>.</p>	
 <b>Pick Bodies</b>	<p>Pick the waveguide, followed by the sheet.</p>	
	<p>Right-click and select <b>Combine Bodies &gt; Subtraction, without regularization</b> from the context menu.</p>	

 <b>Pick Cells</b>	<p>Select <b>Pick Cells</b>, and right-click on the waveguide section that extends from z=0 to z=130. From the context menu choose <b>Cell Properties</b>. Set the <b>Element shape preference</b> to <b>Hexahedral or Prismatic</b>.</p>  <p>Press <b>OK</b>.</p>	
 <b>Pick Cells</b>	<p>The absorber can be meshed with prisms. Select the absorber and repeat the above <b>Element shape preference</b> selection.</p> <p>Press <b>OK</b>.</p>	

The model body should now be created, using  **Create Model Body**. The surface mesh may then be generated. The absorber can be viewed by hiding faces of the waveguide in the region of z=130-200.

 <b>Create Model Body</b>	<p>Set the <b>Target maximum mesh element size</b> to 2 (2 mm = 15 elements per wavelength at 10 GHz in free space). Set the <b>Mesh Type</b> as <b>Prefer Mosaic</b>.</p>  <p>Click on <b>OK</b>.</p>	
---	---	--

Generate the volume mesh using  **Generate Volume Mesh**. Accept the default settings. The Analysis Database may now be created.

 <b>Create Database</b>	<p>Select a suitable <b>Database</b> file name, set the <b>Units</b> to <b>SI with mm</b>. Note that the <b>Element type</b> and <b>Surface element type</b> are both set to <b>Linear</b> by default. (Harmonic HF supports only 1st order, i.e. linear, edge elements.)</p>  <p>The dialog box shows the following settings:</p> <ul style="list-style-type: none"> <li><b>Operations:</b> Create database</li> <li><b>Database parameters:</b> <ul style="list-style-type: none"> <li>Database: wg_TE10_mosaic_meshed.op3</li> <li>Units: SI with mm</li> <li>Element type: Linear</li> <li>Surface element type: Linear</li> </ul> </li> <li><b>Database comments:</b> (Empty)</li> <li><b>Surface mesh parameters:</b> <ul style="list-style-type: none"> <li>Mesh Generator: Automatic choice</li> <li>Target maximum mesh element size: 2</li> <li>Maximum angle between elements: 30.0</li> <li>Maximum deviation from surface: 0.0</li> <li>Absolute tolerance used to test point coincidence: 1.0E-06</li> <li>Mesh Type: Prefer Mosaic</li> </ul> </li> <li><b>Volume mesh parameters:</b> <ul style="list-style-type: none"> <li>Absolute tolerance used to test point coincidence: 1.0E-06</li> </ul> </li> </ul> <p>Buttons at the bottom: OK, Prepare and Solve, Cancel.</p> <p>Click on <b>Prepare and Solve</b>.</p>
---	--

Once the analysis is complete, the solved database may be loaded into the Post-Processor. Some of the post-processing operations that may be performed are described later in this section. Before this, a third model variant will be considered.

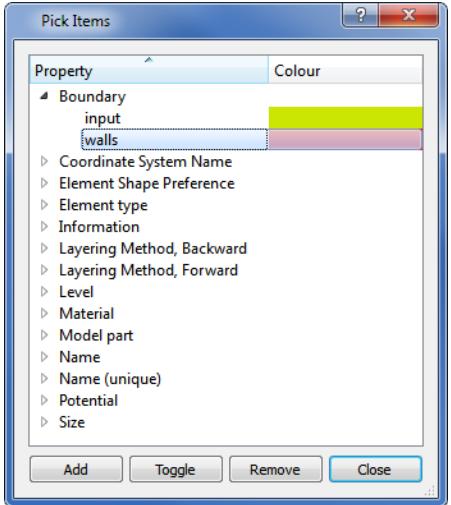
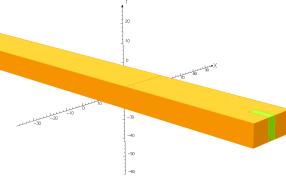
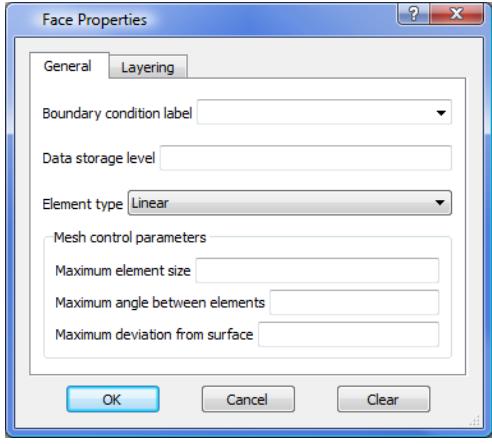
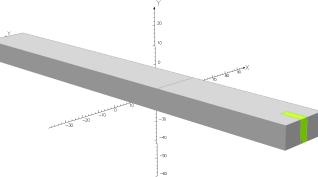
## Model Variant - Walled Waveguide

The existing model may be modified easily to add a lossy metal wall. The first step is to return to

Component mode by deleting the model body, using  **Delete model body**.

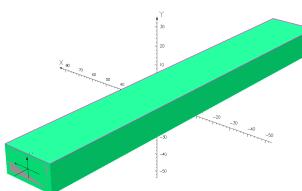
When a metal is used as the waveguide, the perfect conductor boundary on the faces of the waveguide air block is not required, and should be removed. Select the faces with this boundary condition

using the  **Pick Entities by Property** toolbutton.

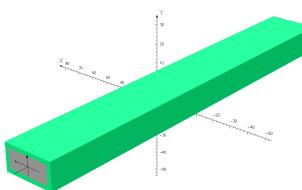
 <b>Pick by Property</b>	<p>Select the <b>Boundary</b> property <b>walls</b>, press <b>Add</b> and <b>Close</b>.</p> 	
	<p>Right-click and select <b>Face Properties</b> from the context menu. Select <b>Clear</b> to clear the boundary property definition.</p>  <p>Press <b>OK</b>.</p>	

The waveguide walls will be produced by trimming the air blocks representing the inside of the waveguide, from a larger solid block. The inside faces of the walls will later be labelled to allow their selection in the Post-Processor.

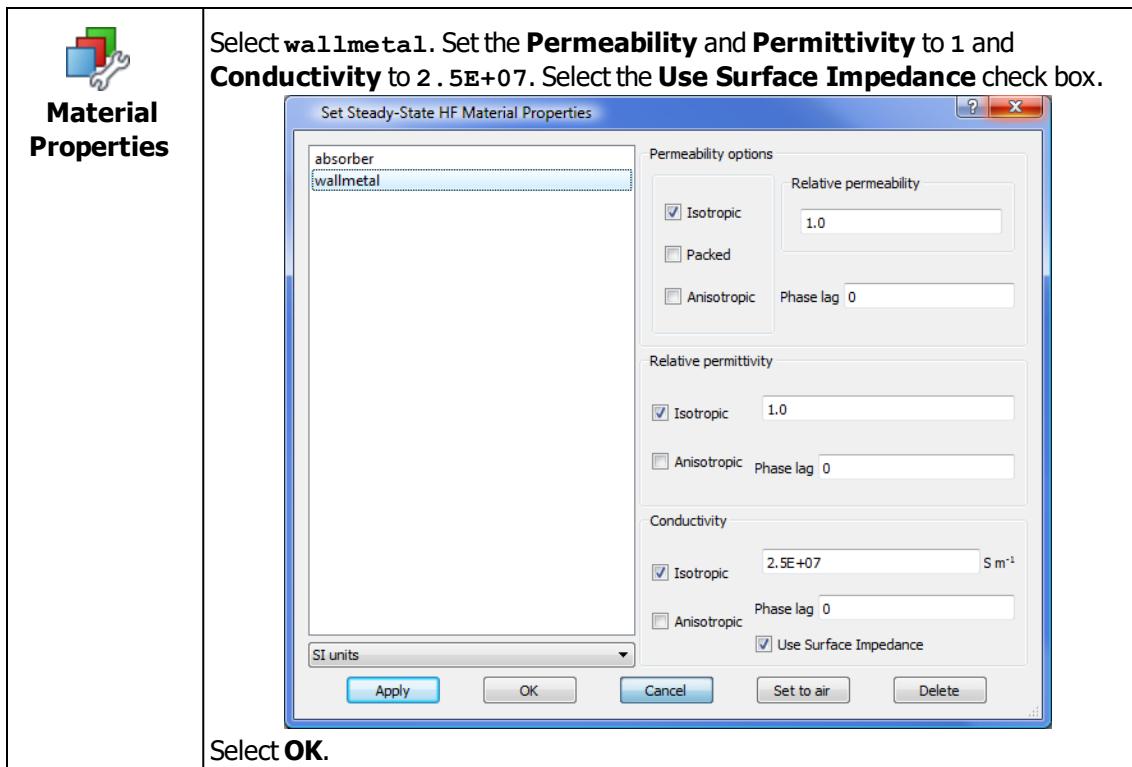
Create the wall block using the  **Block** toolbutton.

 <b>Create Block</b>	<p>Enter the <b>Name Guidewalls</b>, corner points <b>(-13.43, -7.08, 0)</b> and <b>(13.43, 7.08, 202, Material wallmetal</b> and <b>Data storage level 50</b>.</p> <p><b>Create a Block</b></p> <p>Name: Guidewalls</p> <p>First corner:</p> <p>X: -13.43 Y: -7.08 Z: 0</p> <p>Opposite corner:</p> <p>X: 13.43 Y: 7.08 Z: 202</p> <p>Cell Properties:</p> <p>Material: wallmetal Data storage level: 50</p> <p>OK Cancel Preview</p> <p>Select <b>OK</b>. The chosen dimensions of the block will result in a waveguide wall thickness of 2 mm in the final model.</p> 
--	--

The waveguide air block may now be used to trim the wall block. Select  **Pick Bodies**.

 <b>Pick Bodies</b>	<p>First, double-click on the <b>Guidewalls</b> block, then on the waveguide air block. Right-click and select <b>Combine Bodies &gt; Trim overlap, with regularization</b>.</p> 
---	--

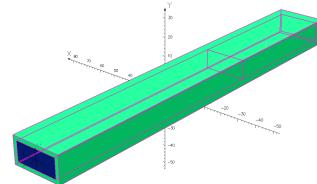
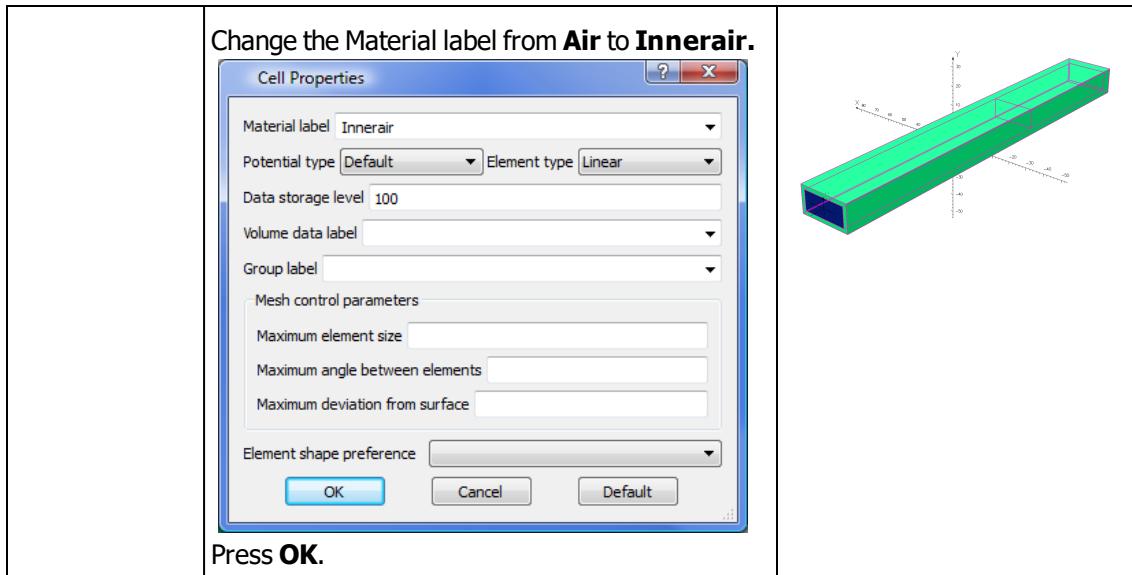
The properties of the wall material need to be defined. Aluminium alloy or brass are commonly used for waveguide devices. For both materials, the alloy grades typically employed have conductivities of approximately  $2.5 \times 10^7 \text{ S m}^{-1}$ .



At the chosen simulation frequency, 10GHz, the skin depth in a material of conductivity  $2.5 \times 10^7 \text{ S m}^{-1}$  is approximately 1 micron. The use of the surface impedance formulation allows materials of such high, but finite, conductivity to be included in the simulation without the mesh needing to resolve the skin depth.

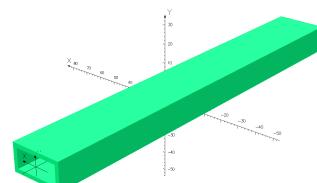
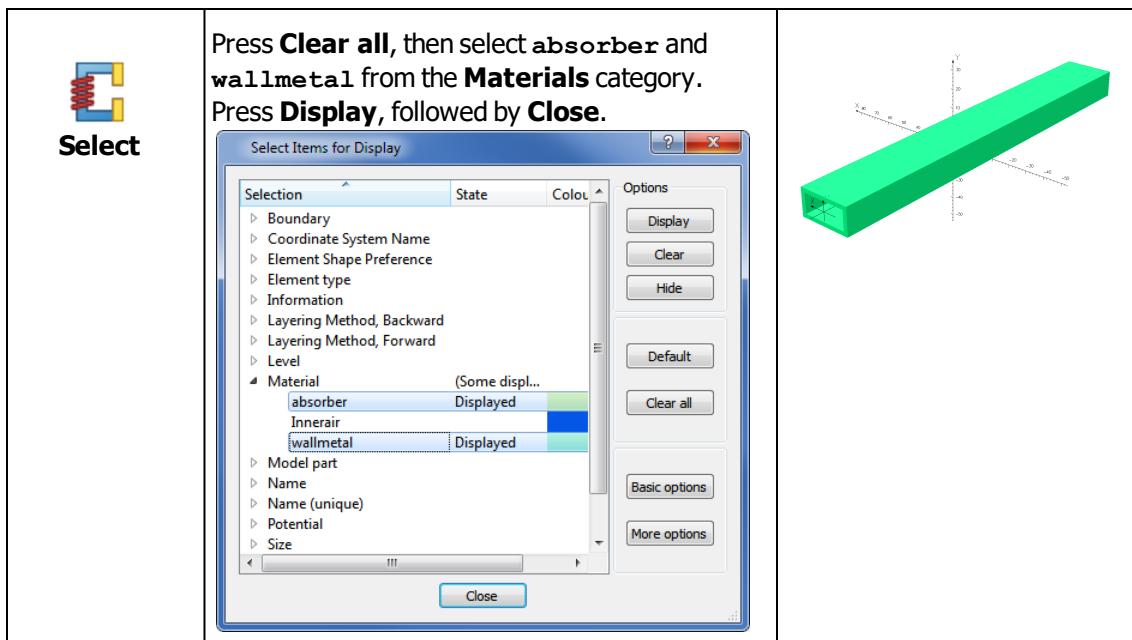
The inside of the waveguide is filled by air blocks. To allow the display of fields on their surfaces in the Post-Processor, they should be given a different material name. Select **Pick by Property**. Select **Air** under the **Material** category and select **Add**, followed by **Close**.

Right-click and choose **Cell Properties**.

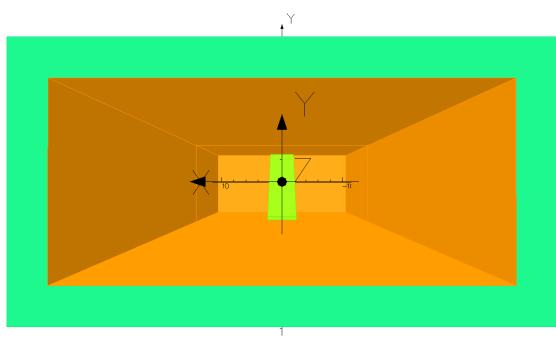


To allow the display of the inside faces of the waveguide walls in the Post-Processor, user labels may

be added to the faces. Initially select only the walls and absorber for display. Select the  **Select** toolbutton.



Manipulate the view of the geometry to display the inside of the waveguide. Select the  **Pick Faces** toolbutton.

 <b>Pick Faces</b>	<p>Pick all the inside faces of the waveguide walls by double-clicking on each in turn. Alternatively, move the mouse cursor over each wall in turn and press <b>p</b> on the keyboard to pick.</p> 
--	--

Select **Additional Labels**.

<b>Additional Labels</b>	<p>Enter the <b>Additional label</b> <b>Insidewalls</b>.</p>  <p>Select <b>Add</b> to add the label and close the dialog.</p>	
--------------------------	--	--

All required modifications have now been made. Use  **Create Model Body** to create the model body. Generate the surface and volume meshes with the same parameters as used in the first variant of the model. The model is now ready to be run. Select  **Create Analysis Database**. Enter a new name for the **Database** and select **Prepare and Solve**.

Once the analysis is complete, the solved database may be loaded into the Post-Processor. Some of the post-processing operations that may be performed are described in the next section, for both this model and the first variant.

## Post-Processing Harmonic HF Results

### First and second model variants

In the previous section, two meshing strategies, tetrahedral and mosaic, were used when creating the solution databases for the first and second variants of the waveguide model. Post-processing of the results is independent of meshing strategy, and the figures illustrating the post-processing of this model assume that the solved database from the mosaic-meshed model has been loaded into the Post-Processor.

In an empty waveguide carrying the  $TE_{10}$  mode, only the y-component of electric field and x- and z-components of magnetic field exist. In this example, the presence of the absorber will generate other components of the fields. However, these represent coupling into evanescent waveguide modes, i.e. they are below cut-off and so will not propagate in the guide. Hence most of the discussion that follows concentrates on the amplitudes, ( $E_0, H_0$ ), moduli ( $E, H$ ), and the dominant  $E_y, H_x$  and  $H_z$  components of the field.

### Complex solutions

The **Harmonic High Frequency** solver provides the full complex solution to the wave equation. Consequently, the solution contains both real and imaginary parts. System variables in the Post-Processor can be prefixed by the letters **R** and **I** respectively to specify these. Alternatively, the phase angle in the AC cycle can be specified with 0 degrees corresponding to the real part of the solution and -90 degrees corresponding to the imaginary part. For example:

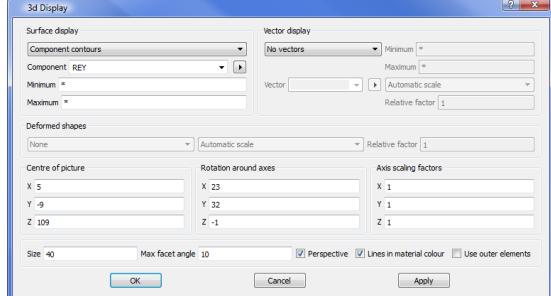
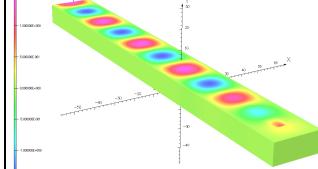
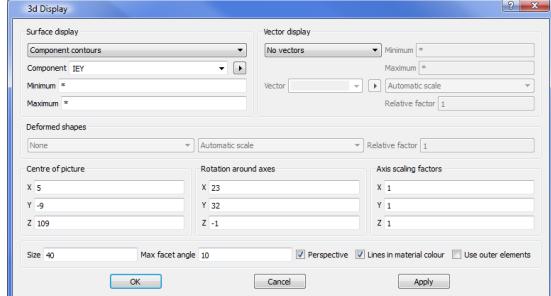
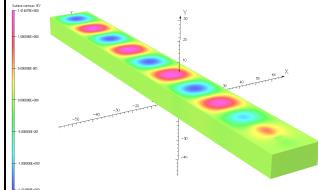
- **REY** is the real part of the y-component of **E**.
- **IEY** is the imaginary part of the y-component of **E**.
- **EY** is the value of the y-component of **E** evaluated at the current setting of AC time.
  - If AC time is zero, **EY=REY**.
  - If AC time is -90, **EY=IEY**.

Some of these features are illustrated below.

### Fields on the geometry



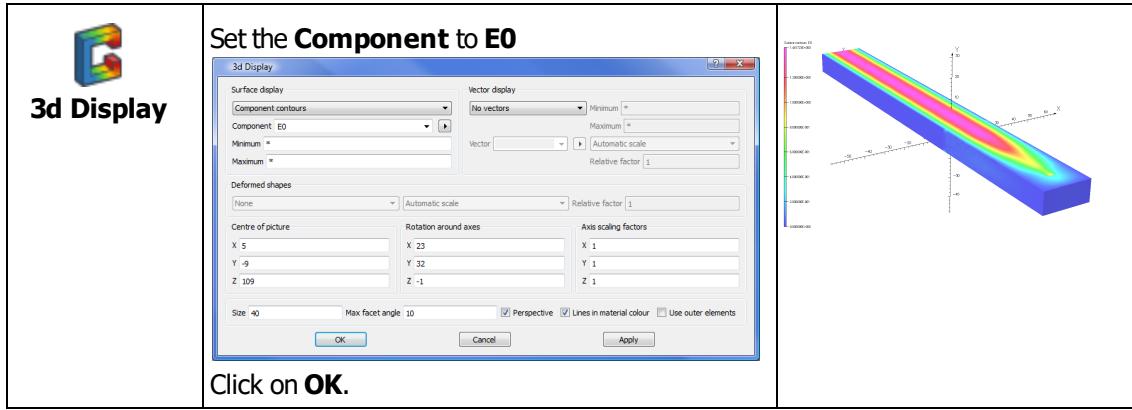
Fields on the geometry may be displayed by pressing the **3d Display** toolbutton. Before examining the fields on the geometry, it is useful to switch off the view of the mesh by selecting the **Outline View of Model** toolbutton. Note that this button toggles the mesh view on and off.

 <b>3d Display</b>	<p>Select <b>Component contours</b> from the <b>Surface display</b> drop-down list and set the <b>Component</b> to <b>REY</b> (note that the names of the field components are not case sensitive).</p>  <p>Click on <b>Apply</b>.</p>	
 <b>3d Display</b>	<p>Change the <b>Component</b> to <b>IEY</b>.</p>  <p>Click on <b>Apply</b>.</p>	

Comparing the above pictures from the software, it can be seen that the electric field distribution moves by approximately one quarter period between the real and imaginary components, and that the components are of similar value. This implies that there is primarily a propagating field in the guide.

In the part of the guide near the input port, the field amplitude is constant, but reduces in the region of the absorber.

This can be seen more easily if the electric field amplitude, **E0**, (i.e. the dyadic modulus) is plotted.



This behaviour is as expected for a loaded waveguide, where almost all of the power is dissipated in the absorbing load, rather than being reflected from the end wall of the waveguide.

## Vector display

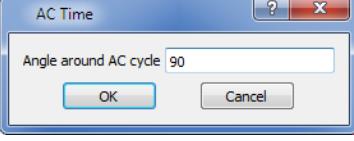
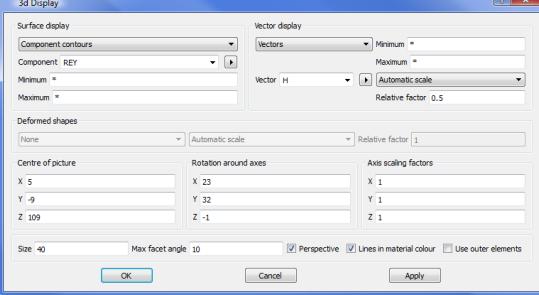
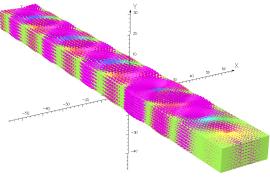
As stated earlier in this example, in the  $TE_{10}$  mode both the x- and z-directed components of the magnetic field exist. Consequently, it can be convenient to display the magnetic field as vectors on the walls of the guide.

The real part of the magnetic field strength is related to the imaginary part of the displacement current,  $\mathbf{D}$ , since from Maxwell's equations

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}. \quad (11.4)$$

Similarly, the imaginary part of the magnetic field strength is related to the real part of the displacement. Because of this relationship, it is not unusual for the real part of one field and the imaginary part of the other to be shown in the same picture. For example, contours of the real part of the electric field and vectors of the imaginary magnetic field are displayed simultaneously below, the vectors being parallel to the contours.

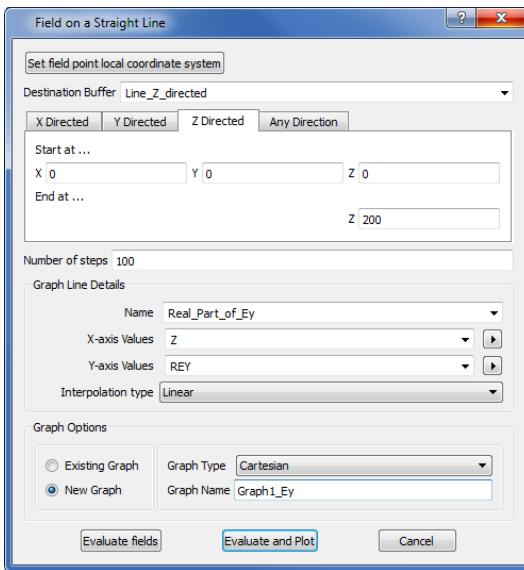
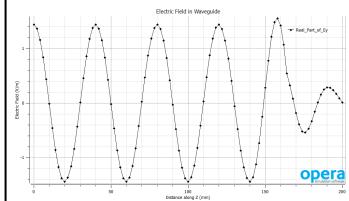
To display the vectors of the imaginary part of the magnetic field, set the phase angle around the AC cycle to 90.

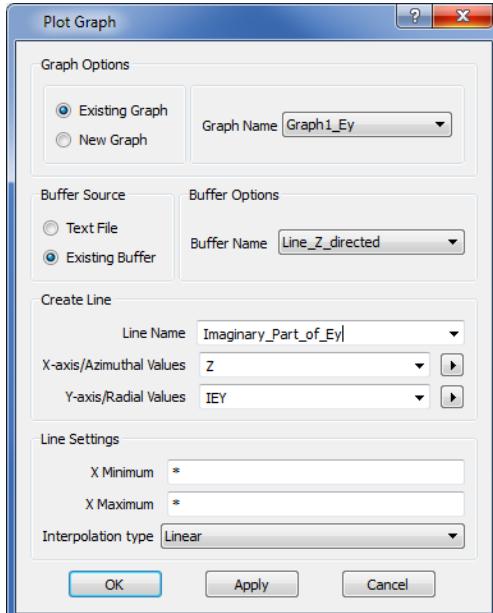
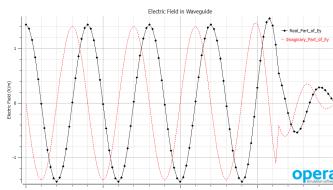
 <b>Set AC Time</b>	<p>Set the <b>Angle around AC cycle</b> to 90.</p>  <p>Click on <b>OK</b>, then on  <b>Refresh</b>.</p>	
 <b>3d Display</b>	<p>Set the <b>Component</b> to <b>rey</b>. Click on <b>Vectors</b> from the <b>Vector display</b> drop-down list and enter <b>H</b> as the <b>Vector component</b>. Choose <b>Automatic scale</b> from the adjacent from the drop-down list and set <b>Relative factor</b> to <b>0.5</b>.</p>  <p>Click on <b>Apply</b>. The size of the vectors can be adjusted by modifying the value of the <b>Relative factor</b> and clicking on <b>Apply</b> again. Click on <b>OK</b> to close the dialog.</p>	

Set the time back to 0 by selecting the  **AC Time** toolbutton so that later field calculations give the real parts (unless system variables starting with **I** are used - see [Complex solutions \[page 420\]](#)).

## Fields along a line

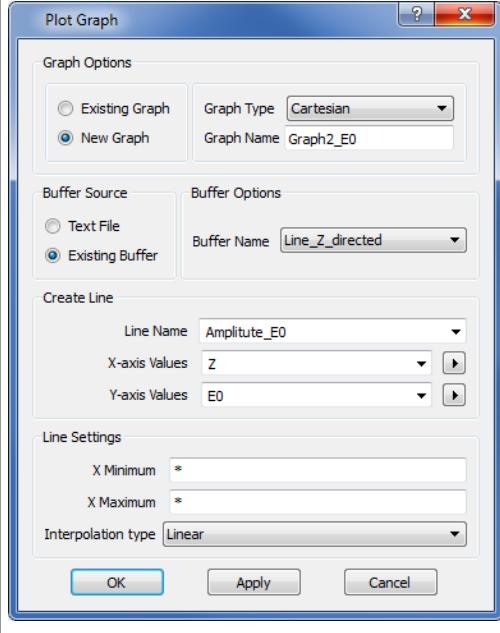
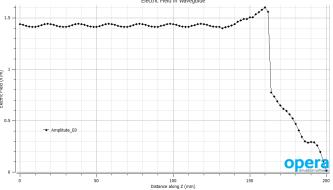
It is instructive to examine the electric field on a line along the centre of the guide by plotting the real and imaginary parts of  $E_y$ .

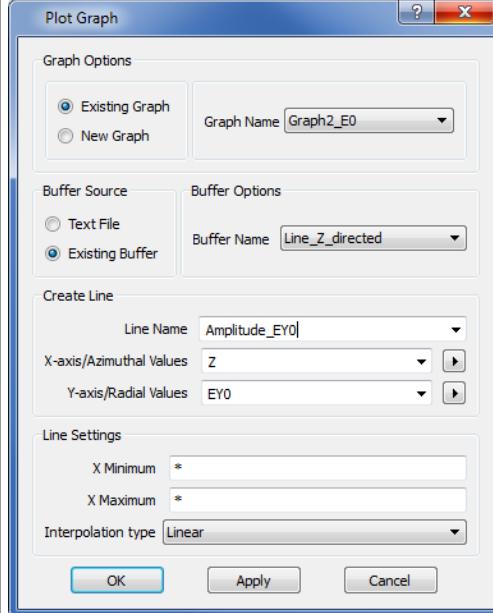
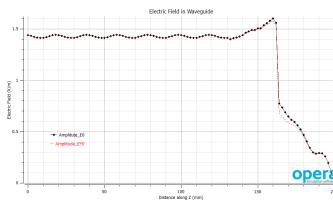
 <h3>Fields on a Line</h3> <p>In <b>Destination Buffer</b>, give a name <b>Line_z_directed</b>.      Click on the <b>Z Directed</b> tab. Start at <b>(0, 0, 0)</b> and end at <b>Z = 200</b>. Set the <b>Number of steps</b> = 100.</p> <p>In the <b>Graph Line Details</b> section, set the <b>Name</b> to <b>Real_Part_of_Ey</b>. For the <b>X-axis Values</b>, type <b>z</b>, and for the <b>Y-axis Values</b>, type <b>REy</b>.</p> <p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <b>Graph1_Ey</b>.</p>  <p><b>Click on Evaluate and Plot.</b></p>	
---	--

 <b>Plot Graph</b>	<p>There are two options to open the following dialog:</p> <p>Either use the <b>Plot Graph</b> toolbutton or stay in the Graphs widget and perform a right-click on the buffer with the previously given name <b>Line_Z_directed</b>. Then select <b>Create Line From Buffer</b>.</p> <p>In <b>Graph Options</b>, select <b>Existing Graph</b>, with the <b>Graph Name</b> <b>Graph1_Ey</b>.</p> <p>In <b>Buffer Source</b>, select <b>Existing Buffer</b>, and confirm the <b>Buffer Name</b>.</p> <p>In the <b>Create Line</b> section, set the <b>Line Name</b> to <b>Imaginary_Part_of_Ey</b>. For the <b>X-axis Values</b>, type <b>z</b>, and for the <b>Y-axis Values</b>, type <b>IEy</b>.</p> 	
--	---	---

It can be seen that up to about  $z=140\text{mm}$ , the start of the absorbing load, the peaks of the real and imaginary components of the electric field have a very similar amplitude, and are in quadrature, as expected for a single forward-propagating wave. The peak value (approximately  $1.4\text{V/m}$  or  $1\text{V/m}$  RMS) is as expected from the applied excitation. The wavelength in the guide can be seen to be approximately  $40\text{mm}$ , again as expected.

Beyond the start of the load, the fields begin to decay; the discontinuity at about  $165\text{mm}$  is where the line along which the fields are evaluated intersects the sloping face of the load. Although the previous plots suggest that there is only a forward-propagating wave, a better evaluation may be made by considering the amplitude, **E0** or **EY0**.

 <b>Plot Graph</b>	<p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <b>Graph2_E0</b>. In <b>Buffer Source</b>, select <b>Existing Buffer</b>, and confirm the <b>Buffer Name</b>. In the <b>Create Line</b> section, set the <b>Line Name</b> to <b>Amplitude_E0</b>. For the <b>X-axis Values</b>, type <b>z</b>, and for the <b>Y-axis Values</b>, type <b>E0</b>.</p>  <p>The dialog box shows the following settings: <b>Graph Options:</b> Existing Graph (radio button), Graph Type: Cartesian, Graph Name: Graph2_E0. <b>Buffer Source:</b> Existing Buffer (radio button), Buffer Name: Line_Z_directed. <b>Create Line:</b> Line Name: Amplitude_E0, X-axis Values: z, Y-axis Values: E0. <b>Line Settings:</b> X Minimum: *, X Maximum: *, Interpolation type: Linear. Buttons at the bottom: OK, Apply, Cancel.</p>	 <p>The plot titled "Electric Field in Waveguide" shows the electric field magnitude (E0) versus distance along Z (cm). The x-axis ranges from 0 to 250 cm, and the y-axis ranges from 0 to 1.0. The data series, labeled "Amplitude_E0", shows a periodic oscillation between approximately 0.8 and 1.0 until about 200 cm, where it drops sharply to near zero. The word "opera" is visible in the bottom right corner of the plot area.</p>
--	--	---

 <b>Plot Graph</b>	<p>In <b>Graph Options</b>, select <b>Existing Graph</b>, with the <b>Graph Name</b> <code>Graph2_E0</code>.</p> <p>In <b>Buffer Source</b>, select <b>Existing Buffer</b>, and confirm the <b>Buffer Name</b>.</p> <p>In the <b>Create Line</b> section, set the <b>Line Name</b> to <code>Amplitude_EY0</code>. For the <b>X-axis Values</b>, type <code>z</code>, and for the <b>Y-axis Values</b>, type <code>EY0</code>.</p>  <p>Click on <b>OK</b>.</p>	
--	---	---

As can be seen in [Figure 11.6](#), the values of `E0` and `EY0` are essentially identical up to the region of the absorber, where additional modes are generated. The values are almost, but not quite constant, up to  $z=140$  mm. The variation is caused by a small-amplitude reflection from the load and end wall.

One important performance metric of any waveguide device such as this, is the value of the reflected amplitude compared with the incident amplitude. This is usually specified by its voltage standing wave ratio (VSWR), voltage reflection coefficient or return loss. These are different ways of expressing the same quantity, and we may calculate each of them from the peak-to-peak ripple of the field in the first, empty, part of the guide.

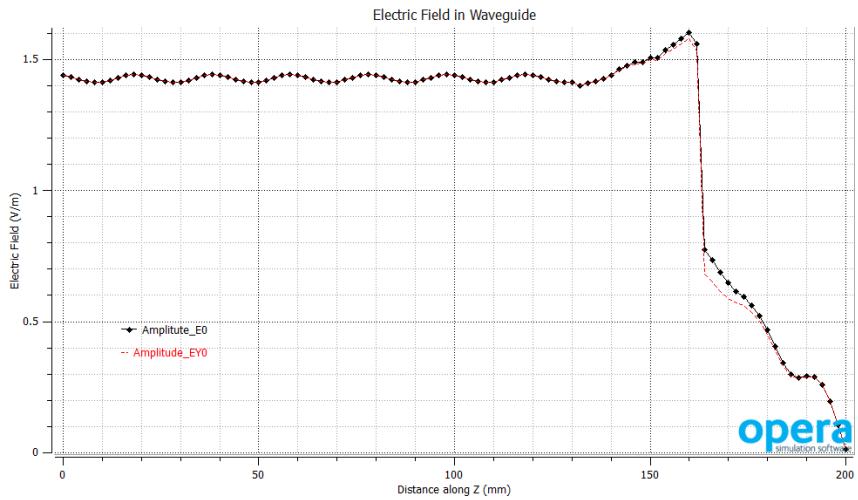
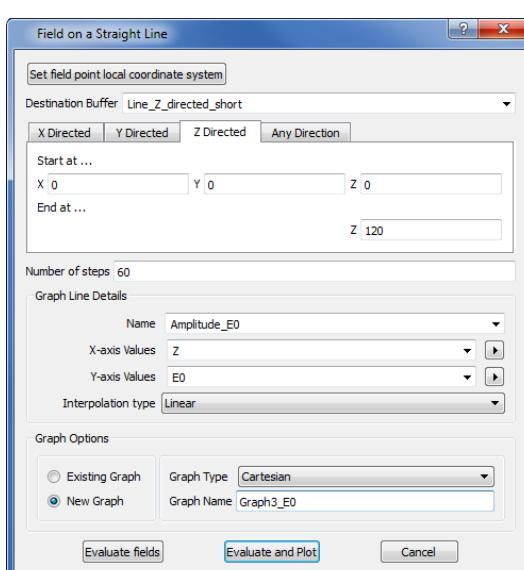
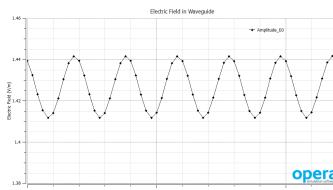


Figure 11.6 Comparison of  $E0$  and  $EY0$  in the waveguide

 <h3>Fields on a Line</h3>	<p>In <b>Destination Buffer</b>, give a name <code>Line_z_directed_short</code>. Click on the <b>Z Directed</b> tab. Start at <code>(0, 0, 0)</code> and end at <code>Z = 120</code>. Set the <b>Number of steps</b> = 60.</p> <p>In the <b>Graph Line Details</b> section, set the <b>Name</b> to <code>Amplitude_E0</code>. For the <b>X-axis Values</b>, type <code>z</code>, and for the <b>Y-axis Values</b>, type <code>E0</code>.</p> <p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <code>Graph3_E0</code>.</p> <p></p> <p>Click on <b>Evaluate and Plot</b>.</p>	
---	--	---

When the fields are plotted, the minimum and maximum values on the plot are assigned to the `line_MINIMUM` and `line_MAXIMUM` system variables where `line` is the derived<sup>1</sup> from the name of the line on the graph. The VSWR may be evaluated directly as the ratio of these; the reflection coefficient and return loss are derived from this. The calculations may be performed on the command line, using the following expressions:

```
$constant #VSWR Amplitude_E0_MAXIMUM/Amplitude_E0_MINIMUM
$constant #refcoeff (#VSWR-1)/(#VSWR+1)
$constant #retloss ABS(20*log10(#refcoeff))
```

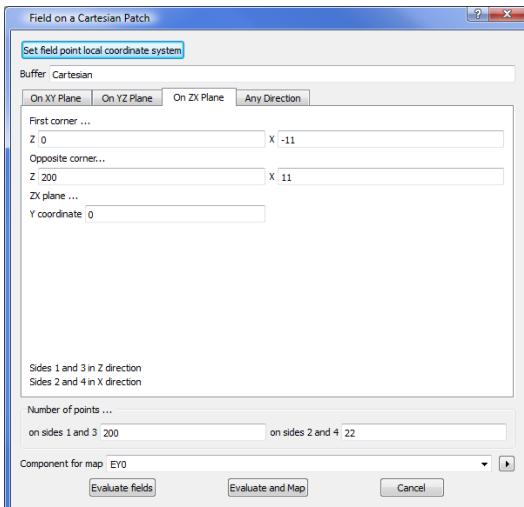
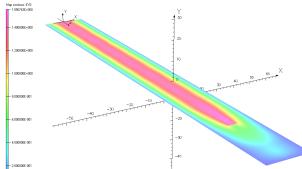
---

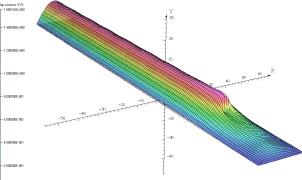
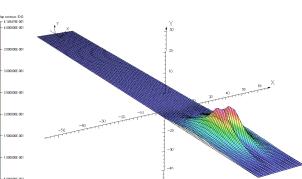
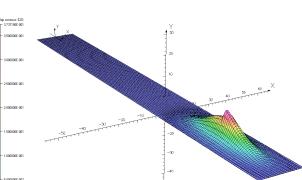
<sup>1</sup>If the line name contains spaces, these will be replaced by underscores in the variable names.

A convenient way to perform the arithmetic is to include these commands in a command file.

### Fields on a plane

All components of the fields inside the waveguide may be displayed on arbitrary planes. A ZX plane along the mid-plane of the waveguide is used as an example.

<p> <b>Cartesian Patch</b></p>	<p>Click on the <b>On ZX Plane</b> tab. Define a patch from <math>z=0</math>, <math>x=-11</math> to <math>z=200</math>, <math>x=11</math> at <math>y=0</math> and with 200 by 22 points. Enter <b>EY0</b> as the <b>Component for map</b>.</p> <p></p> <p>Click on <b>Evaluate and Map</b>. Since the plane is inside the waveguide block, the <b>Solid View of Model</b> and <b>Outline View of Model</b> toolbuttons should be pressed to hide the geometry.</p> <p></p>
---	--

 <b>Contour or Vector Map</b>	<p>Enter the required component (<b>EY0</b>) in the <b>Component</b> field. Clicking the <b>Histogram</b> radio button and checking the <b>Grid through data points</b> box gives an enhanced impression of the variation over the plane.</p> <p><b>Contour or Vector Map</b></p> <p>Data: Internal buffer or table file <input checked="" type="checkbox"/> Data from internal buffer</p> <p>Buffer: Cartesian</p> <p>Contour Maps Map style: Histogram Component: EY0 Number of lines: 10 Minimum: Height of histogram SIZE/3 Maximum: <input checked="" type="checkbox"/> Show grid through data points</p> <p>Vector Maps No vectors Vector: H Automatic scale Relative factor: 1</p> <p>Options <input checked="" type="checkbox"/> Replace existing maps <input type="checkbox"/> Print values to ip file</p> <p>OK      Apply      Cancel</p> <p>Click on <b>OK</b> to display the field.</p> <p>Also shown on the right are the peak fields, Ex0 and Ez0 on the plane. As can be seen, the absorber introduces components of the electric field in x and z, but these are unable to propagate, and so are localized around the load.</p>	  
---	--	--

## Power dissipation

The instantaneous ohmic loss  $P_l$  in any region is given by

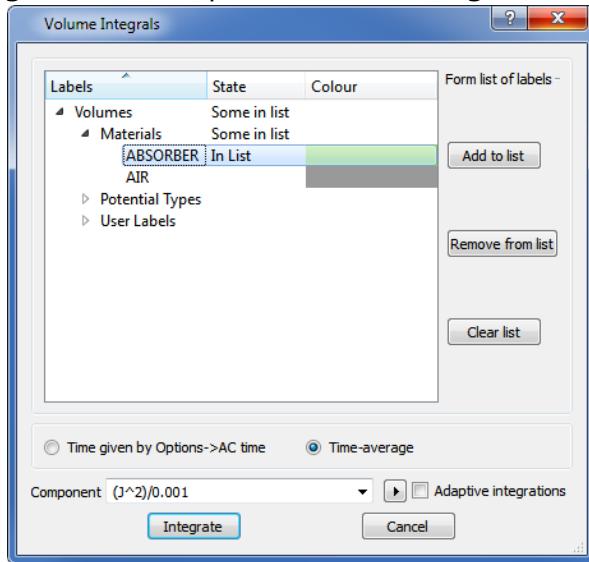
$$P_l = |\mathbf{J}|^2 / \sigma, \quad (11.5)$$

where  $\mathbf{J}$  is the local instantaneous current density and  $\sigma$  is the conductivity. The total average power dissipation is the time-average of  $P_l$ , integrated over all regions of finite conductivity. The Post-Processor has a general purpose command, the **VOLUME** command, for performing such calculations.

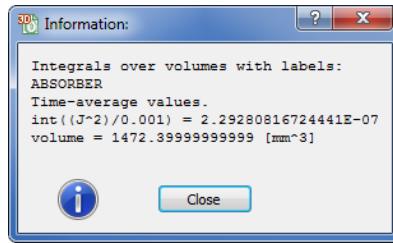
This may be accessed by pressing on the  **Other Volume Integrals** toolbutton.

### Other Volume Integrals

To calculate the power dissipation in only the absorber, Press on **Clear list** then select the label **ABSORBER** in the **Volumes -> Materials** list and press **Add to list**. Enter the function  $(J^2) / 0.001$  in the **Component** field. Ensure that the **Time-average** radio button is picked and select **Integrate**.



Once the calculation has been performed, an information box is displayed showing the time-averaged volume integral, which is the power loss in the absorber in watts.

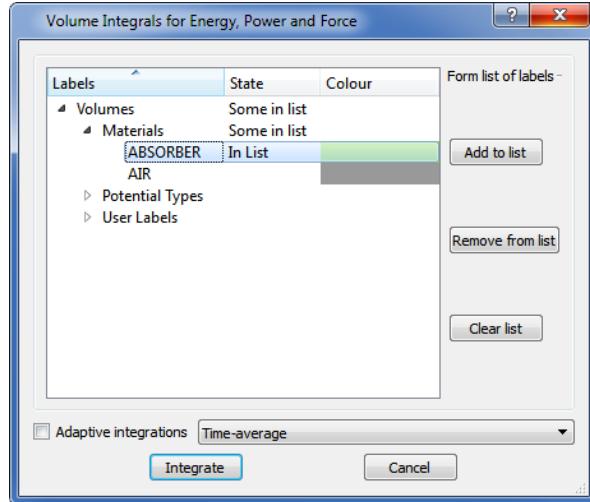


It should be noted that the choice of units that was made when creating the solver database was **SI with mm**. Hence the conductivity of  $1 \text{ S m}^{-1}$ , used in the Modeller, is represented by default in the Post-Processor as  $0.001 \text{ S mm}^{-1}$ .

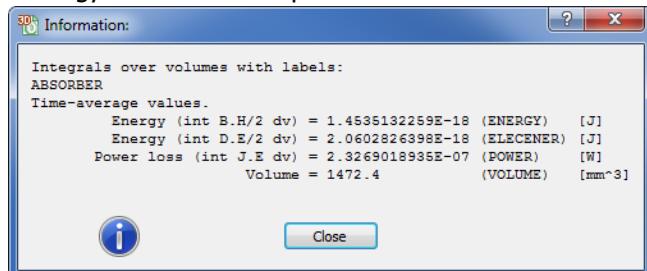
The power dissipated in the absorbing load may also be evaluated with the **ENERGY** command. This calculates the loss in a similar way to that shown above, but in addition gives the energy contained in the electric and magnetic fields.

  
**Energy,  
Power and  
Force**

As before, to calculate the power dissipation only in the absorber, press on **Clear list** then select the label **ABSORBER** in the **Volumes -> Materials** list and press **Add to list**. Ensure that the **Time-average** is chosen from the drop-down list, then select **Integrate**.



Once the calculation has been performed, an information box is displayed showing the energy in the fields and power loss in the absorber.

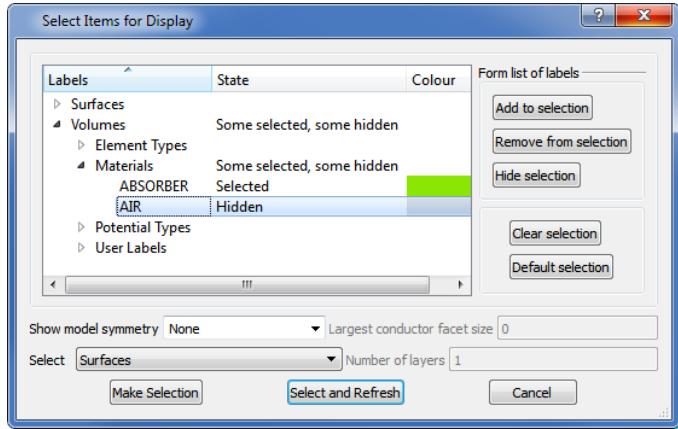
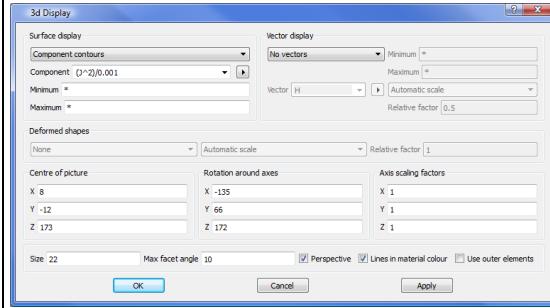
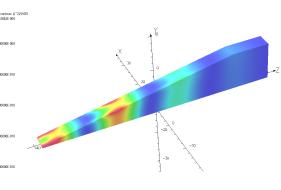


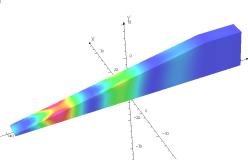
As can be seen, the power loss result is similar, but not identical to that calculated previously. The reason for this difference is that the **VOLUME** command uses nodally averaged fields, whereas the **ENERGY** command uses values derived directly from the solution potentials. The latter gives a better representation of the enhanced loss around sharp corners.

For a waveguide of the size used here, a 1 V/m RMS excitation corresponds to an input power of approximately  $2.33 \times 10^{-7}$  W. Hence we can see that most of the power is dissipated in the load. The rest is reflected, giving the small VSWR calculated earlier. In principle the VSWR could be calculated from the difference between the known input and the dissipated power. However, this is not an appropriate technique in this case, since small errors in the dissipated power calculation will result in large relative errors in the apparent reflected power.

Note that in this example, the rest of the model is lossless, so selecting only the absorber is not strictly necessary. However, the calculation would then be performed unnecessarily over lossless regions.

The distribution of instantaneous dissipated power may be visualized by displaying the ohmic loss on the surface of the absorber.

 <b>Select</b>	<p>The <b>Select</b> command may be used to hide the waveguide, allowing the absorber to be seen. Having pressed the  <b>Select</b> toolbutton, select <b>AIR</b> in the <b>Volumes -&gt; Materials</b> list, and press <b>Hide selection</b>.</p>  <p>Press <b>Select and Refresh</b>.</p>
 <b>3d Display</b>	<p>In the 3d Display dialog, ensure that the <b>Component contours</b> and <b>No vectors</b> are selected, and that the Component is <math>J^2/0.001</math>.</p>   <p>Press <b>OK</b>.</p>

 <b>Set AC Time</b>	<p>The power dissipation at other points in the RF cycle may be displayed by changing the time.</p> <p>Using the  <b>AC Time</b> toolbutton , set the <b>Angle around AC cycle</b> to, for example, 90.</p> <p>Press <b>OK</b>, then press the  <b>Refresh</b> button , to refresh the display.</p>	
---	---	---

A sequence of displays at different times can be displayed using a simple command file to give an animation of the power dissipation, or of any other field component. An example command file could include commands such as:

```
SET TIME=0
$DO INDEX=#t START=0 FINISH=3600 INCREMENT=5.0
SET TIME=#t
THREED OPTION=REFRESH
$END TYPE=DO
SET TIME=0
THREED OPTION=REFRESH
```

This would display the selected component over 10 AC cycles at 5 degree intervals.

This completes the post-processing for the first and second variants of the Steady-State HF example. Close the Post-Processor.

### Third model variant

All of the post-processing operations, such as calculating fields and losses, that were performed on the first and second models may be performed on the third variant. In addition, the contribution to the power dissipation (i.e. the loss of the waveguide) made by the finite conductivity waveguide walls may be evaluated.

Load the solved database for the simulation on this model into the Post-Processor by selecting **Post-Process** from the Solver window, or by using one of the methods available in the Manager.

### Fields on the geometry

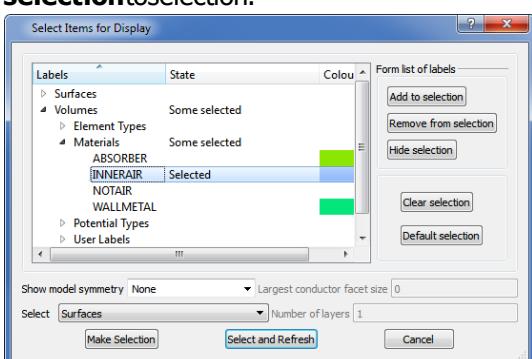
Fields may be displayed on the surface of all parts of the geometry that have appropriate labels that allow them to be selected for display. In this model, these include, in particular, the **Innerair**, **absorber** and **wallmetal** material labels, and the **Insidewalls** label on the inner faces of the waveguide wall.

The default view displays the guide walls, the inside air block and the absorber, with an outline of the mesh. As before, the view of the fields is improved if the mesh is hidden. Select the  **Outline View of Model** toolbutton to toggle this off.

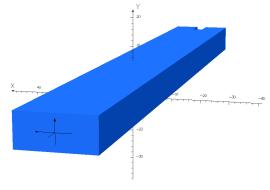
To view the fields on the surface of the inside air block, the guide walls may be removed from the display. Use the  **Select** toolbutton, to achieve this.



Press **Clear selection** first, then select **INNERAIR** label from **Volumes -> Materials**. Press **Add to selection** to selection.

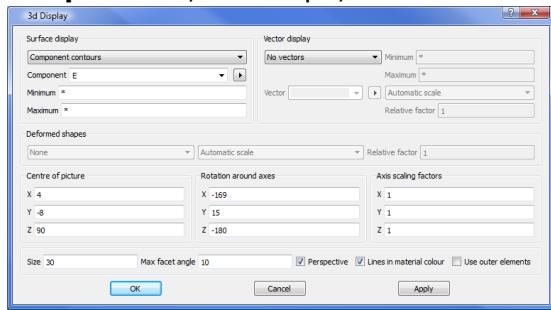
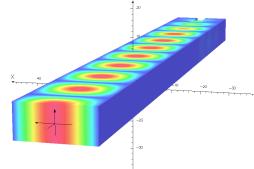
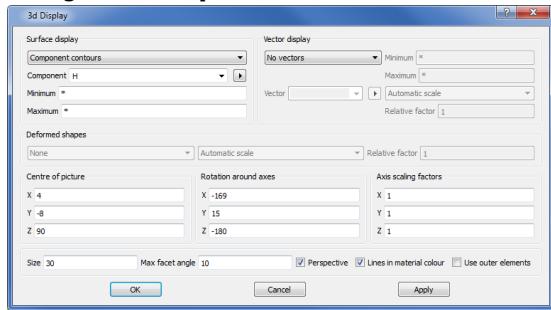
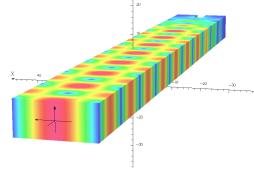


Click on **Select and Refresh**.

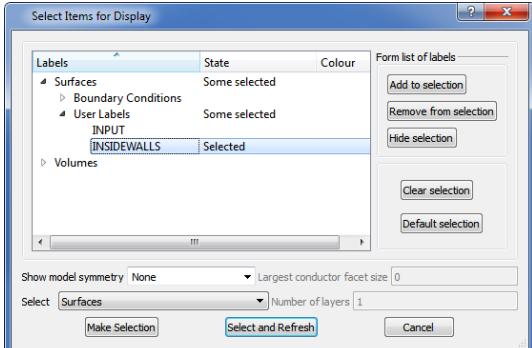
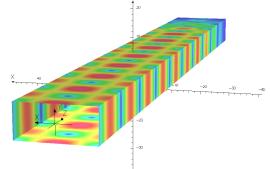
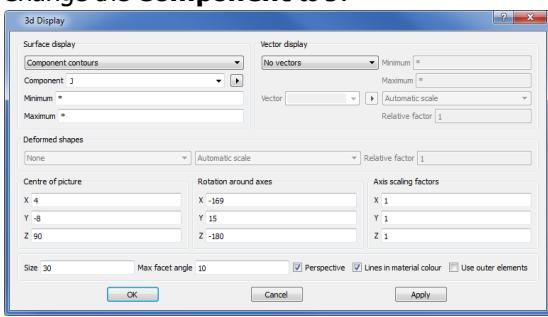
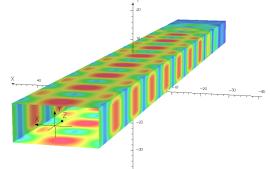


Note that the viewing angles and zoom have been changed from the initial view values.

Display the electric fields by selecting the  **3d Display** toolbutton.

 <b>3d Display</b>	<p>Select <b>Component contours</b> from the <b>Surface display</b> drop-down list and set the <b>Component</b> to, for example, <b>E</b>.</p>  <p>Click on <b>Apply</b>.</p>	
	<p>Change the <b>Component</b> to <b>H</b>.</p>  <p>Click on <b>OK</b>.</p>	

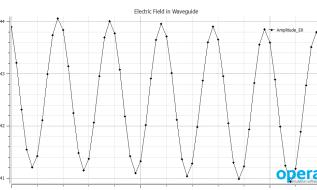
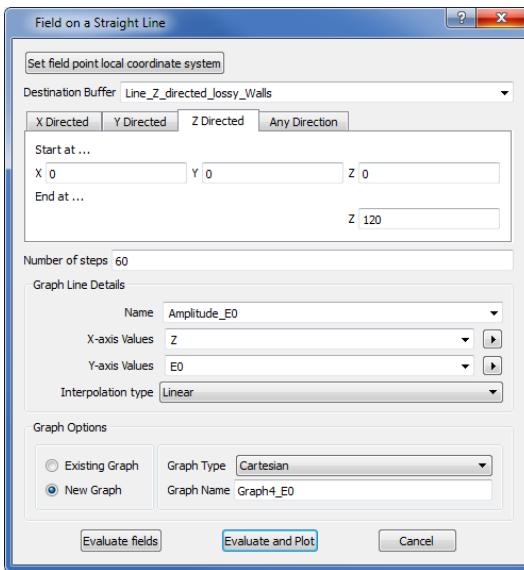
The fields may also be displayed on the inside surfaces of the waveguide walls. Use the  **Select** toolbutton again to include only these surfaces.

 <b>Select</b>	<p>Press <b>Clear selection</b> first, then select <b>INSIDEWALLS</b> label from <b>Surfaces -&gt; User Labels</b>. Press <b>Add to selection</b>.</p>  <p>Click on <b>Select and Refresh</b>. The display now shows <b>H</b>, since this was the last <b>Component</b> chosen. The walls are conducting, so the surface current density may also be displayed.</p>	
 <b>3d Display</b>	<p>Change the <b>Component</b> to <b>J</b>.</p>  <p>Click on <b>OK</b>.</p>	

## Fields along a line

As in the previous model, examining the fields along lines can be useful. The electric field along the centre of the guide will be considered again, concentrating on the amplitude E0.

Select the  **Fields on a Line** toolbutton and enter the data below to generate field values along a z-directed line in the centre of the empty waveguide section.

 <b>Fields on a Line</b>	<p>In <b>Buffer</b>, give a name <b>Line_Z_directed_short_lossy_Walls</b>. Click on the <b>Z Directed</b> tab. Start at <b>(0, 0, 0)</b> and end at <b>Z = 120</b>. Set the <b>Number of steps</b> = 60.</p> <p>In the <b>Graph Line Details</b> section, set the <b>Name</b> to <b>Amplitude_E0</b>. For the <b>X-axis Values</b>, type <b>z</b>, and for the <b>Y-axis Values</b>, type <b>E0</b>.</p> <p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <b>Graph4_E0</b>.</p>	
		<p>Click on <b>Evaluate and Plot</b>.</p>

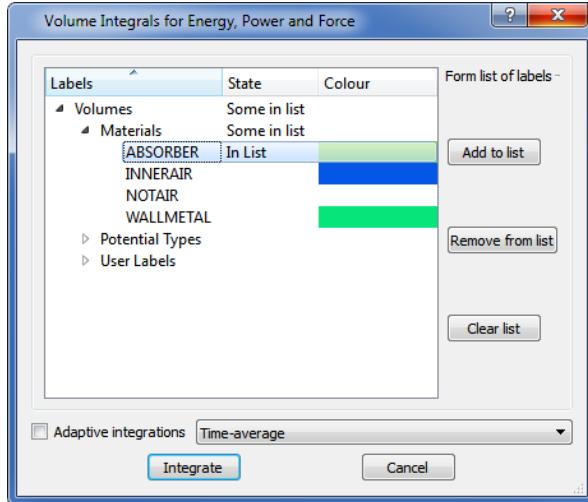
As can be seen there is very little cyclic variation in the amplitude of the wave along the guide, indicating, as before, that the reflection from the region containing the absorber is low. However, unlike previously, there is a gradual decrease in the maximum amplitude owing to the loss in the finite conductivity walls.

## Power dissipation

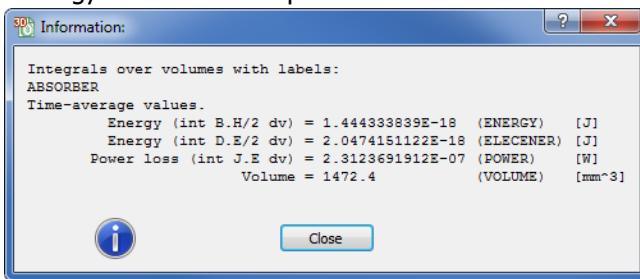
Both the walls and absorber contribute to the power dissipation in this model. Their contributions can be evaluated using the **ENERGY** command, which is available by selecting the  **Energy, Power and Force** toolbar.

  
**Energy,  
Power and  
Force**

To calculate the power dissipation only in the absorber, press on **Clear list** then select the label **ABSORBER** in the **Volumes > Materials** list, and press **Add to list**. Ensure that the **Time-Average** is chosen from the drop-down list, then select **Integrate**.

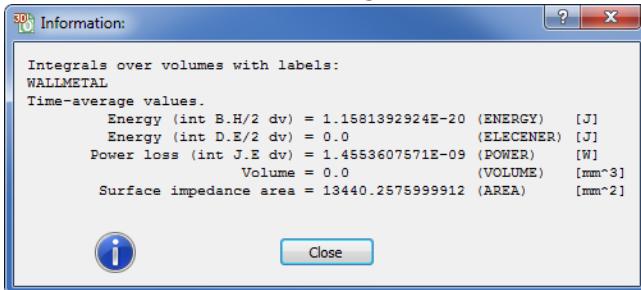


Once the calculation has been performed, an information box is displayed showing the energy in the fields and power loss in the absorber.

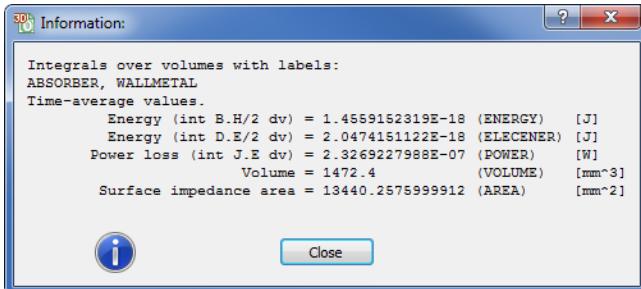


The **ENERGY** command may then be repeated for the walls alone, and for the absorber and walls combined.

For the walls alone, the ENERGY command gives:



and for the combination of absorber and walls:



As can be seen, the calculated loss in the absorber is very close to the value seen in the first model. This is as expected, since the loss in the walls is some two orders of magnitude lower than the absorber loss.

This completes the post-processing for the Steady-State HF example. The Post-Processor may now be closed.

# **Chapter 12**

## **Stress Examples and Multiphysics**

### **Introduction**

---

In this chapter, the process of solving stresses and deformations in two examples using the Opera-3d **Static Stress** module is presented.

The first model is a simple cantilever beam deformed by a distributed load applied to its upper surface. This introduces the basic concepts of the stress analysis simulation:

- specifying mechanical material properties;
- applying boundary conditions for mechanical loads;
- applying boundary conditions for physical constraints;
- including forces due to gravity and rotation;
- viewing results in the Post-Processor.

The second examples couples **Electromagnetic** and **Static Stress** simulations in a multiphysics solution. In addition to the above, this example shows how Opera is capable of automatically calculating a body force density distribution from the electromagnetic simulation and using it within the stress analysis.

## Cantilever Beam

### Building the Geometry

The cantilever beam model consists of a single cell created using the  **Block** primitive shape in the Modeller.

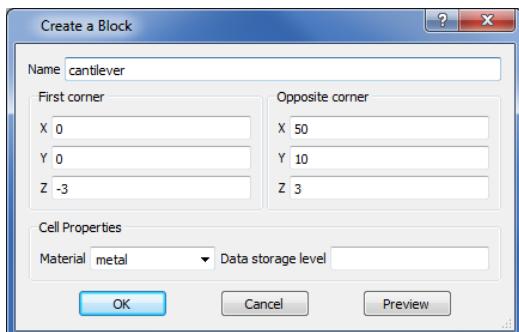
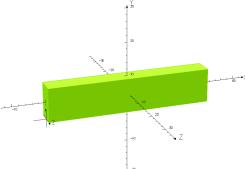
#### Analysis type

Firstly, open the Modeller and in the Analysis Settings window change the analysis type to **Static Stress**.

The stress analysis module allows the effects of gravity to be included in a model. Gravity can act in any of the coordinate system directions and is applied to all elements. In this example, gravity will be ignored.

The effects of changes in temperature on mechanical behaviour can also be included as pre-stressing. The user defines the Reference Temperature (the temperature at which the body is unstressed) and the operating temperature, which can be functional or imported through a table. Material properties include the thermal expansion coefficient. In this example, thermal expansion is ignored.

#### The cantilever

 <b>Create Block</b>	<p>Create the <math>50 \times 10 \times 6</math> mm cantilever with coordinates <b>(0,0,-3)</b> and <b>(50,10,3)</b>. Use the <b>Material</b> name <b>metal</b>.</p> <p></p> <p>Click on <b>OK</b>.</p>	
--	--	---

### Boundary Conditions

Two types of mutually exclusive boundary conditions can be applied to surfaces of models.

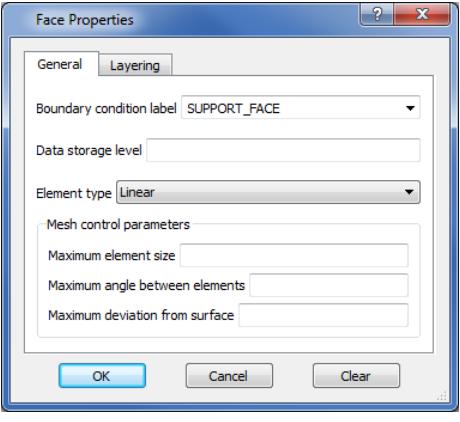
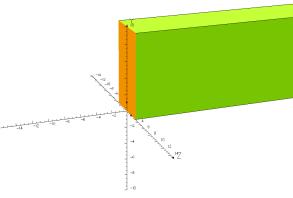
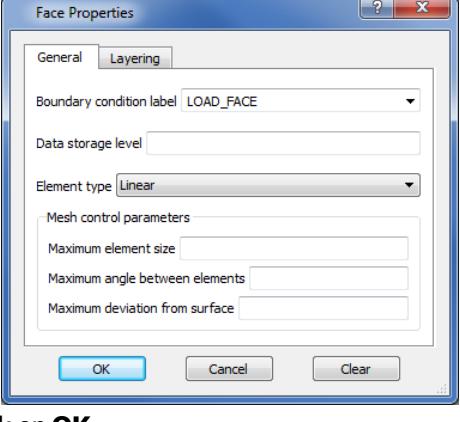
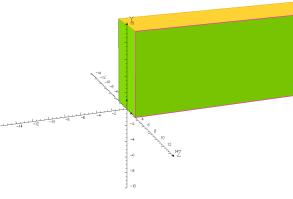
- Constraint conditions control the freedom that a node on the boundary surface has to move. This can vary from being completely fixed to being free to move in any direction.
- Load conditions apply a traction over the surface. Nodes on the surface are free to move in any direction, unless they also lie on a constraint condition that inhibits this.

In this example, both types of boundary condition are used. The left face of the cantilever (at  $X = 0$ ) is assumed to be fixed. A pressure is applied to the upper face (at  $Y = 10$ ) of the cantilever.

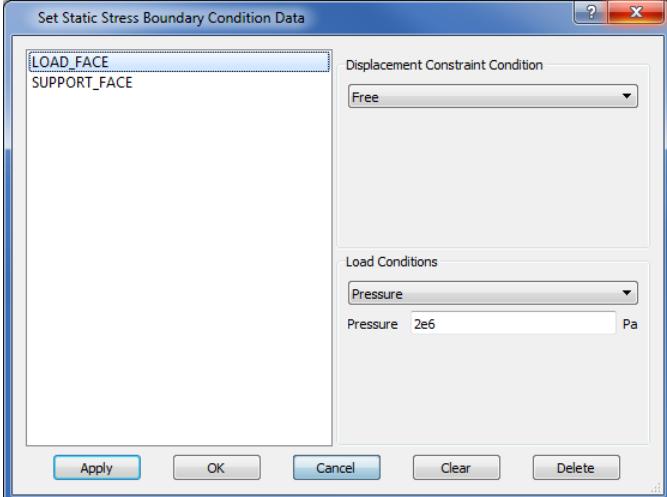
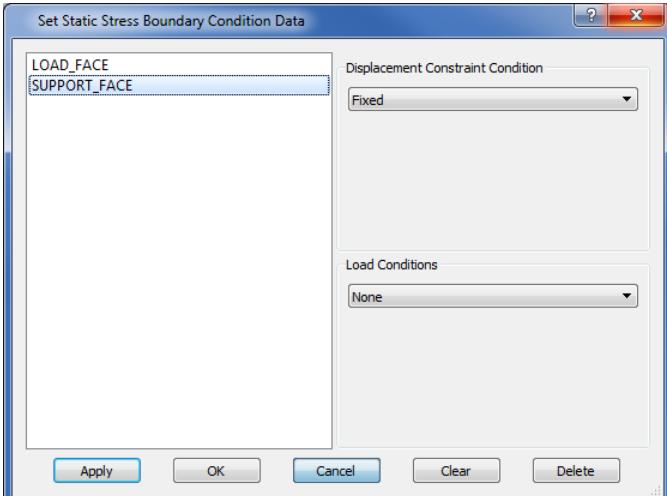
This is performed in two stages. Firstly, **Boundary condition labels** are applied to the two faces. Then the appropriate boundary condition is applied to each label.

This allows boundary conditions to be easily changed without having to reselect faces in the geometry.

## Applying face labels

 <b>Pick Faces</b>	<p>Pick the face at <math>X=0</math> with the right mouse button. The face will be high-lighted. In the context sensitive menu, select <b>Face properties</b> and use <b>support_face</b> as the <b>Boundary condition label</b>.</p>  <p>Click on <b>OK</b>.</p>	
 <b>Pick Faces</b>	<p>Now pick the face at <math>Y=10</math> with the right mouse button. In the <b>Face properties</b> dialog, give <b>load_face</b> as the <b>Boundary condition label</b>.</p>  <p>Click on <b>OK</b>.</p>	

## Setting the boundary conditions

 <b>Boundary Conditions</b>	<p>Select the <b>load_face</b> label and set the <b>Displacement Constraint Condition</b> to <b>Free</b> and the <b>Load Conditions</b> to <b>Pressure</b> with a value of <b>2E6</b> Pascals.</p>  <p>Click on <b>Apply</b>.</p>
	<p>Select the <b>support_face</b> label and set the <b>Displacement Constraint Condition</b> to <b>Fixed</b>.</p>  <p>Click on <b>OK</b>.</p>

## Material Properties

The stress analysis simulation module allows isotropic, orthotropic (referred to as anisotropic in Opera-3d) and fully anisotropic materials. In this model, the cantilever will be assumed to be isotropic which requires only its Young's modulus and Poisson's ratio to be defined.

Other material properties that can be defined are

- the thermal expansion coefficient
- the mass density

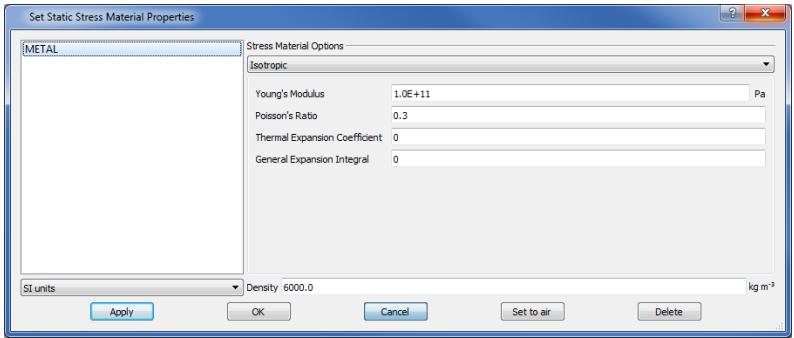
Neither of these are needed in this example as gravity and thermal expansion are being ignored and the cantilever is not rotating. However, the density is also set for completeness of the material information

Note that orthotropic materials also require values for the shear modulus.



**Material Properties**

Select the **METAL** label. Set the material as **isotropic**. Give a value of **1E11** Pascals to **Young's modulus** and give a value of **0.3** to **Poisson's ratio**. A value of **6000 kg m<sup>-3</sup>** has been given to the **Density**.

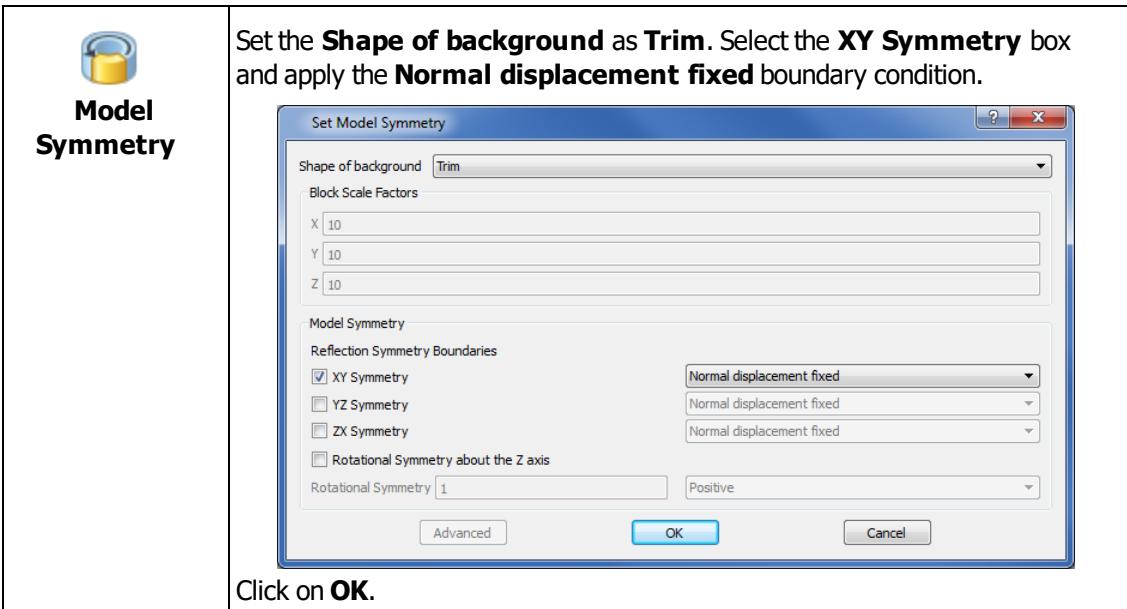


Click on **OK**.

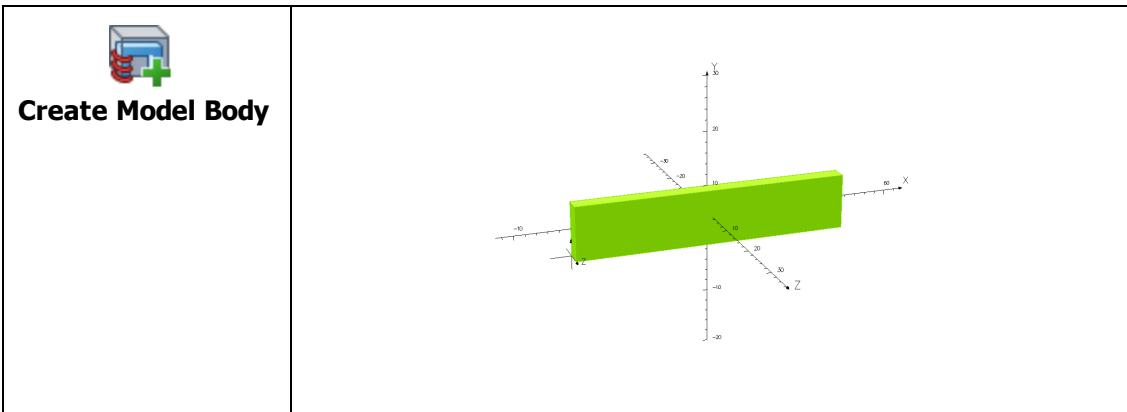
## Exploiting Symmetry

The final data to be defined before creating the finite element mesh and running the simulation is the symmetry of the model. Exploiting any symmetry in the model is always worthwhile for speed of analysis.

In this example, the model is symmetric on the plane Z = 0. Nodes of the model that lie on this plane will not be displaced in the Z direction because the constraint and load boundary conditions are in mirror symmetry about the plane.



The effect of this can be seen when the model body is created.



## Creating the Mesh and Running the Simulation

Building the finite element mesh and submitting the stress analysis simulation can be achieved through a single dialog. However, in this example, the mesh generation and submission will be handled through three individual dialogs to clarify the stages. In the second example, the "all-in-one" process will be used.

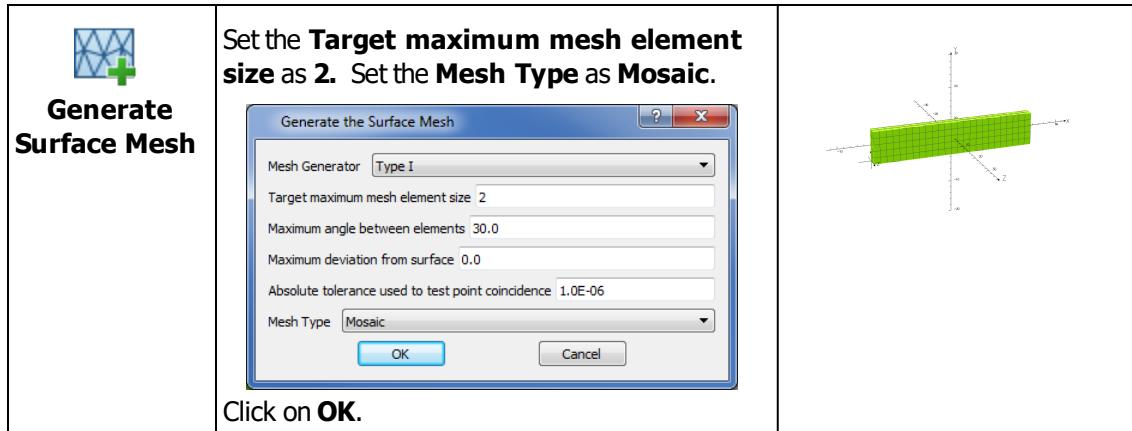
The dialogs specify parameters for the construction of both the surface and volume meshes, the name of the Opera-3d database (**op3**) file and whether the simulation is to be run immediately.

In this model, the geometry of the cantilever allows the use of quadratic hexahedral elements which will always give the most accurate results for the stress analysis module for elements of a particular

size. However, they can only be used where the geometry satisfies certain topological constraints. Further information can be found in [Meshing in the Modeller \[page 541\]](#).

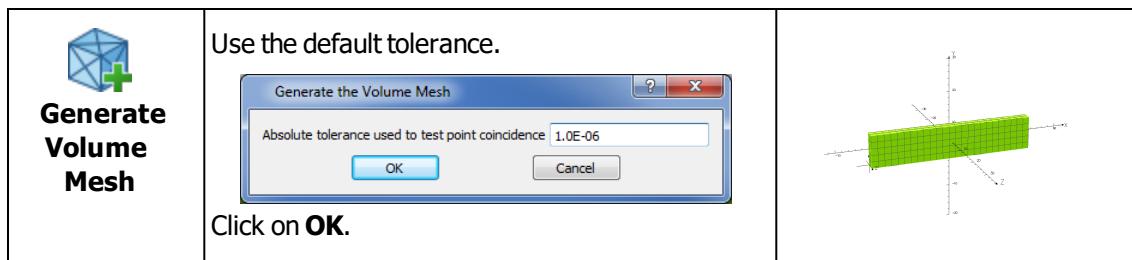
## Surface mesh

Meshing is performed in 2 stages. Firstly the surface mesh is created on all the faces of the cell.



## Volume Mesh

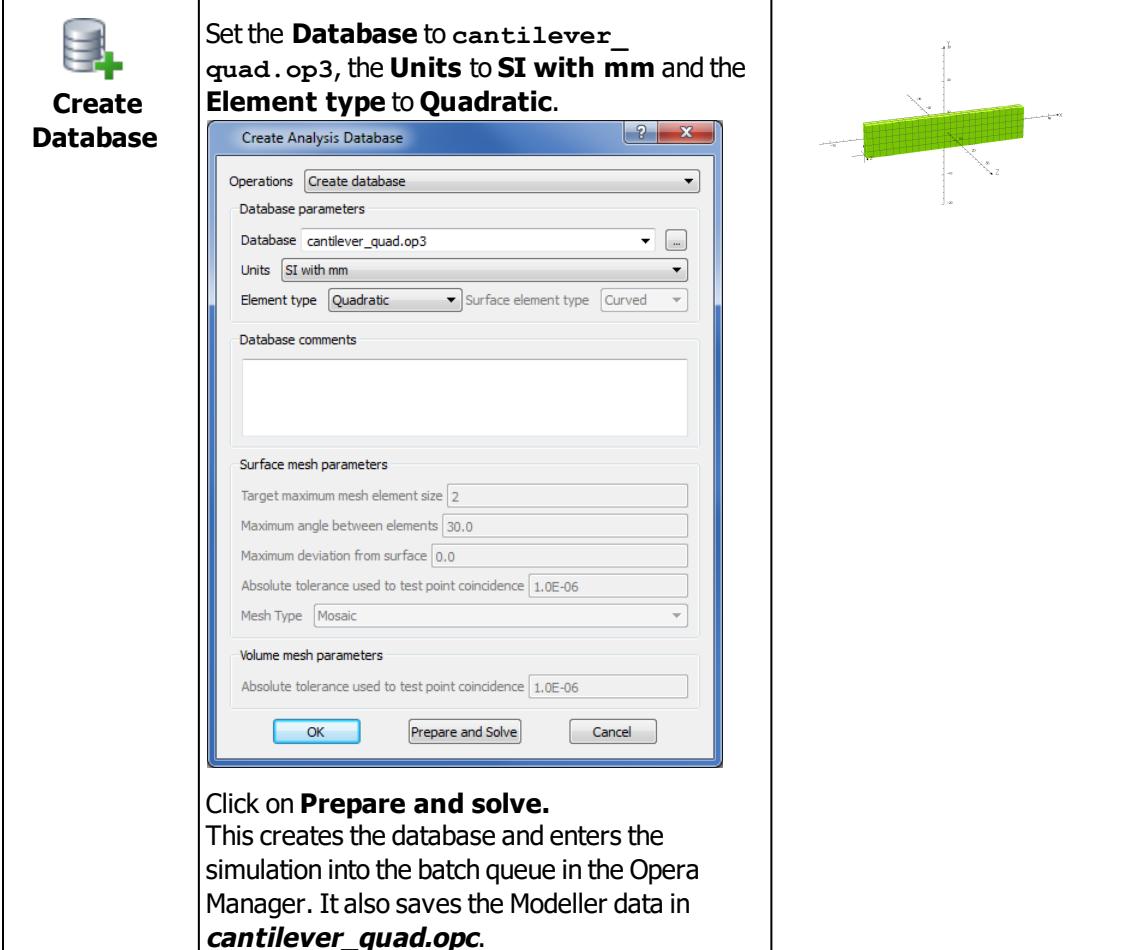
The second stage of mesh generation is to form the mesh of volume elements in the cell, starting from the surface mesh.



## Starting the simulation

The data for the simulation is stored in a binary database file (**op3**) which contains the finite element mesh, material properties, boundary conditions and any additional data (such as gravity) needed for the stress analysis.

As a general note, stress analysis using tetrahedral elements should always be set up with quadratic elements in order to get accurate results.



The screenshot shows the 'Create Database' dialog box from the Opera Modeller. The 'Operations' dropdown is set to 'Create database'. In the 'Database parameters' section, the 'Database' is set to 'cantilever\_quad.op3', 'Units' to 'SI with mm', and 'Element type' to 'Quadratic'. Below this, there are sections for 'Database comments', 'Surface mesh parameters' (with target element size of 2), and 'Volume mesh parameters' (with absolute tolerance of 1.0E-06). At the bottom are 'OK', 'Prepare and Solve', and 'Cancel' buttons. To the right of the dialog box is a 3D rendering of a green rectangular cantilever beam with a coordinate system (X, Y, Z) and a grid overlay.

Set the **Database** to **cantilever\_quad.op3**, the **Units** to **SI with mm** and the **Element type** to **Quadratic**.

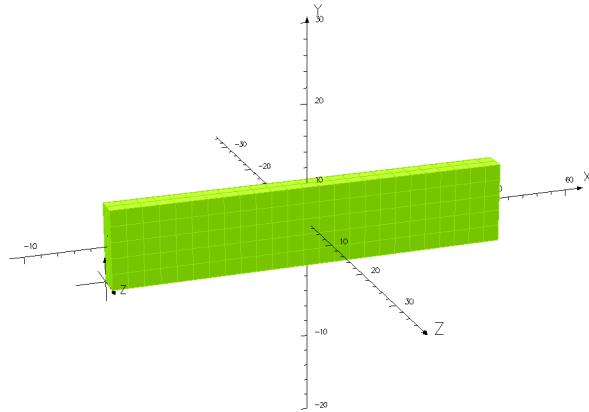
Click on **Prepare and solve**. This creates the database and enters the simulation into the batch queue in the Opera Manager. It also saves the Modeller data in **cantilever\_quad.opc**.

This completes the Modeller session.

## Displaying Results

Progress on the simulation can be monitored in the Opera Manager and simulation window. However, this is a small simulation and finishes in a few seconds.

When the simulation is complete, start the Post-Processor by clicking on the Post-process button on the simulation window. Alternatively launch the **Post-Processor** from the Modeller or double-click on *cantilever\_quad.op3* in the Opera Manager. Any of the methods will open the Post-Processor and load and display the geometry. **Figure 12.1** shows the geometry displayed in the Post-Processor.

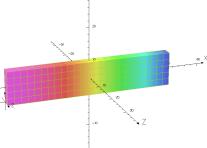


*Figure 12.1 Geometry of cantilever*

## Displacements

The Post-Processor facilities to display results from electromagnetic and thermal results may also be used to display results from the stress analysis simulation.

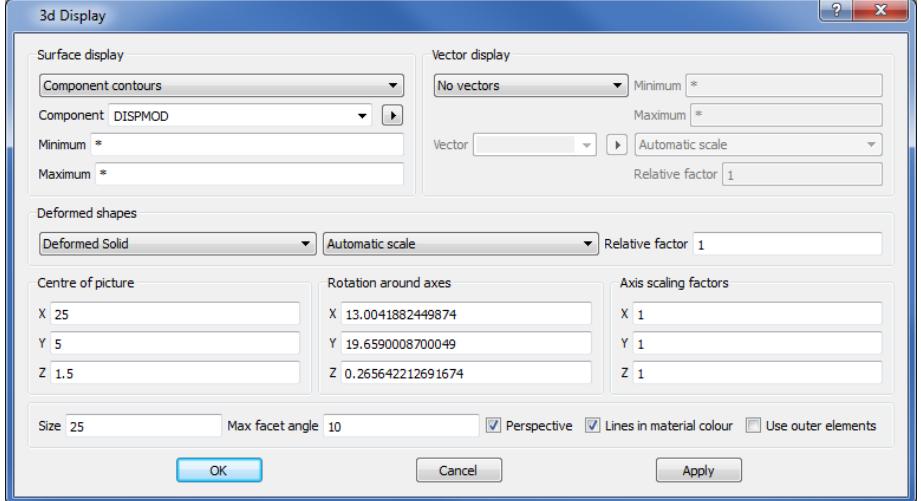
The stress analysis module computes the displacement of each node in X, Y and Z as the solution to the finite element equations. In the cantilever, the pressure was applied to the upper surface, so it is expected that a significant displacement in the Y-direction will be seen.

 <b>3d Display</b>	<p>Change from <b>Material colours</b> to <b>Component contours</b> as the <b>Surface display</b> option and set the <b>Component</b> to <b>DISPy</b>, the displacement in the Y-direction. Note that the "pull-right" button next to the <b>Component</b> input allows this to be selected.</p> <p><b>3d Display</b></p> <p>Surface display: Component contours Component: DISPy Minimum: * Maximum: *</p> <p>Deformed shapes: None Centre of picture: X: 25, Y: 5, Z: 1.5 Rotation around axes: X: 20, Y: 20, Z: 0 Axis scaling factors: X: 1, Y: 1, Z: 1</p> <p>Size: 25, Max facet angle: 10, Perspective: <input checked="" type="checkbox"/> Lines in material colour: <input type="checkbox"/> Use outer elements</p> <p>OK Cancel Apply</p> <p>Click on <b>OK</b>.</p>	
--	--	---

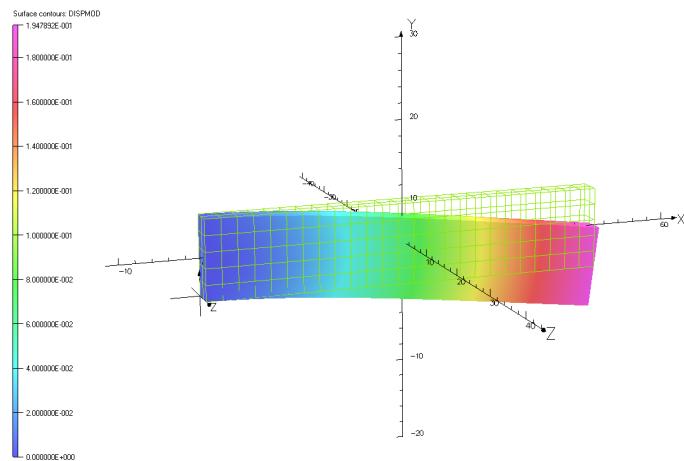
The physical deflection of the beam can be displayed visually.

**3d Display**

From the pull down menu, select **Deformed solid** and use **Automatic scale**. The software will determine an appropriate scaling factor to make the deflection clearly visible. Change the **Component** to **DISPmod**.



Click on **OK** to produce the result shown in [Figure 12.2](#).



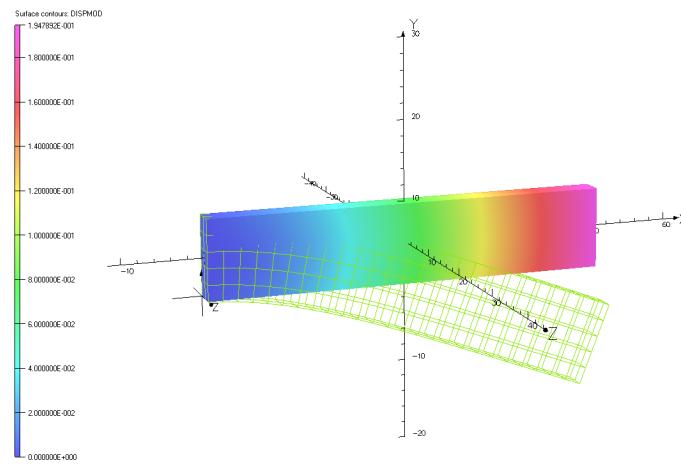
*Figure 12.2 Deflection of beam with automatic scaling*

There are also two other options to display the deformation. The mesh can be deformed and the solid geometry displayed as originally defined, or both mesh and solid can be deformed.

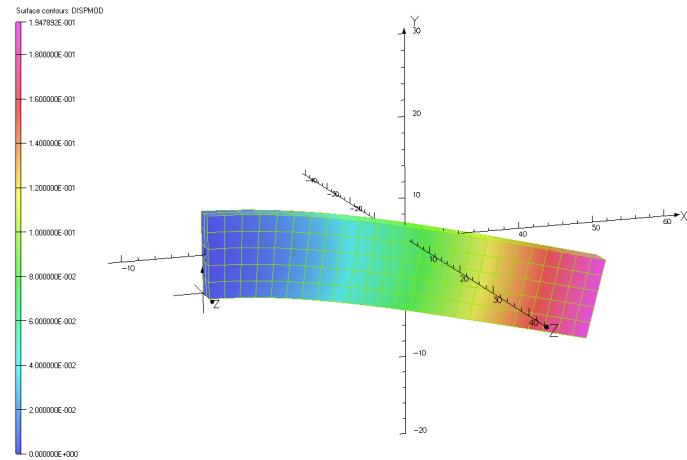
There are also two options to manually adjust the scaling factor chosen by the software. A relative scaling of the automatic scaling factor can be chosen, or manual scaling can be selected. The latter option requires the user to choose the absolute scaling factor based on the display size and the deformation that has occurred.

**Figure 12.3** shows the option where the mesh is deformed and original geometry displayed using a relative scaling of 3 from the automatic scaling factor.

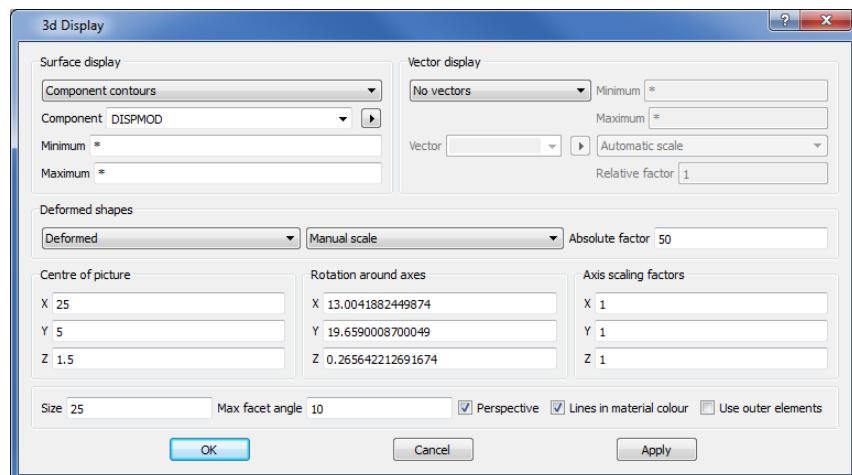
**Figure 12.4** shows the option where both mesh and geometry are deformed and a manual scaling factor is used. The size of the display is about 30 mm and the maximum deformation is just under 0.2 mm. To make the deformation clearly visible, a manual scaling factor of 50 has been chosen - making the maximum displayed deformation just under 10 mm. **Figure 12.5** shows the dialog for these options.



*Figure 12.3 Deflection of beam with relative scaling*



*Figure 12.4 Deflection of beam with manual scaling*



*Figure 12.5 Dialog for manual scaling and both mesh and geometry deformed*

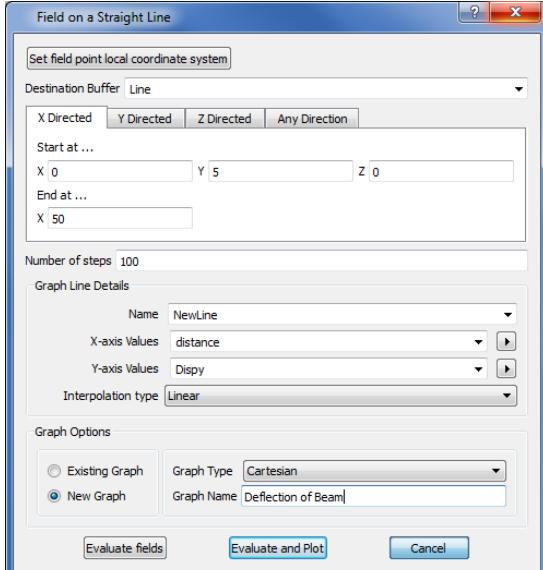
The results can be examined more critically by evaluating the displacement in the Y direction along any line parallel to the X-axis.



**Fields on a Line**

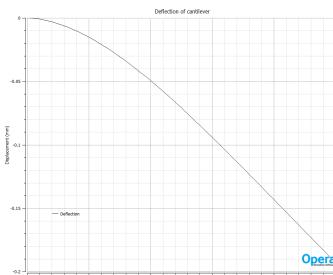
On the **X-directed** tab, set the starting point as (0,5,0) and end at X = 50. Set the **Number of steps** as **100**. Set the **X-Axis/Azimuthal Values** to **Distance** and the **Y-Axis/Radial Values** to **DISP<sub>y</sub>**. Select the **New Graph** radio button and enter **Deflection of Beam** for the **Graph Name** and **Cartesian** as the **Graph Type**.

Click on **Evaluate and plot**.



The dialog box shows the following settings:

- Set field point local coordinate system:** Line
- X Directed** tab selected.
- Start at ...**: X: 0, Y: 5, Z: 0
- End at ...**: X: 50
- Number of steps**: 100
- Graph Line Details**:
  - Name: NewLine
  - X-axis Values: distance
  - Y-axis Values: Disp<sub>y</sub>
  - Interpolation type: Linear
- Graph Options**:
  - New Graph
  - Graph Type: Cartesian
  - Graph Name: Deflection of Beam



The graph displays the deflection curve for a cantilever beam under a uniform load. The x-axis is labeled "distance in x (mm)" and ranges from 0 to 50. The y-axis is labeled "deflection in z (mm)" and ranges from -0.2 to 0.6. The curve starts at approximately (0, 0.5) and follows a parabolic shape, ending at approximately (50, -0.1).

This shows the familiar shape for a cantilever beam with a uniform load.

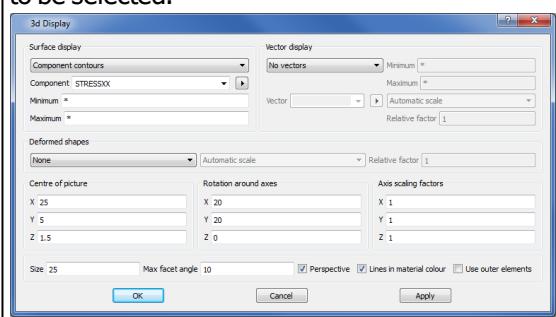
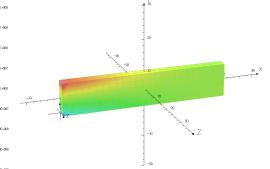
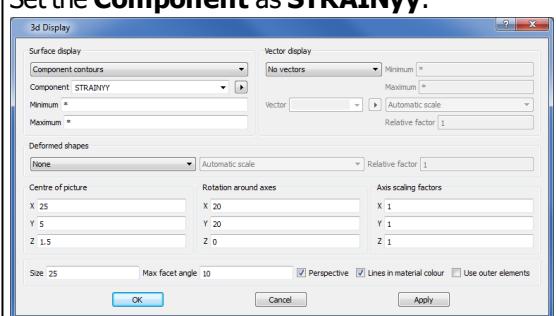
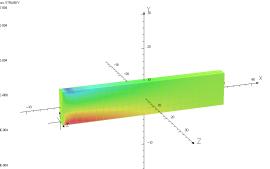
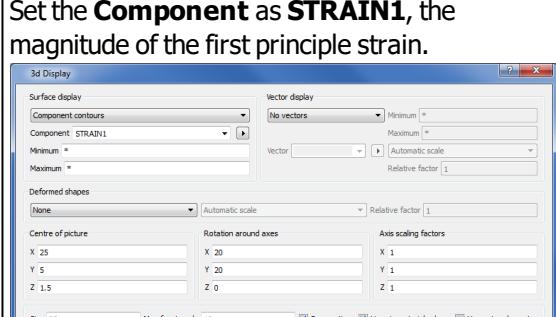
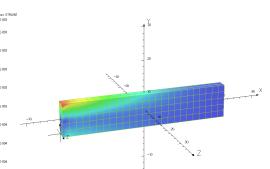
Note that, in the graph shown, the line style has been modified and it has been annotated using the facilities in the **Graphs** tab.

## Stress and strain

The Post-Processor can also display results that are derived from the displacements. Both the stress and strain (including any initial strain produced by thermal expansion) can be determined.

These are full tensors with 9 components, referenced by system variables such as **STRESS<sub>xx</sub>** - the normal stress to the X-plane and **STRESS<sub>xy</sub>** - the shear stress in the Y-direction on the X-plane.

The Post-Processor can also evaluate principal stresses and strains, and stress and strain invariants. More details of the names of the system variables can be found in the **Opera-3d Reference Manual**.

 <b>3d Display</b>	<p>Set the <b>Component</b> as <b>STRESSxx</b>. Note that the "pull-right" button on the field also allows this to be selected.</p>  <p>Click on <b>OK</b>.</p>	
 <b>3d Display</b>	<p>Set the <b>Component</b> as <b>STRAINyy</b>.</p>  <p>Click on <b>OK</b>.</p>	
 <b>3d Display</b>	<p>Set the <b>Component</b> as <b>STRAIN1</b>, the magnitude of the first principle strain.</p>  <p>Click on <b>OK</b></p>	

This completes the first stress analysis example.

## Steel Plate Deformed by Energized Solenoid

---

In the second example, a "pot-core" style DC solenoid produces magnetic fields that attract a steel plate above the core. The magnetically induced body force density distribution in the plate is computed and used to determine the deflection and stress in the plate. The DC solenoid model is solved using the **Magnetostatic** simulation module within a multi-case, multiphysics simulation.

Opera-3d allows the same model and mesh to be used for both the magnetostatic and stress simulations. The stress analysis simulation ignores any material in the model labelled **AIR** and any materials that do not have stress analysis properties (Young's modulus etc.).

In this example, we shall see that the body force density can also be applied automatically if a solution from a previous electromagnetic analysis is available.

It is also possible to use a different model and mesh in the stress analysis simulation, as long as the body force density distribution can be determined in the relevant parts of the stress analysis mesh. In this case the multiphysics options can not be used and the body force density table must be added to the stress simulation using the **TABLE** tools in the Post-processor. For example, a coil may be supported by a non-magnetic support structure. In a magnetostatic simulation, the support structure is unnecessary but is an important part of the stress analysis simulation. The introduction of the support structure may modify the mesh in the coil for the stress analysis from that in the magnetostatic mesh. However, because the geometry of the coil is identical in both models, it is possible to obtain the distribution of the body force density in the coil at the centroids of the elements used in the stress analysis mesh.

### Building the Geometry

The geometry of the solenoid and plate is constructed with a few primitive objects in the Modeller. Data storage levels are used to cut the slot in the yoke in which the solenoid coil sits. It is assumed that the user is already familiar with using Opera-3d to perform magnetic field calculations, so the instructions to build the model are minimal.

Firstly, the analysis sequence will be specified. The default options for the **Stress** analysis will be used. However, options will be set to run the **Magnetostatic** model with non-linear materials and at 3 different excitation levels.

In the **Analysis Settings** window add a **Magnetostatic** analysis followed by a **Static Stress** one. To be able to run the multiphysics simulations at several different values of current density in the coil, select the **Magnetostatic** analysis and set the following **Drive Scaling** factors: 1, 2 and 5. The magnetic simulations will be run with non-linear materials. Select **Use nonlinear properties**.

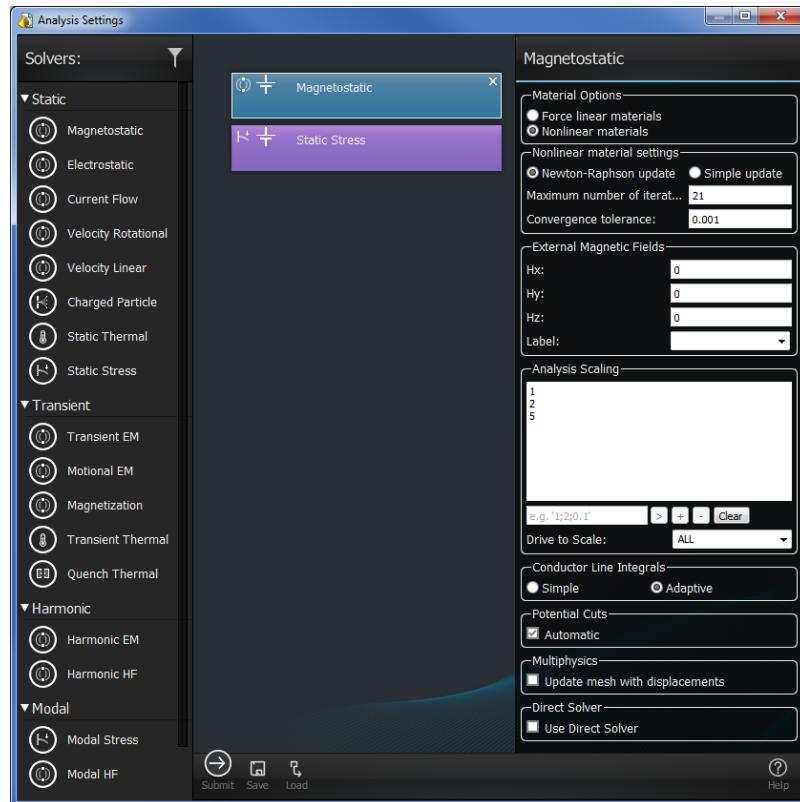
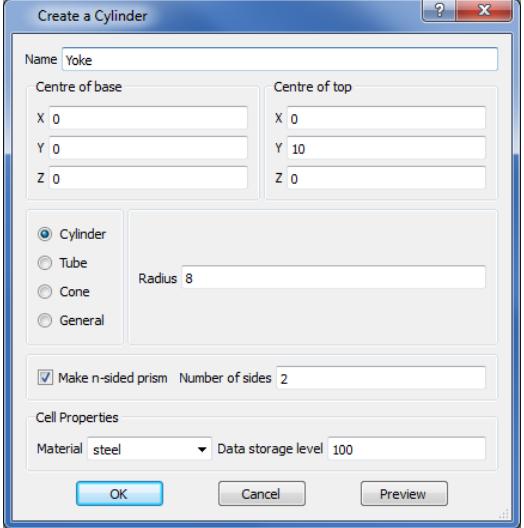
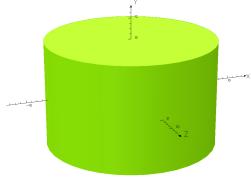
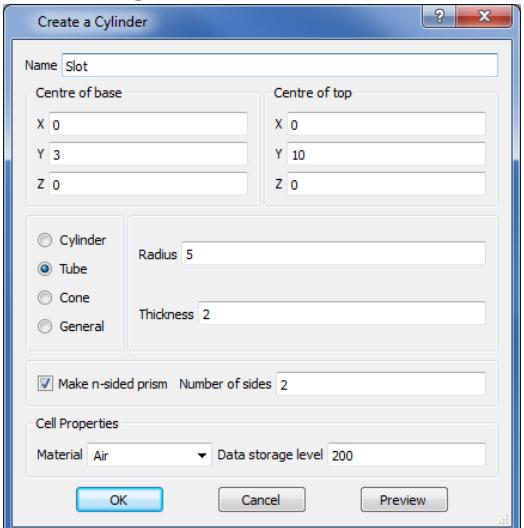


Figure 12.6 Setup the multiphysics analysis

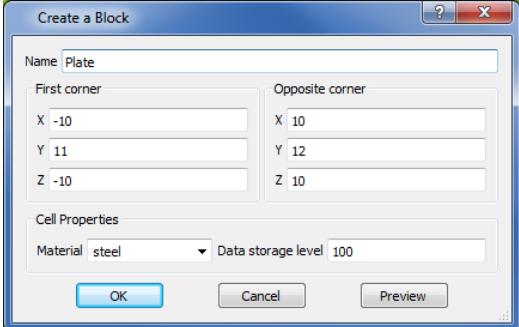
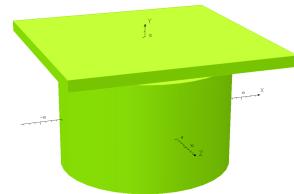
The changes made will run a total of 6 simulations when the analysis is submitted. Each magnetostatic simulation will be followed by a stress analysis using the forces calculated in the electromagnetic model. The effect of the non-linear magnetic material will be seen when the results are examined in the Post-processor.

## Yoke

The yoke is a 10 cm long steel cylinder, with radius 8 cm, with a 2 cm wide x 7 cm deep groove cut into it.

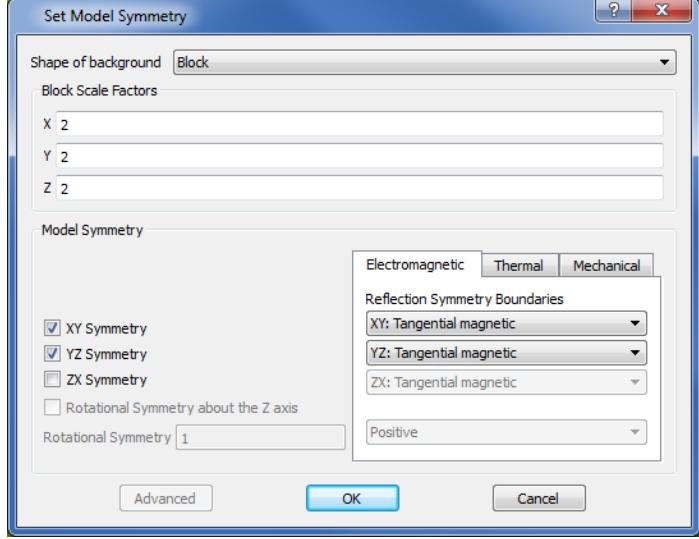
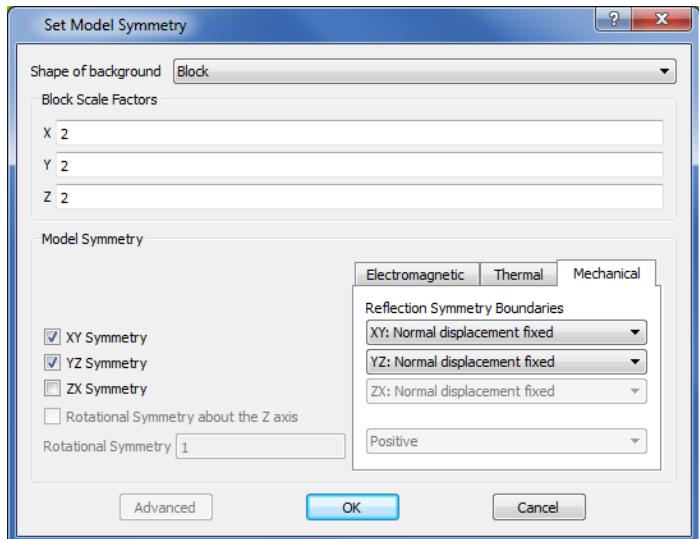
 <b>Create Cylinder</b>	<p>Create a <b>Cylinder</b> named <b>Yoke</b> with coordinates <b>(0,0,0)</b> and <b>(0,10,0)</b> and radius <b>8</b>. Use the <b>Material</b> name <b>steel</b> and assign a <b>Data storage level of 100</b>.</p>  <p>Click on <b>OK</b>.</p>	
 <b>Create Cylinder</b>	<p>Create a <b>Tube</b> named <b>Slot</b> with coordinates <b>(0,3,0)</b> and <b>(0,10,0)</b>, radius <b>5</b> and thickness <b>2</b>. Use the <b>Material</b> name <b>air</b> and assign a <b>Data storage level of 200</b>.</p>  <p>Click on <b>OK</b>.</p>	

## Plate

 <b>Create Block</b>	<p>Create the 20 x 1 x 20 cm <b>Plate</b> with coordinates <b>(-10,11,-10)</b> and <b>(10,12,10)</b>. Use the <b>Material name steel</b> and <b>Data storage level of 100</b>.</p> <p></p> <p>Click on <b>OK</b>.</p>	
--	--	---

## Background

One quarter of the model will be solved. The symmetry can be seen after creating the model body.

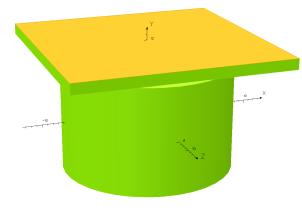
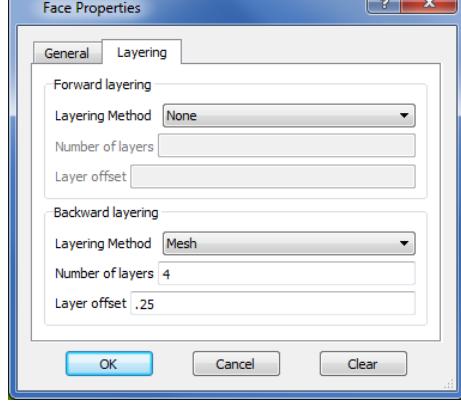
 <b>Model Symmetry</b>	<p>A <b>Block</b> background region twice as large as the defined geometry in each of the coordinate directions will be applied. This will only be used in the magnetostatic analysis as the stress simulation ignores material <b>AIR</b>. Set all the <b>Block Scale Factors</b> to <b>2</b>. Symmetry is applied to the <b>XY</b> and <b>YZ</b> planes.</p> <p>In the <b>Electromagnetic</b> tab on both planes, the <b>Tangential magnetic</b> boundary condition is applied.</p>  <p>Select the <b>Mechanical</b> tab</p>
	<p>Similarly, on the <b>Mechanical</b> tab, both planes are set to <b>Normal displacement fixed</b></p>  <p>Click on <b>OK</b>.</p>

## Stress Analysis Considerations in the Magnetostatic Model

### Mesh in the plate

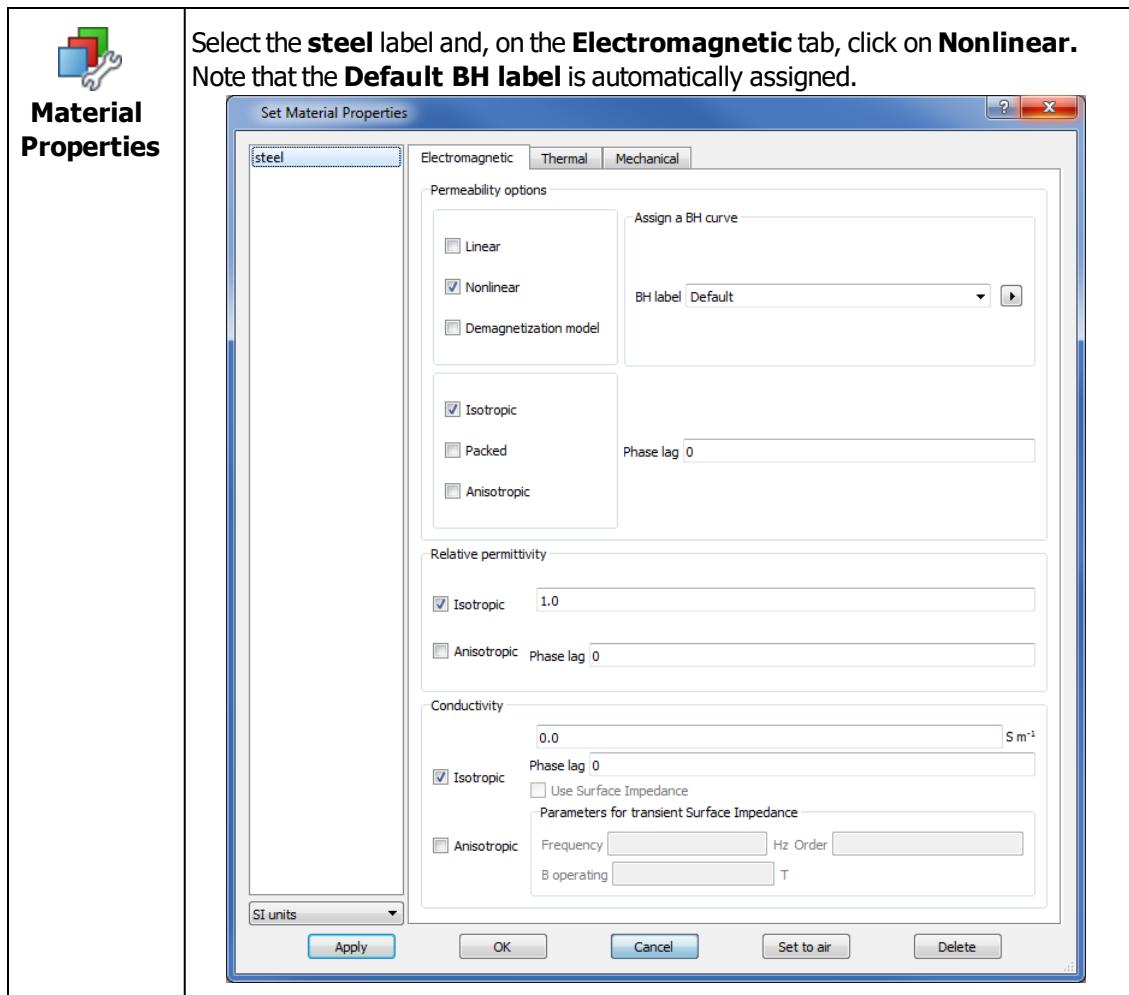
In the stress analysis, there will be quite wide variation of stress across the thickness of the plate, especially near the supports. However, in the other two directions, the variation will be more gradual. Consequently, it is important to define sufficient finite elements through the thickness of the plate to capture the variation. This is achieved by layering the plate.

As previously stated, it is not mandatory that the magnetic and stress analysis have the same mesh, although this must be obeyed when the two solutions are run as part of one multiphysics simulation.

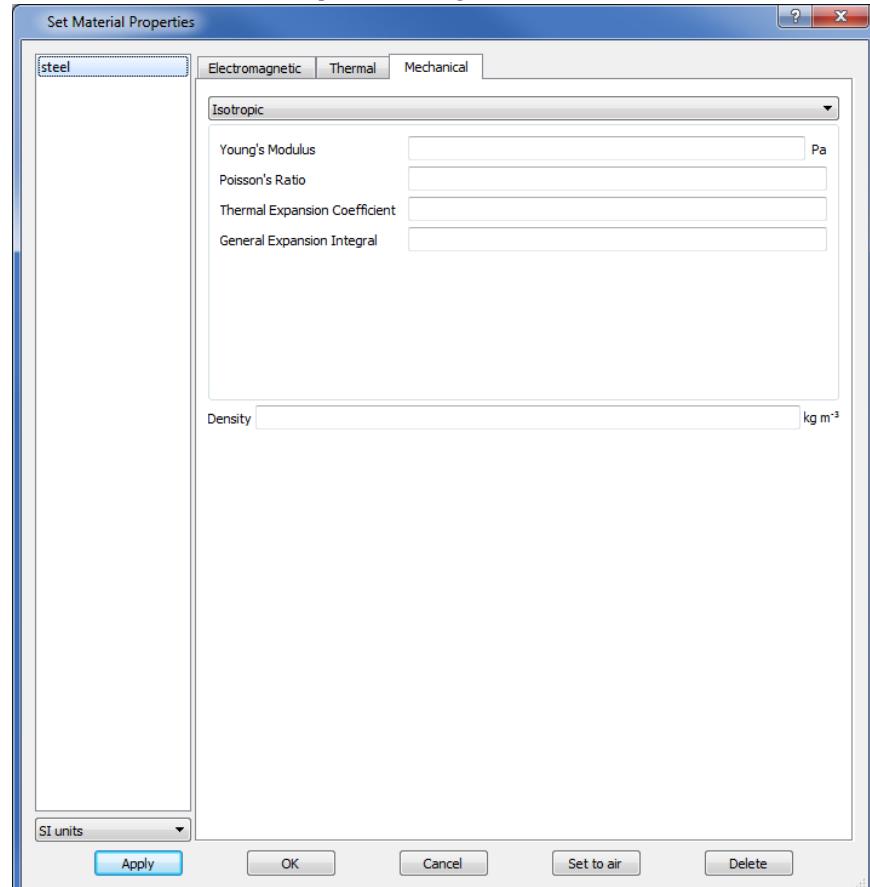
 <b>Pick Faces</b>	<p>Pick the top face of the plate at Y = 12 using the right mouse button.</p>	
	<p>From the context sensitive menu, select <b>Face properties</b>. On the <b>Layering</b> tab, select <b>Backward Layering</b> using the <b>Mesh Layering Method</b>, <b>4</b> layers and <b>0.25</b> offset.</p>  <p>Click on <b>OK</b>.</p>	

## Materials

In the magnetostatic simulations, the steel material will use the Default magnetic characteristic for its non-linear permeability.



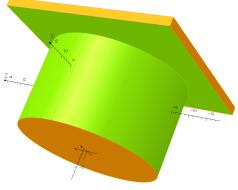
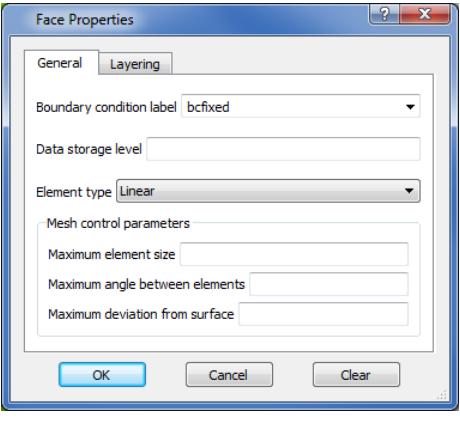
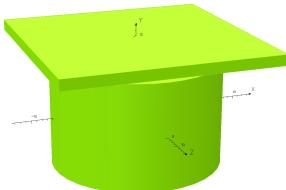
On the **Mechanical** tab, set **Young's Modulus** as **1E11** (Pa), **Poisson's Ratio** as **0.3** and the **Density** as **8000 kg/m<sup>3</sup>**.

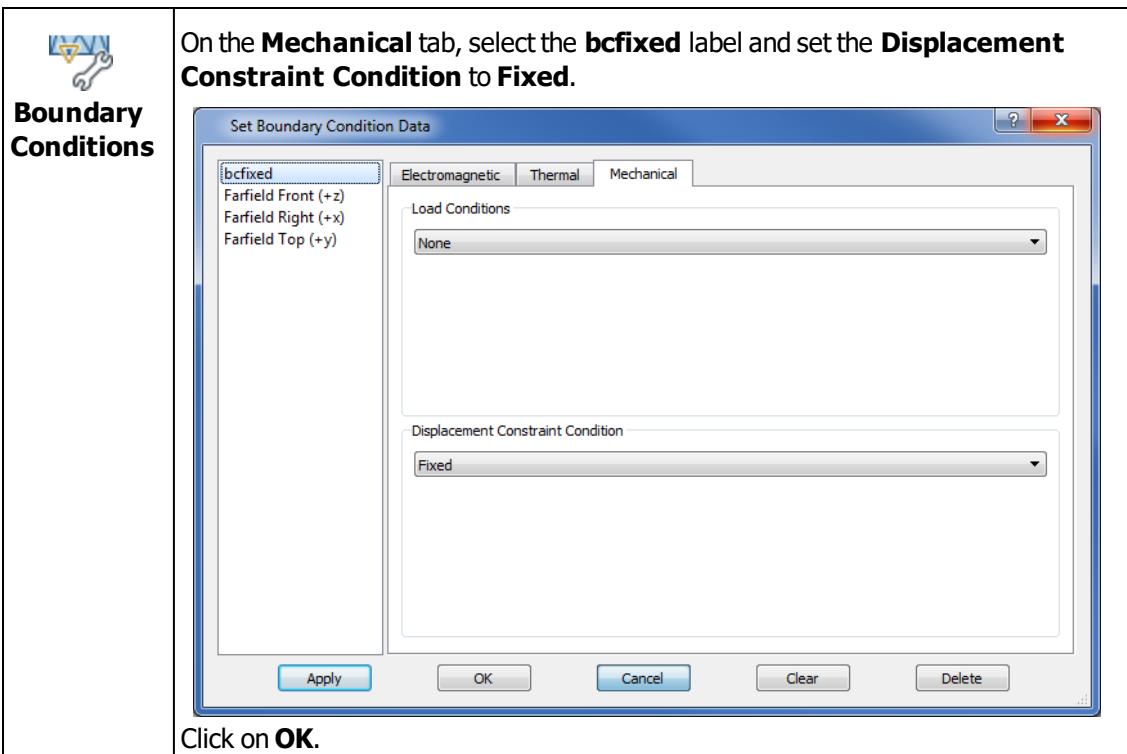


Click on **OK**

## Constraints

In this model, the plate will be fixed on the exterior faces at  $11 < Y < 12$ . It is necessary to add a boundary condition label to these faces and then constrain them. The base of the pot core will also be constrained so that it is unable to move.

 <b>Pick Faces</b>	<p>Pick the faces on the plate at <math>X=10</math>, <math>X=-10</math>, <math>Z=10</math> and <math>Z=-10</math> by double-clicking with the left mouse button. The faces will be high-lighted. With the right mouse button, pick the face on the pot core at <math>Y=0</math>. The fifth face will be highlighted.</p>	 <span style="color: blue;">opera</span>
	<p>In the context sensitive menu, select <b>Face properties</b> and use <b>bcfixed</b> as the <b>Boundary condition label</b>.</p>  <p>Click on <b>OK</b>.</p>	



Note that these boundary condition labels will be unused by the magnetostatic analysis.

The background air can now be added to the model and the symmetry imposed.

### Create Model Body

Select  **Create Model Body**. This gives the geometry shown in [Figure 12.7](#).

The solenoid coil will be added next

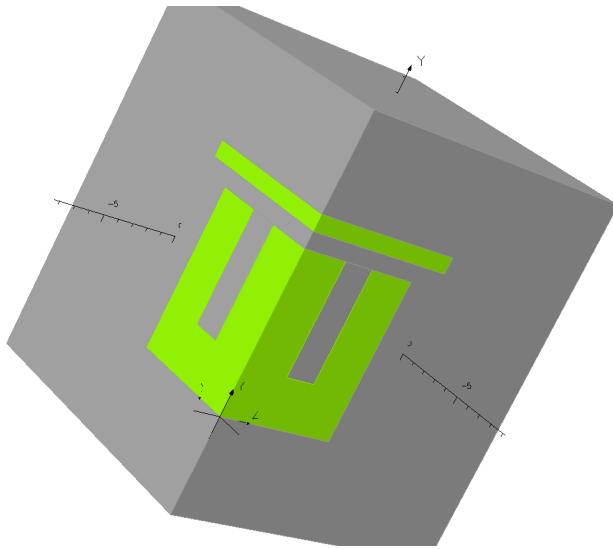
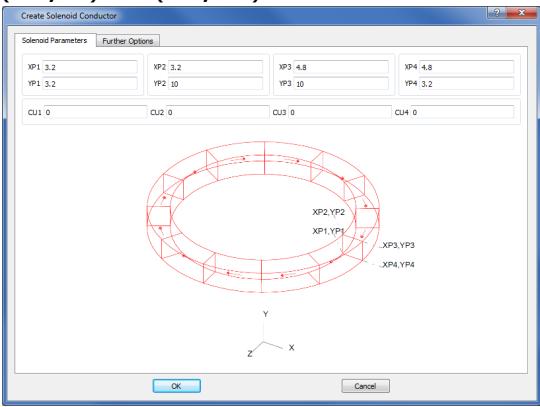
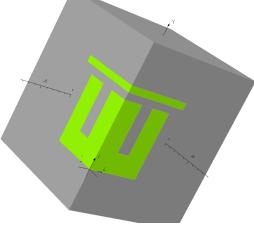
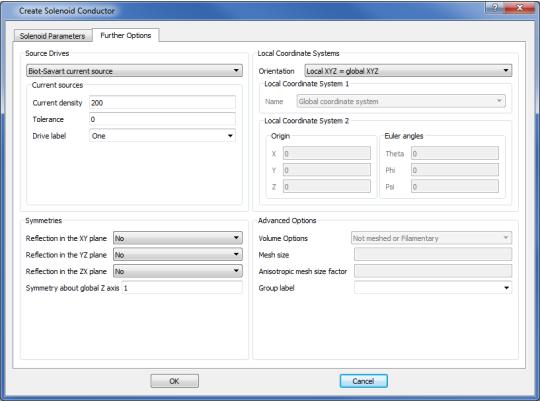
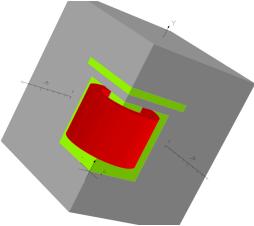


Figure 12.7 Geometry of model

## Adding the Coil

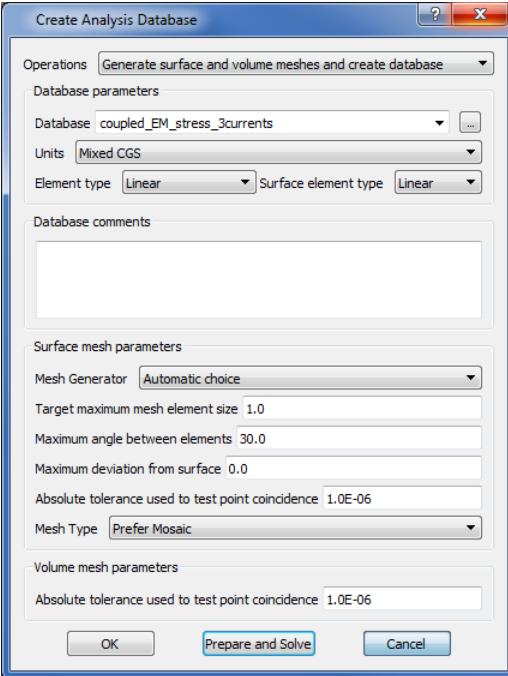
The coil could have been defined during any of the previous stages. However, it is now easier to be certain that it is correctly fitting inside the slot in the yoke.

A 0.2 cm gap surrounds the solenoid, but its top is at the same level as the top of the yoke.

 <b>New Solenoid</b>	<p>On the <b>Solenoid Parameters</b> tab, the 4 corners of the solenoid are <b>(3.2,3.2)</b>, <b>(3.2,10)</b>, <b>(4.8,10)</b> and <b>(4.8,3.2)</b>.</p> 	
	<p>Move to the <b>Further Options</b> tab.</p>	
	<p>Set the <b>Current density</b> as <b>200 A cm<sup>-2</sup></b>.</p> 	
	<p>Click on <b>OK</b>.</p>	

## Building the Mesh and Starting the Multiphysics Simulation

The mesh is constructed and the model submitted for analysis within a single dialog.

 <b>Create Database</b>	<p><b>Set Operations as Generate surface and volume meshes and create database.</b> Use the <b>Database name</b> <i>coupled_EM_stress_3currents.op3</i>.  <b>Set the Mesh Type as Prefer Mosaic.</b></p>  <p>Click on <b>Prepare and Solve</b>.</p>
---	--

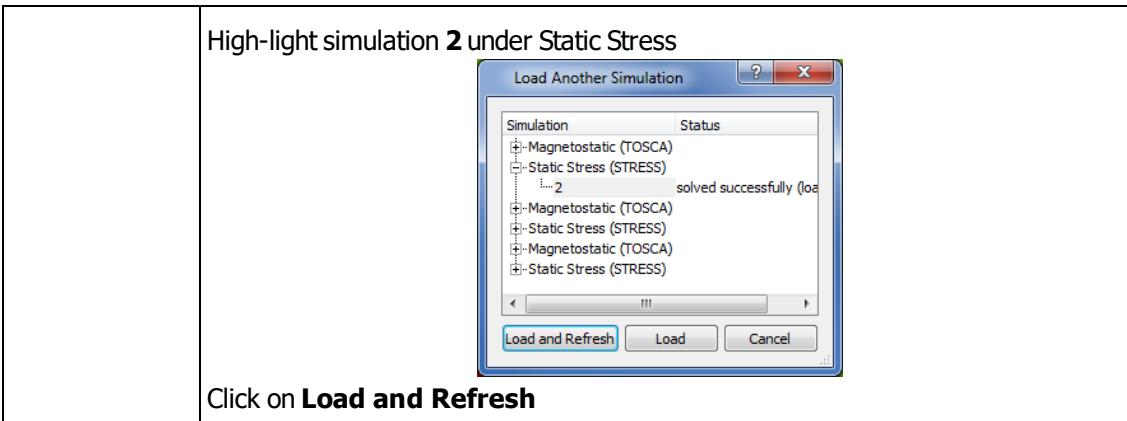
This completes the Modeller stage of this example.

The multiphysics simulation will run each magnetostatic analysis followed by the corresponding stress analysis, using body force density values computed automatically in the elements after the magnetostatic analysis has completed. Consequently, there will be a total of 6 simulations.

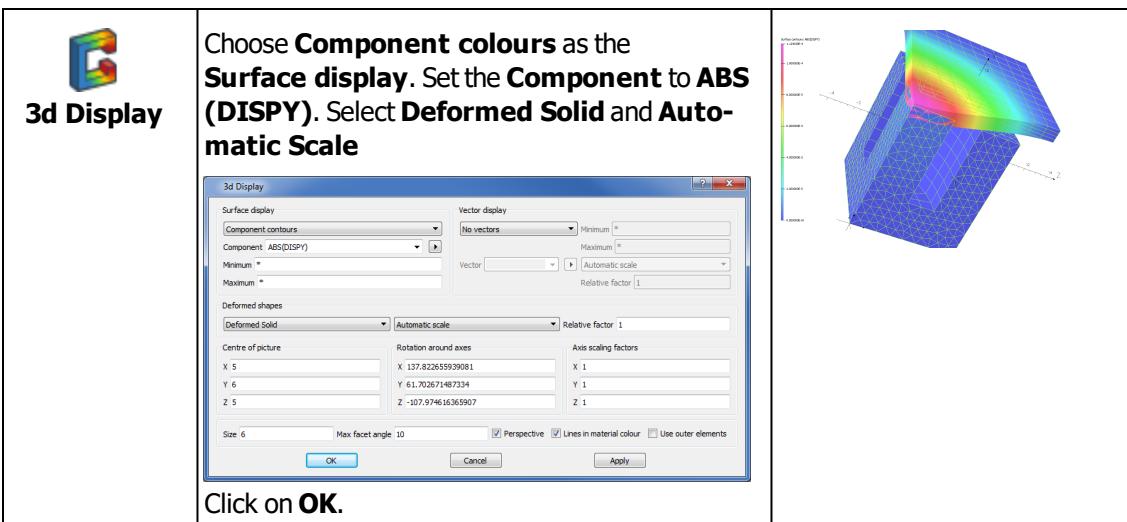
When the analyses are complete, launch the Post-Processor from the Modeller using **Launch Post-Processor**. Alternatively, launch the Post-Processor from the simulation console. This will load and display case 1 of the multiphysics simulation database (***coupled\_EM\_stress\_3currents.op3***).

## Post-Processing

The database contains the results from six simulations. To display the results from the first coupled stress analysis (where the forces were calculated from a magnetostatic simulation with scaling factor = 1.0), it is necessary to load in case 2. The dialog for selecting the second case will already be displayed.



Note that the results from the stress simulation are easier to see if the coil is not displayed with the geometry.



As expected, the maximum y-deflection (about -1.1 microns) is seen at the centre of the plate.

**3d Display**

Set the **Component** to **STRESS1** (either by typing or selecting from the pull right arrow). In **Deformed shapes**, select **None**

Click on **OK**.

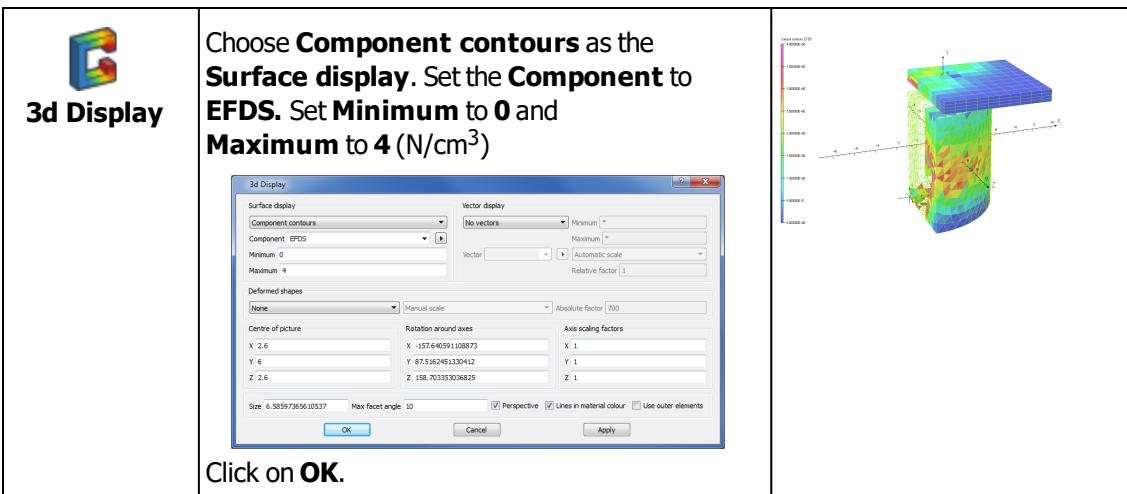
The element force density was calculated in the magnetostatic simulation but this can also be displayed in the stress simulation.

**3d Display**

Choose **Component colours** as the **Surface display**. Set the **Component** to **EFDS**.

Click on **OK**.

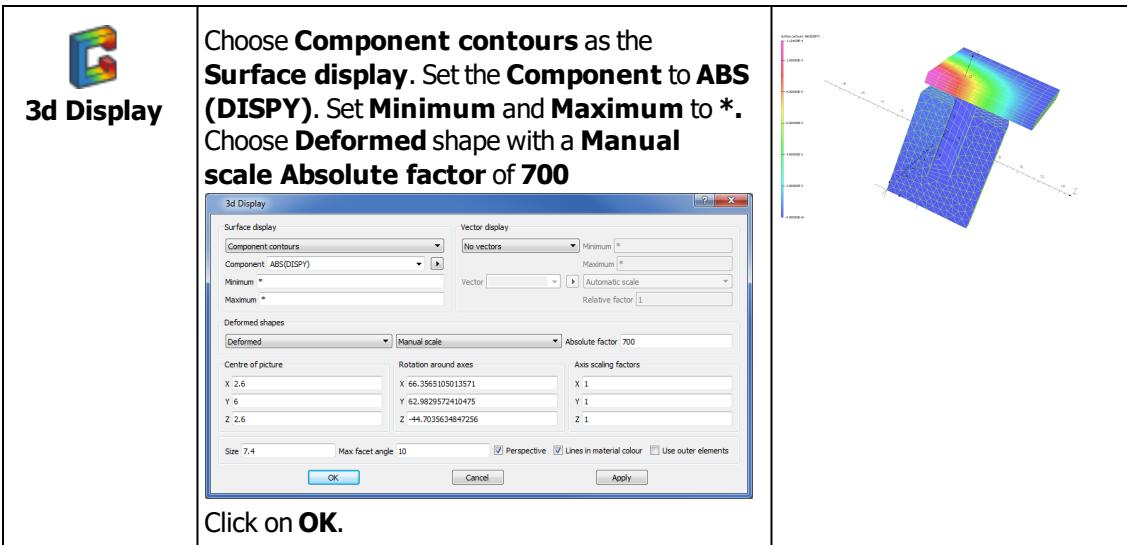
Although the maximum deflection was in the plate, the highest force densities are in the yoke. The density in the plate can be obtained by limiting the range of the force density displayed.



## Results from other simulations

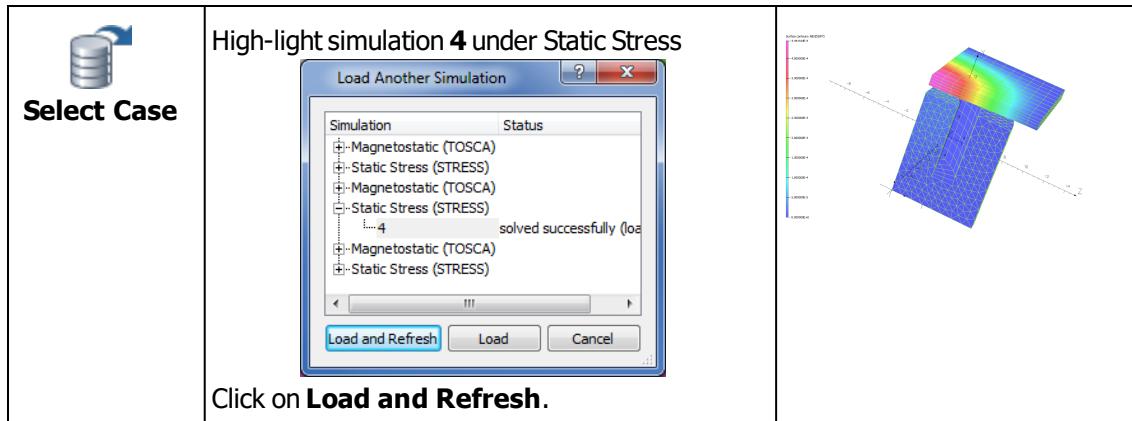
It is possible to display similar results from the other simulations in the database. One interesting comparison is the relative amount of displacement for each excitation scaling factor of the coil.

To examine this meaningfully, it is important to use the same scaling factor for the distorted mesh/geometry in each case.

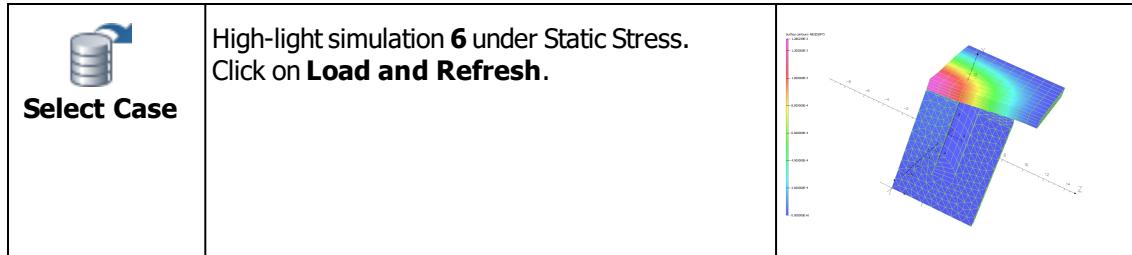


Even with a scaling factor of 700, the displacement can barely be seen. However, this value has been chosen so that for the highest excitation, the displacement is not too large.

A display with the same attributes for the deflection produced when the current in the coil is multiplied by a scaling factor of 2.0 can be displayed simply by opening a different case.



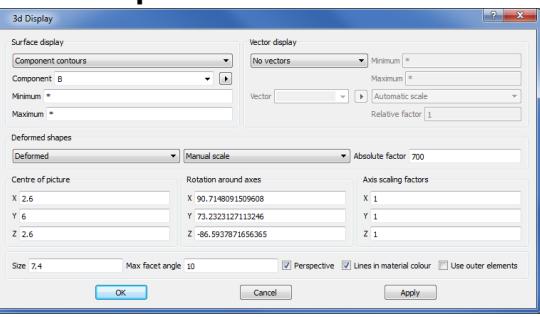
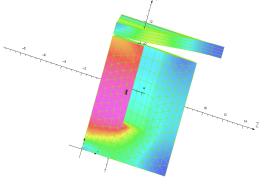
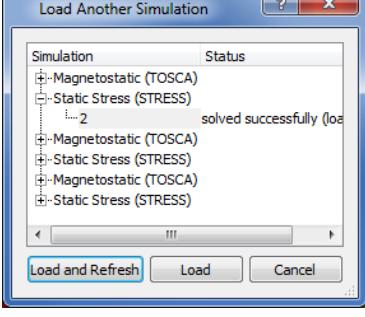
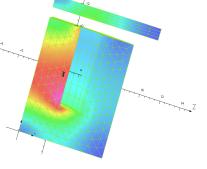
The maximum displacement is about 5 microns.



The effects due to the magnetic non-linearity of the steel material are clearly seen here. If the material was linear, the forces in the plate would have increased by  $2.5^2$  ( $= 6.25$ ) between a current scaling factor of 2.0 and 5.0. However, as can be seen the maximum displacement has only increased to about 13 microns from 5 microns. Of course, the mechanical properties of the steel also determine the displacement.

The non-linear magnetic effects can also be seen by displaying the flux density values for each case. These are available in both the magnetostatic simulation or the corresponding stress simulation results, even though the stress simulation in this model does not need the values.

Another multiphysics magnetostatic and static stress example where the flux density values are used is shown in [Coupled Electromagnetic and Stress \(Magnetostriction\) \[page 746\]](#).

 <b>3d Display</b>	<p><b>Set the Component to B.</b></p>  <p>Click on <b>OK</b>.</p>	
 <b>Select Case</b>	<p><b>High-light simulation 2 under Static Stress.</b></p>  <p>Click on <b>Load and Refresh</b></p>	

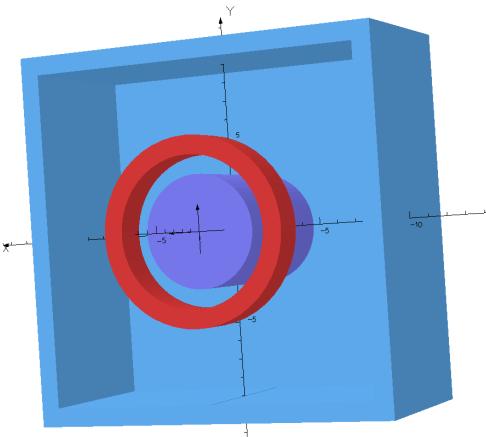
This completes the multiphysics stress analysis example.

# **Chapter 13**

## **An Example Using the Pre-Processor**

### **A Simple Model of an Inductor**

The first example using the Pre-Processor is an enclosed inductor core with two air gaps. A cut away diagram is shown in [Figure 13.1](#).



*Figure 13.1 A section of the example inductor core*

This example model will be created in the Pre-Processor and analysed using both TOSCA and ELEKTRA analysis modules. The results of this analysis will then be examined using the Post-Processor.

The model has symmetry, and only 1/8 of the model needs to be built. The complete model can be reconstructed later in the Post-Processor.

## The Base Plane of the Model

The model is built using a *Base Plane and Extrusion* method. This requires that a 2-dimensional section of the model is created. This is then extruded in the third dimension to build a 3-dimensional structure.

The method used to define the base plane requires a series of steps:

1. Specify points in the base plane that will be used to define the cross-section geometry.
2. Use the points to create *facets*. These are surfaces which define the cross-sectional geometry of the model.
3. Divide the facets to form a 2-dimensional finite element mesh.

### Starting the Pre-Processor

Use the Opera Manager to select a project folder and then start the Pre-Processor using the toolbar icon: . For more information, see the ***Opera Manager User Guide***.

### Defining the Base Plane Points

#### Setting the coordinate system and axis size

Before starting to define the base plane, switch on the 3d viewer. The 3d viewer is optional, and is not required for building a model.

**DEFINE -> Define new mesh -> No 3d Viewer (toggles to Show 3d view of the model)**

By selecting the 3d viewer in this way, a separate window will pop up automatically after the first layer of elements has been created (see later).

To set the coordinate system of the base plane, select

**DEFINE -> Define new mesh -> Finite element mesh -> XY plane, extrude in Z**

This specifies that the base plane will be in the XY plane and the extrusion direction is along the Z axis. To specify where on the Z axis the base plane should be positioned, set

**W coordinate of plane = 0**

To enter the points to define the base plane, an initial display size should be specified. To do this select

Minimum on horizontal axis	=	0
Maximum on horizontal axis	=	10
Minimum on vertical axis	=	0
Maximum on vertical axis	=	10
Accept		Dismiss

and **Accept** the settings. Then select **Point Input** from the **Define Baseplane** menu.

## Construction lines

Construction lines may be used to define the geometry of the base plane. These lines do not form part of the geometry but act as an aid to specifying coordinates. Specify the following construction lines to help with point definition.

**... Construction lines -> Enter C\_Lines -> By parameters -> Arc**

Arcs may be specified by a centre point, start and finish points (in polar coordinates). To do this, specify the following:

Centre U	=	0
..... V	=	0
Start R	=	3
...Theta	=	0
Finish R	=	3
...Theta	=	90
Accept		Dismiss

and **Accept** the settings. Note that if the start and end radii are not the same, this will generate an Archimedean spiral.

Add a second arc using:

Centre U	=	0
..... V	=	0
Start R	=	2
...Theta	=	0
Finish R	=	2
...Theta	=	90
Accept		Dismiss

Similarly, a number of straight construction lines are required and may be specified by start and finish points with an angle of rotation. Enter these using **Line**:

Start.	U	=	0
.....	V	=	0
Finish	U	=	10
.....	V	=	0
Rotation		=	0
Accept		Dismiss	

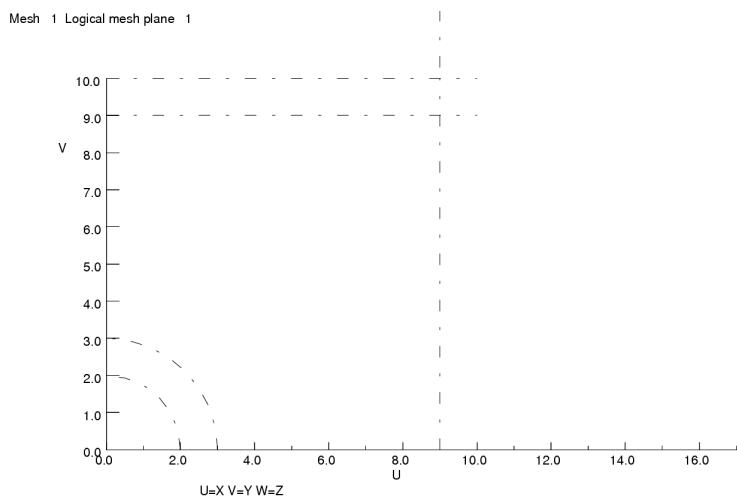
Start.	U	=	0
.....	V	=	10
Finish	U	=	10
.....	V	=	10
Rotation		=	0
Accept		Dismiss	

Start.	U	=	0
.....	V	=	9
Finish	U	=	10
.....	V	=	9
Rotation		=	0
Accept		Dismiss	

Start.	U	=	<b>9</b>
.....	V	=	<b>0</b>
Finish	U	=	<b>9</b>
.....	V	=	<b>15</b>
Rotation		=	<b>0</b>
	Accept		Dismiss

These lines are shown in Figure 13.1. Now **Return** three times to the **Point Definition** menu.

*Figure 13.2 Display of construction lines*



## Construction line intersections

The construction line intersections and line ends may be used to position the points required in the base plane. Use **At C\_line intersection** and follow this by using the cursor to *click on*<sup>1</sup> the ends and intersections of all the construction lines you have defined, i.e. at the following coordinates<sup>2</sup>:

0, 0

0, 2

0, 3

0, 9

0, 10

2, 0

<sup>1</sup>Click on means placing the cross cursor near the point and pressing the left mouse button. The cursor does not have to be placed precisely on the point

<sup>2</sup>The function key F1 or PF1 acts as a toggle switch to hide and reinstate the menus

```

3, 0
9, 0
9, 9
9, 10
10, 0
10, 9
10, 10

```

It is also possible to draw construction lines using existing points. To do this select

**... -> Construction  
 lines -> Enter  
 C\_lines -> By picking  
 points -> Straight line**

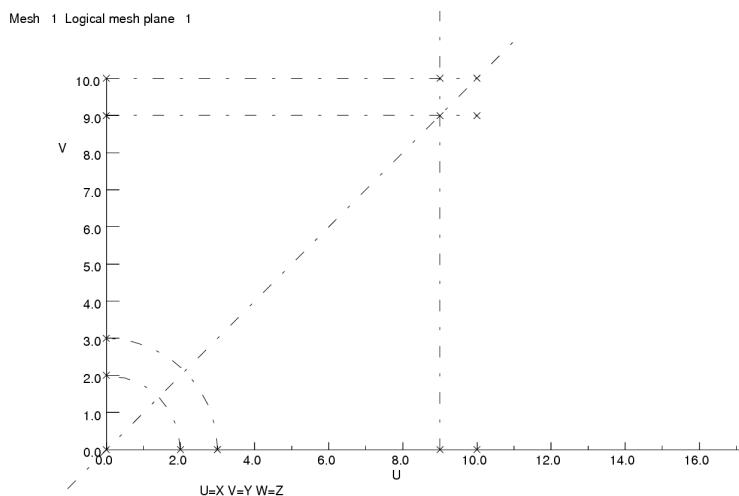
Select **Pick 2 points** and then *click on* the following points:

```

0, 0
10, 10

```

A construction line is drawn as shown in [Figure 13.1](#).



*Figure 13.3 Display Showing Additional Points and Construction lines*

Leave construction line input with **Return** three times. This will bring you back to the **Point Definition** menu.

Now we use this new construction line to place two points at the intersection by selecting

**... At C\_line intersection**

and *click on* the points:

```

1.5, 1.5

```

2.4, 2.4

### Cartesian coordinates

Points can also be defined explicitly in terms of their coordinate positions using cartesian or cylindrical polar coordinate systems. To enter points using cartesian coordinates select in the **Point Definition** menu

... Give U, V, W

and enter the following:

Cartesian Coordinate Input	
U coordinate	1
V coordinate	1
W coordinate	0
Accept	Exit

and **Accept**. Follow this with the next two points:

Cartesian Coordinate Input	
U coordinate	1
V coordinate	0
W coordinate	0
Accept	Exit

Cartesian Coordinate Input	
U coordinate	0
V coordinate	1
W coordinate	0
Accept	Exit

**Accept** both points and then **Exit** the cartesian coordinate input.

## Polar coordinates

To enter points using cylindrical polar coordinates select in the  
**Point Definition** menu ... **Give R, Theta, W**

Polar Coordinate Input	
R coordinate	2
T coordinate	22.5
W coordinate	0
Accept	Exit

and **Accept**.

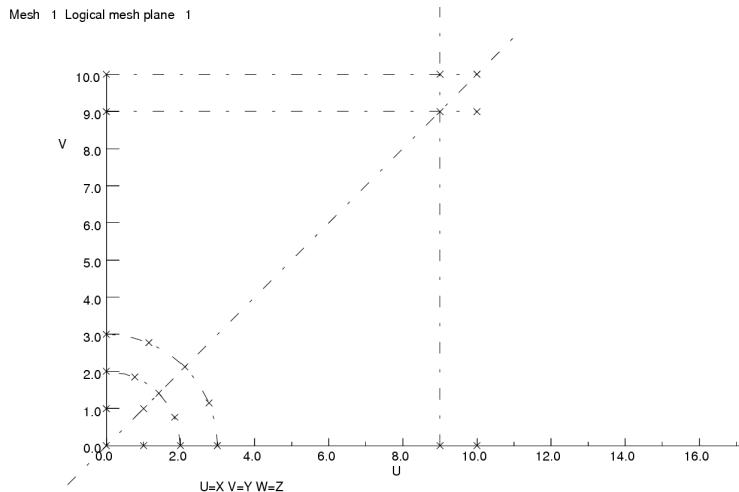
Follow this with points at the following coordinates:

Polar Coordinate Input	
R coordinate	3
T coordinate	22.5
W coordinate	0
Accept	Exit

Polar Coordinate Input	
R coordinate	3
T coordinate	22.5+45
W coordinate	0
Accept	Exit

Polar Coordinate Input	
R coordinate	2
T coordinate	67.5
W coordinate	0
Accept	Exit

This gives the display shown in [Figure 13.1](#).



*Figure 13.4 Display Showing the Polar Coordinate Points*

Select **Exit** to leave the polar coordinate input. Press **Return** to close the **Point Definition** menu.

## Defining the Base Plane Facets

The base plane points have now been positioned and these are used to form the base plane facets. To do this select **Facet Input** from the **Define Baseplane** menu.

The **Facet Definition** submenu is displayed. The simplest facet to define consists of four corner points. Select **auto-close after 4** and *click on* the following points:

- 0, 10
- 9, 10 (this creates Facet 1)
- 9, 9
- 0, 9

to complete the first facet. This is shown in [Figure 13.1](#)<sup>1</sup>

---

<sup>1</sup>N.B. If you create an incorrect facet, you can delete it using **Delete facet**

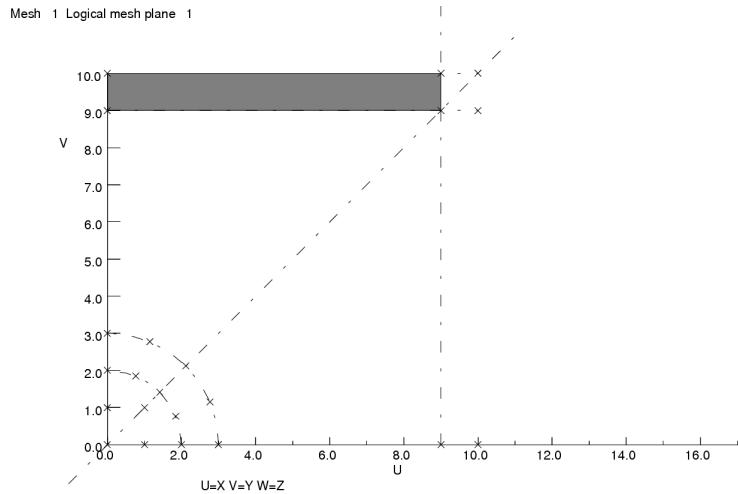


Figure 13.5 Display of the first facet.

Continue to create facets by selecting **auto-close after 4** and *click on* the following points:

9, 10

10, 10 (Facet 2)

10, 9

9, 9

9, 9

10, 9 (Facet 3)

10, 0

9, 0

0, 1

1, 1 (Facet 4)

1, 0

0, 0

To create facets with curved edges, points should be used as mid-side nodes. To do this select **no auto-close** and *click on* the following points:

9, 0

9, 9 Three corners of Facet 5

2.4, 2.4

Select **Mid-side** and *click on*

3, 1.2 Mid-side node creates curved edge of Facet 5

Select **no auto-close** and *click on*

3, 0 Final corner of Facet 5

and close the facet with **Close**.

Continue by using the following menu selections and points:

**no auto-close** 2.4, 2.4

9, 9

0, 9 Facet 6

0, 3

**Mid-side** 1.2, 2.7

Note how the facet is closed automatically when a mid-side point is selected for the fourth side.

**no auto-close** 0, 3

0, 2

**Mid-side** 0.7, 1.9

**no auto-close** 1.5, 1.5 Facet 7

2.4, 2.4

**Mid-side** 1.2, 2.7

**no auto-close** 2.4, 2.4

1.5, 1.5

**Mid-side** 1.9, 0.8

**no auto-close** 2, 0 Facet 8

3, 0

**Mid-side** 2.8, 1.2

**no auto-close** 2, 0

1, 0

1, 1 Facet 9

1.5, 1.5

**Mid-side** 1.9, 0.8

**no auto-close** 0, 2

0, 1

1, 1 Facet 10

1.5, 1.5

**Mid-side** 0.8, 1.9

This completes the facet definition. The display of all the base plane facets is shown in [Figure 13.1](#).

Press **Return** to close the **Facet Definition** menu.

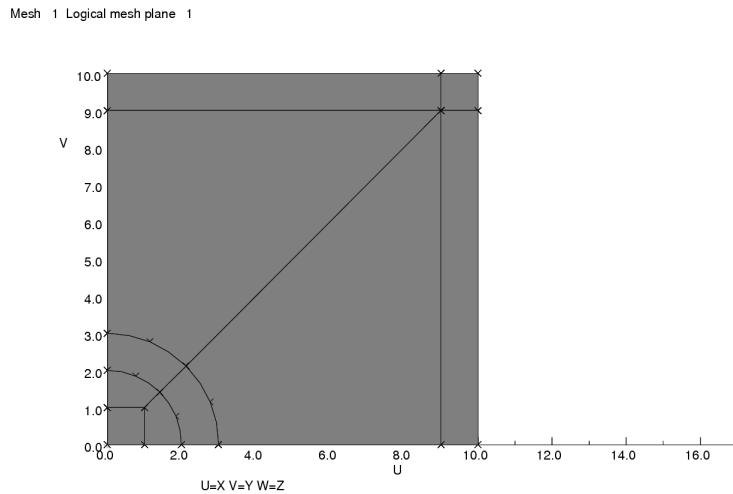


Figure 13.6 Display of all the facets

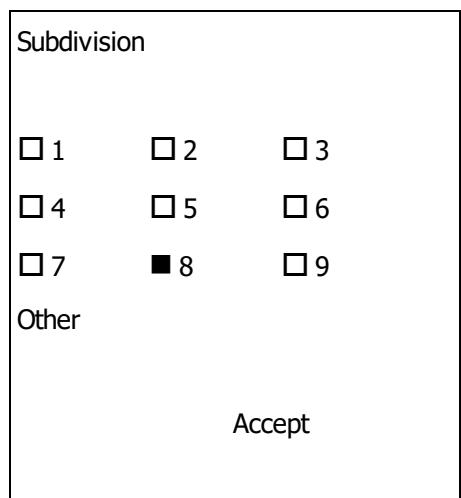
## Defining the Base Plane Subdivisions

Move to the subdivision definition menu by selecting **Subdivision** from the **Define Baseplane** menu.

The base plane subdivisions define the distribution of finite elements in the base plane. These may be defined globally (the same value over the whole base plane) or on individual lines (facet edges) or both.

Set the number of subdivisions to 8 by selecting **Set subdivision**.

Then *click on* button 8 and **Accept**.



Now select **Apply globally** to give all facet edges 8 divisions<sup>1</sup>. The display indicates the distribution of subdivisions. Clear the message box displayed and select **Return** twice to return to the **Define Baseplane** menu.

---

<sup>1</sup>Note that this method ensures all facet edges have subdivision defined and that the mesh is continuous

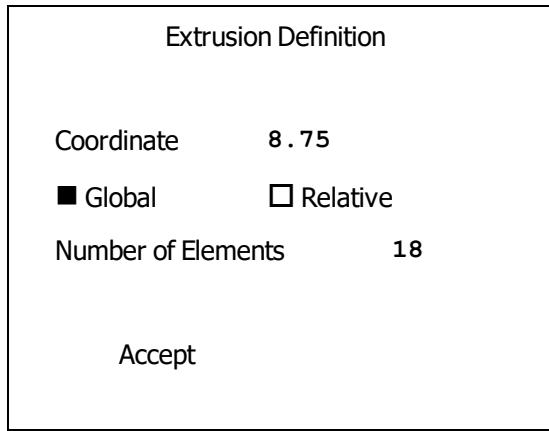
## Extruding in the Third Dimension

Once the base plane has been completed, the third dimension is defined by sweeping the base plane through space in the third dimension. To move to the extrusion definition, select

### Extrude

and select **Linear extrusion**. Note that once extrusions have been started, it is only possible to go back to add further facets to the base plane by removing all of the extrusions in the 3rd dimension using **DEFINE -> Re-define a mesh**.

Complete the subsequent dialog box as shown below. This sets the first extrusion to a coordinate of 8.75 with 18 subdivisions in the extrusion direction.



**Accept** this dialog box and clear the message box displayed. Select **Finish Editing** followed by **Finish** four times to close the submenus (materials and boundary conditions will be defined later). The first extrusion is now completed.

Note that since the 3d-Viewer was previously activated, a window now pops up after having created the first layer of elements.

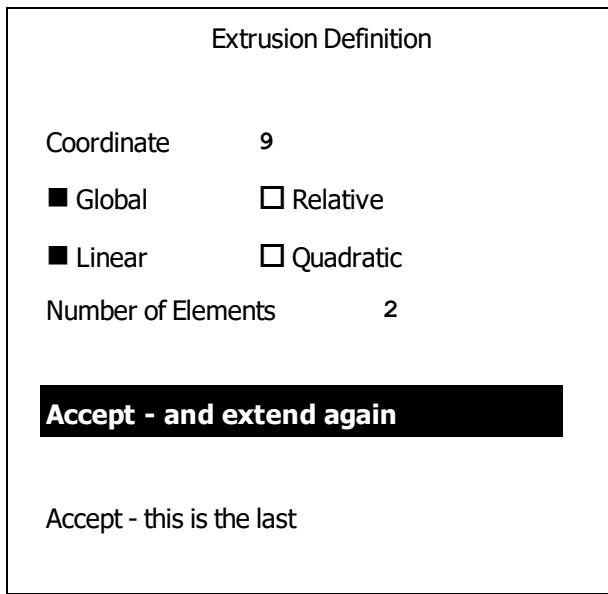
**Return** once to the top-level menu.

Extend the extrusion further by selecting

**DEFINE -> Extend existing mesh -> Extend without editing**

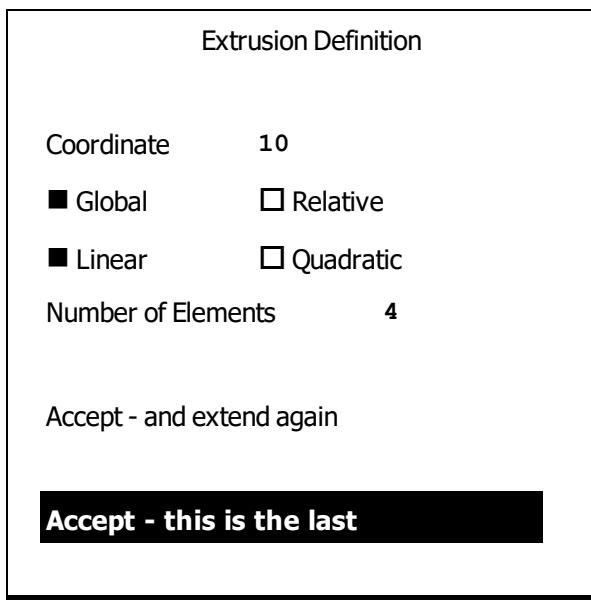
and **Accept** Mesh number = 1.

Select **XY plane, extrude in Z** from the coordinate system submenu to continue extruding in the same direction. To create a second linear extrusion complete the subsequent dialog box as follows:



accepting the extrusion with **Accept – and extend again**.

Repeat the procedure to extend the model to coordinate 10 with 4 subdivisions but this time use **Accept – this is the last**.



**Return** twice to the top-level menu.

The 3d-Viewer contains a wire-frame view of the model. The options of the 3d-Viewer will be discussed later in this chapter after having defined materials, boundaries and the conductor.

## Objects in the 3D Model

### Defining Materials and Potentials

The model is built up of different materials. These are used to describe the objects or parts of the model. So far, the whole model is made up of the default material – **AIR**.

It is also necessary to define the type of magnetic potential to be used for each material. For a detailed explanation of the choice of potentials please refer to the ***Opera-3d Reference Manual***. Note that in TOSCA, **TOTAL** and **REDUCED** are scalar potentials, and in ELEKTRA they are vector potentials.

To change the material definition from the default setting i.e. **AIR** and **TOTAL** potential, select

**MODIFY -> Material properties**

and **Accept** Layer number = 1.

On the display, the facets represent the volumes in the first layer of the model.

### Make everything air

From the **Materials** submenu select **Select and define** and *click on* any volume. A material definition dialog box is displayed, which should be completed as follows:

- Material name: **Air**
- Potential type: **Reduced**
- Element type: **Linear**
- Other volumes and layers<sup>1</sup>: **From 1 to \***
- All volumes

The complete dialog box should look like this:

---

<sup>1</sup>\* implies the highest value available i.e. maximum value or all layers.

**Accept** the settings. This redefines all regions to be **AIR** and **REDUCED** scalar potential. This is the correct setting for regions containing source conductors.

## Define CORE

Objects in the model are defined by their material names. To change from the **AIR** definition (from the **Materials** submenu) select

**Select/de-select volume** and select volumes by using the cursor to *click on* all but one of the volumes which form the central cylindrical part of the model. Use the following coordinates<sup>1</sup>

0.5, 0.5

0.5, 1.5

1.0, 2.5

1.5, 0.5

and then select **Select and define** and *click on* the last volume in the central part at coordinate 2.5, 1.0

Again a material definition dialog box is displayed. Complete this material's name, potential and element types as follows:

---

<sup>1</sup>If an incorrect volume is selected, *click on* the volume again to deselect it.

Material Definition

Material Name	<b>core</b>	
Potential Type:	Element Type:	
<input checked="" type="checkbox"/> Total	<input checked="" type="checkbox"/> Linear	
<input type="checkbox"/> Reduced	<input type="checkbox"/> Quadratic	
<input type="checkbox"/> Vector		
Options:		
Jx, Jy, Jz		
Vx, Vy, Vz		
Scalar: Charge Density or Rotational Velocity		
Scalar		
Packing factor		
Material orientation		
<input type="checkbox"/> Local XYZ=XYZ	<input type="checkbox"/> Local XYZ=YZX	<input type="checkbox"/> Local XYZ=ZXY
Other vector		
Other volumes and layers:		
From	To	<input type="checkbox"/> All volumes
Accept	Keep	Help
		Quit

and **Accept** the settings. This defines the central core of the device which should now be displayed in blue.

### Define BOX in layers 1 and 2

Repeat the procedure for the facets which form the box using the following coordinate with

#### Select/de-select volume

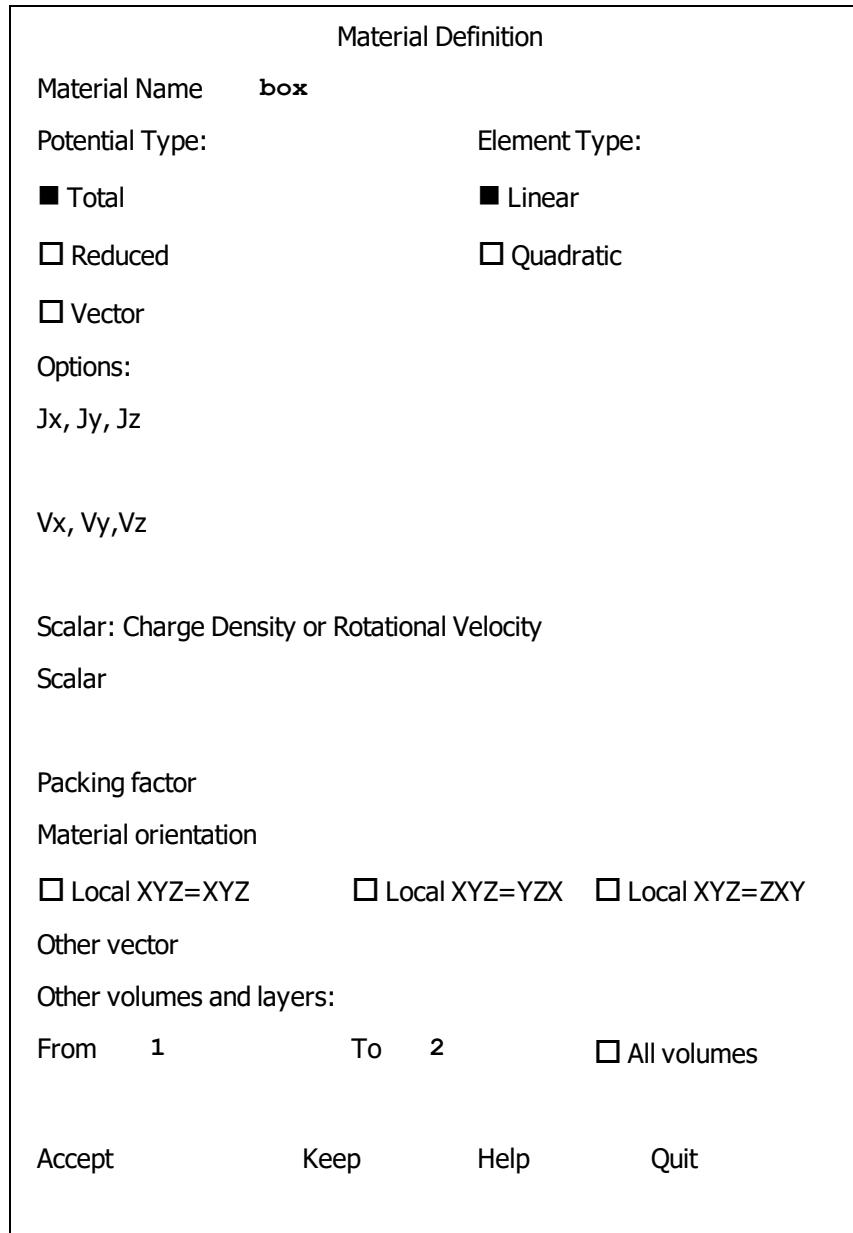
5.0, 9.5

9.5, 9.5

and **Select and define**

9.5, 5.0

Complete the subsequent dialog box with material name, potential and element types and layer numbers as shown.



This defines the sides of the outer case which should be displayed in a lighter shade of blue. To further see the effect of this (from the **Materials** submenu) select **Show properties**.

The material properties are displayed as shown in [Figure 13.1](#). Select

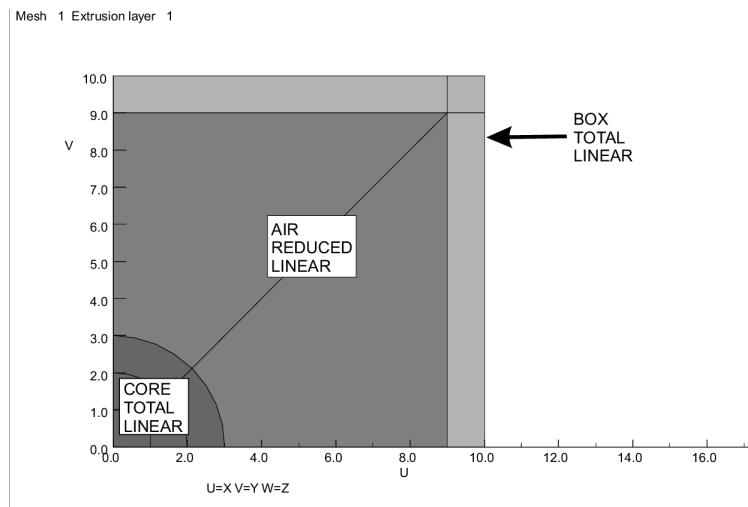


Figure 13.7 Display of material properties in Layer 1

**Finish** to close the submenu.

### Define BOX in layer 3

To complete the material definition, select

**MODIFY -> Material properties**

and

**Accept** Layer number = 3.

From the **Materials** submenu select **Select and define** and *click on* any volume. Complete the dialog box as follows to change **All** the volumes in this layer:

Material Definition			
Material Name	<b>box</b>		
Potential Type:	Element Type:		
<input checked="" type="checkbox"/> Total	<input checked="" type="checkbox"/> Linear		
<input type="checkbox"/> Reduced	<input type="checkbox"/> Quadratic		
<input type="checkbox"/> Vector			
Options:			
Jx, Jy, Jz			
Vx, Vy, Vz			
Scalar: Charge Density or Rotational Velocity			
Scalar			
Packing factor			
Material orientation			
<input type="checkbox"/> Local XYZ=XYZ	<input type="checkbox"/> Local XYZ=YZX	<input type="checkbox"/> Local XYZ=ZXY	
Other vector			
Other volumes and layers:			
From	To	<input checked="" type="checkbox"/> All volumes	
Accept	Keep	Help	Quit

and **Accept**. This defines the top of the outer case. To see the effect of this (from the **Materials** submenu) select **Show properties**.

The material properties are displayed as shown in [Figure 13.1](#).

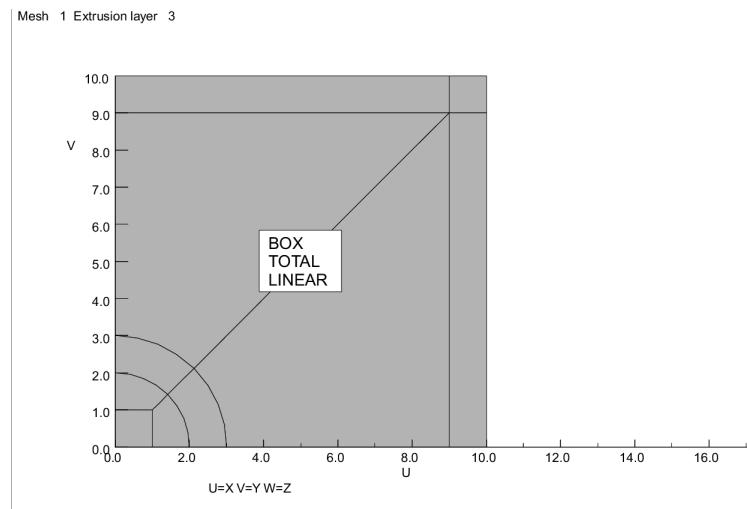


Figure 13.8 Display of material properties in Layer 3

Select **Finish** to close the submenu.

## Magnetic Boundary Conditions

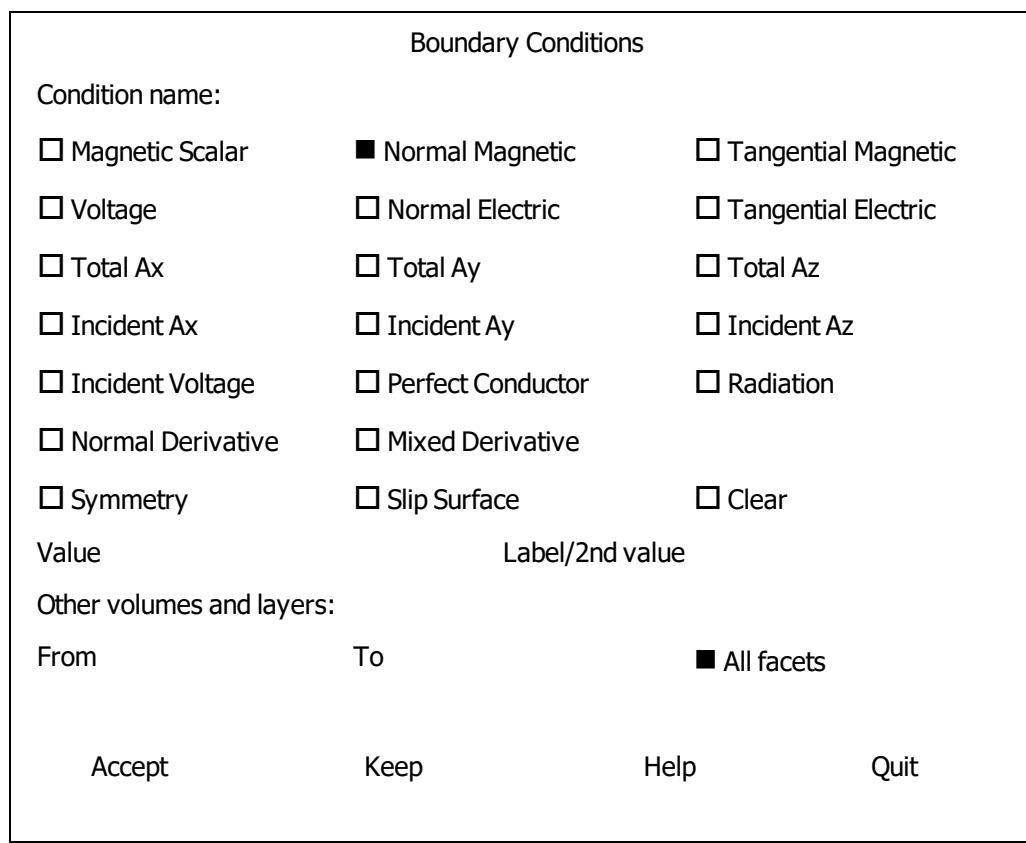
The magnetic field conditions on the model boundaries should now be specified.

### Base plane

To define boundary conditions on the base plane select

**MODIFY -> Boundary conditions -> Base plane -> Select and define**

and *click on* any facet. Complete the subsequent dialog box as follows to apply the **Normal Magnetic** condition to **All** facets:



and **Accept**.

This sets the magnetic field to be normal to the whole of the base plane and is required to give the correct reflection symmetry to the model. Select **Finish** to close the submenu.

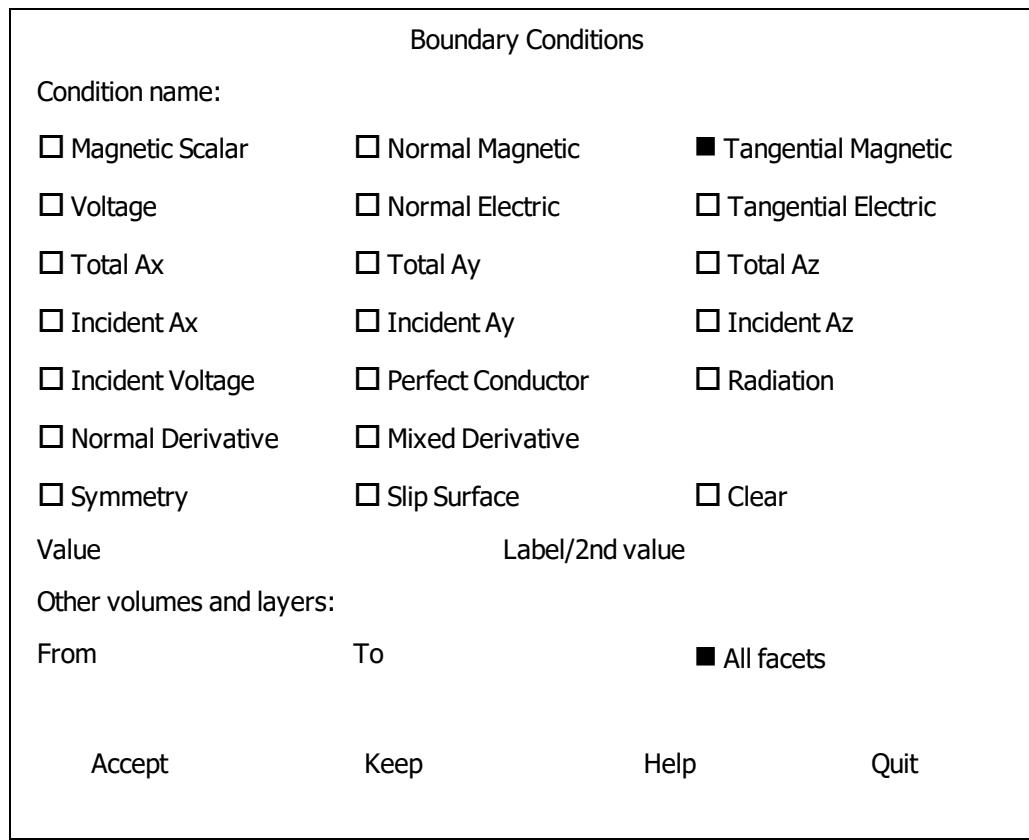
All the other outer surfaces of the model need the **Tangential Magnetic** condition.

## Top plane

To define the boundary conditions on the outer top surface of the box select

**...Top plane -> Select and define**

and *click on* any facet. Complete the subsequent dialog box as follows



and **Accept**.

This sets the magnetic field to be tangential to the whole of this plane and implies that there is no flux leakage from the box. Select **Finish** to close the submenu.

## Extruded facets

To define the boundary conditions on the extruded facets select **Extrusion facets**.

Then set Layer number = 1 and **Accept**.

Select **Select/de-select facet**. In the picture the *lines* represent *facets* orthogonal to the plane of the screen. Select each of the facets around the outside of the model by *clicking on* the following coordinates:

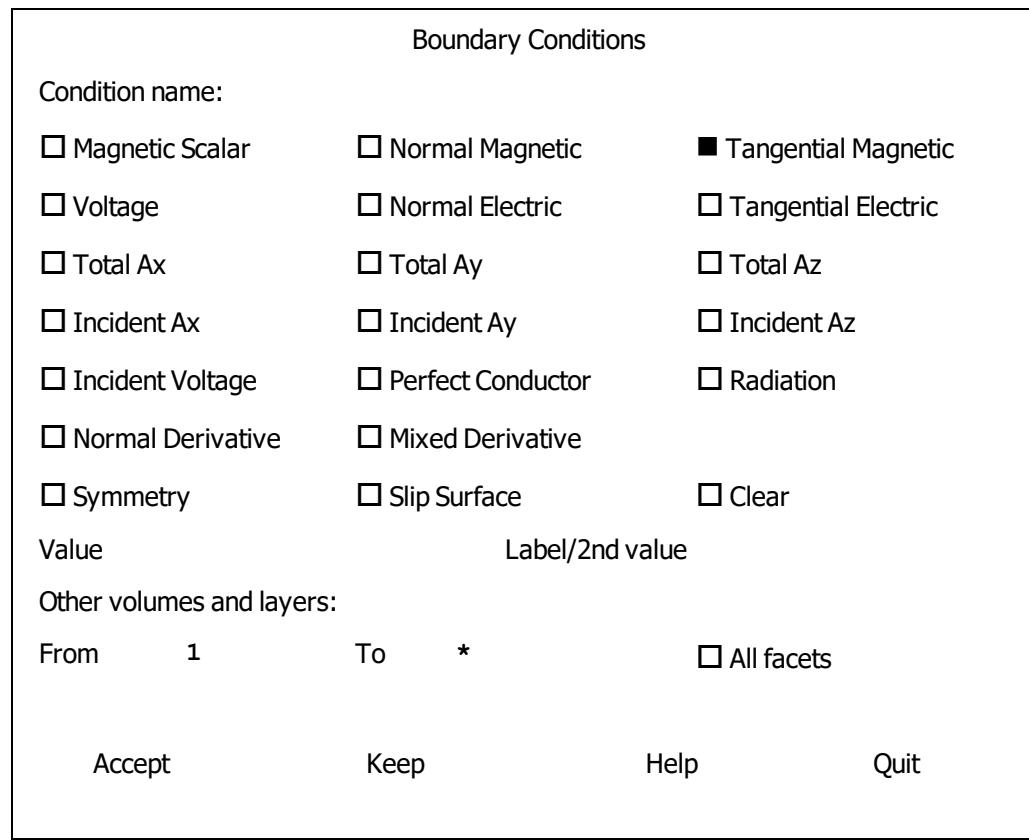
0.1, 0.5  
0.1, 1.5

0.1, 2.5  
 0.1, 6.0  
 0.1, 9.5  
 5.0, 9.9  
 9.5, 9.9  
 9.9, 9.5  
 9.9, 5.0  
 0.5, 0.1  
 1.5, 0.1  
 2.5, 0.1  
 9.5, 0.1

Then select **Select and define** and *click on* coordinate:

5.0, 0.1

Complete the subsequent dialog box to apply the boundary condition to all layers as follows:



**Accept** and this sets the magnetic field to be tangential on the outer surfaces of the box and implies that there is no flux leakage from the box. Select **Finish** to close the submenu followed by **Return** twice. This completes the boundary condition definition of the model.

## Displaying the 3D Model

To examine the 3D model select

**DISPLAY -> Display Command... ...view**

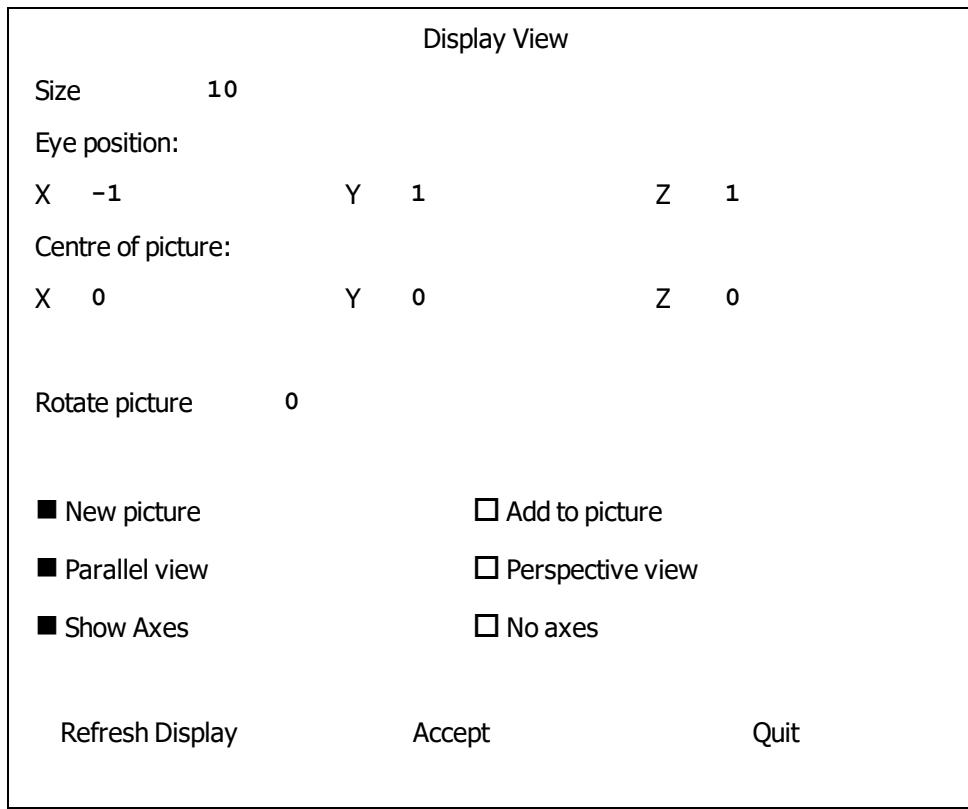
and complete the parameter box:

Display View

Size	10		
Eye position:			
X	1	Y	1
Z	1		
Centre of picture:			
X	0	Y	0
Z	0		
Rotate picture	0		
<input checked="" type="checkbox"/> New picture	<input type="checkbox"/> Add to picture		
<input checked="" type="checkbox"/> Parallel view	<input type="checkbox"/> Perspective view		
<input checked="" type="checkbox"/> Show Axes	<input type="checkbox"/> No axes		
<input type="button" value="Refresh Display"/>	<input type="button" value="Accept"/>	<input type="button" value="Quit"/>	

and **Refresh display**.

A wire-frame display of the model is produced. Change the parameter box settings to give a different view by negating the x-coordinate of the eye position as follows:



and **Refresh display** again. Select **Return** to close the submenu.

To view the model with hidden surfaces removed, it is first necessary to construct the surface mesh. This is done using:

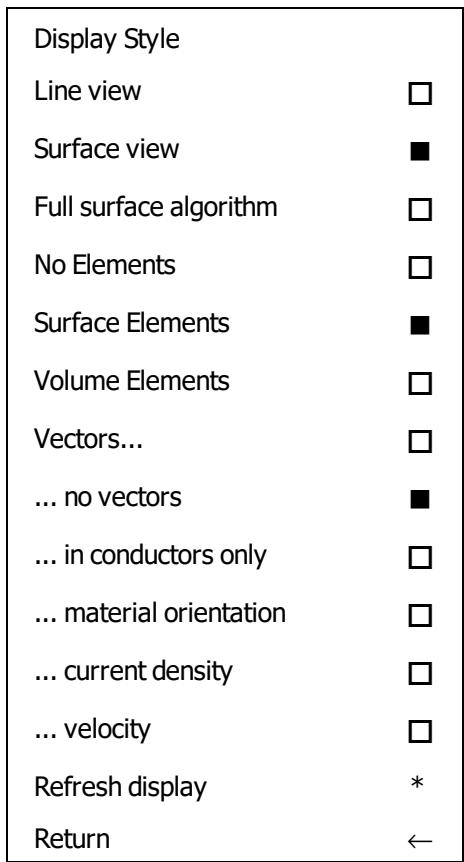
**MESH -> ...quadrilaterals**

Clear the message box, and select **Return**.

To now display the model, select

**DISPLAY -> Display Command... ...style**

and select the following settings of the subsequent dialog box:



Finish by selecting **Refresh display**.

**Return** twice to close the submenus. The model with the discretization of the mesh is shown in [Figure 13.1](#).

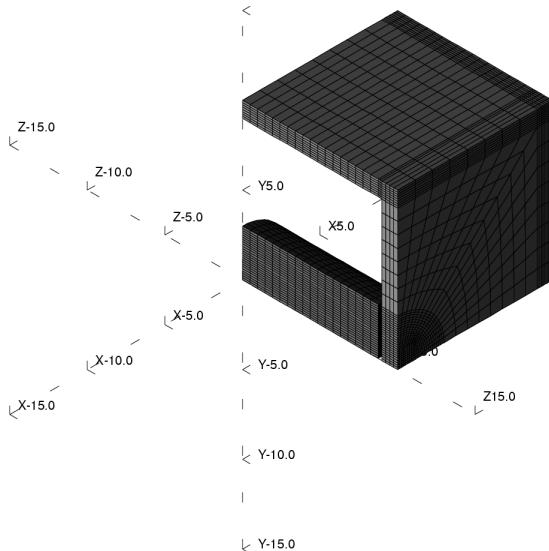


Figure 13.9 Display of 3D finite element mesh

The final stage is to create the volume mesh. This is completed using:

**MESH -> Mesh**

and **Accept** to accept the default parameters. On successful completion of the volume mesh, clear the message box.

If desired, the volume mesh can also be displayed, but for large models it is warned that this can be a slow procedure.

## Defining the Conductor

The conductors in Opera-3d do not form part of the finite element mesh but are assumed to be source conductors. They are defined separately from the mesh. To do this select

**DEFINE -> Conductors -> Define a  
conductor -> Generally  
orientated set -> Solenoid**

A sequence of parameter boxes and menus is now displayed. Complete them and **Accept** as follows:

Local coord 1: X - origin	=	0
Local coord 1: Y - origin	=	0
Local coord 1: Z - origin	=	0
Accept		

Local Coordinate system 1

XYZ local = XYZ global	<input checked="" type="checkbox"/>
XYZ local = YZX global	<input type="checkbox"/>
XYZ local = ZXY global	<input type="checkbox"/>
Other system	<input type="checkbox"/>
Return	←

Local coord 2: X - origin	=	0
Local coord 2: Y - origin	=	0
Local coord 2: Z - origin	=	0
Accept		

## Local Coordinate system 2

XYZ local = XYZ global XYZ local = YZX global XYZ local = ZXY global Other system Return 

Cross-section: X1 = 4

Cross-section: Y1 = -1

Cross-section: X2 = 5

Cross-section: Y2 = -1

Accept

Cross-section: X3 = 5

Cross-section: Y3 = 1

Cross-section: X4 = 4

Cross-section: Y4 = 1

Accept

Curvature CU1 = 0

Curvature CU2 = 0

Curvature CU3 = 0

Curvature CU4 = 0

Accept

```
Current Density      =  1000
Symmetry code       =  1
Drive label         =  ONE
Accept
```

## Conductor reflections

- |                        |                                     |
|------------------------|-------------------------------------|
| No reflection in XY(1) | <input checked="" type="checkbox"/> |
| + reflection in XY(1)  | <input type="checkbox"/>            |
| - reflection in XY(1)  | <input type="checkbox"/>            |
| No reflection in YZ(1) | <input checked="" type="checkbox"/> |
| + reflection in YZ(1)  | <input type="checkbox"/>            |
| - reflection in YZ(1)  | <input type="checkbox"/>            |
| No reflection in ZX(1) | <input checked="" type="checkbox"/> |
| + reflection in ZX(1)  | <input type="checkbox"/>            |
| - reflection in ZX(1)  | <input type="checkbox"/>            |
| Return                 | *                                   |

```
Tolerance on flux density      =  0.0001
```

```
Accept
```

Select **Return** four times to close all the menus and select

**DISPLAY -> Display command... ...refresh display**

to see the superposition of the solenoid on the finite element mesh. This is shown in [Figure 13.1](#).

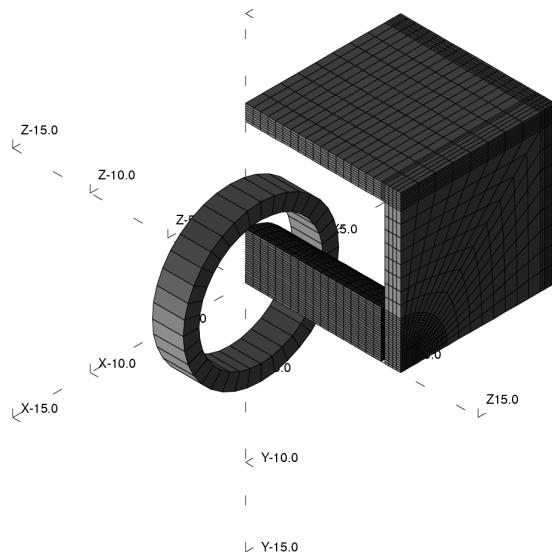


Figure 13.10 Display of mesh with solenoid

## Examining the Model with the 3d Viewer

On a machine with OpenGL support the model can be examined with the "3d Viewer". In this worked example the 3d Viewer is being used to examine the position of the conductor.

```
DISPLAY -> 3d Viewer...
...style
Surface elements
DISPLAY -> 3d Viewer...
...refresh display
```

This will create a new display in the 3D viewer window of the Pre-Processor window. Depending on your installation this new 3d Viewer window may overlap your existing Graphics Window in the Pre-Processor window. Resize and move the new 3d Viewer window to any suitable location. Figure 13.11 shows a screenshot of a Windows installation, where the 3d Viewer is overlapping part of the Graphics Window.

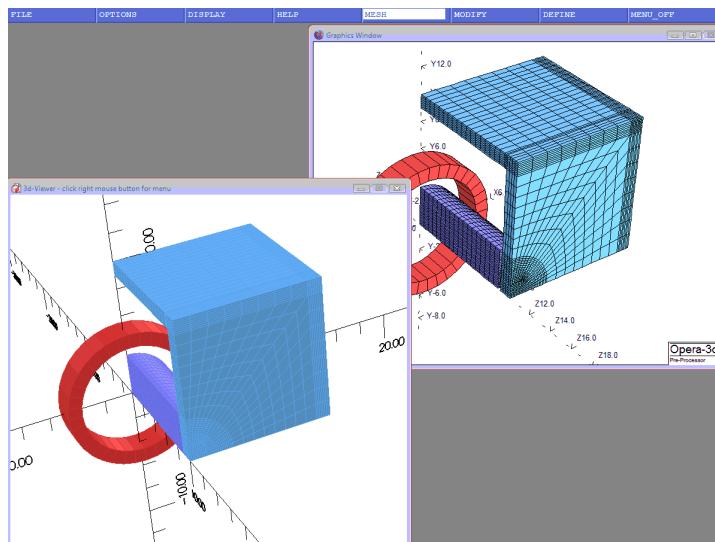


Figure 13.11 3d Viewer after initialization

The view within the 3d Viewer can be manipulated using the mouse. The main control is the left mouse button. Moving the mouse with the left button pressed changes the view of the model. The way in which the view is changed and other options can be selected from a menu which pops up when the right mouse button is clicked.

After initialization, the **Rotate** menu option is selected by pressing the right mouse button, and selecting **Rotate** from the menu.

Rotate the model to obtain a view similar to the screenshot in Figure 13.12 by holding the left mouse button, and dragging the mouse.

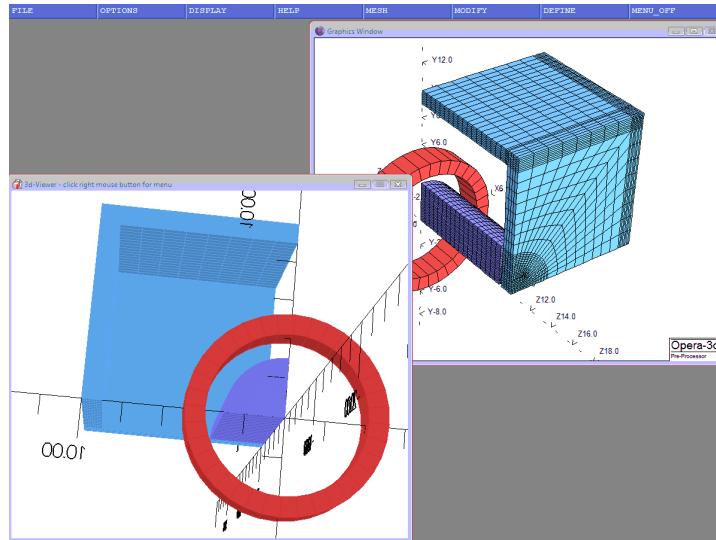


Figure 13.12 3d Viewer after rotation

Click the right button and select the **Translate** option. Move the centre of the coil to a location in the middle of the 3d Viewer window (again by holding down the left mouse button and dragging the mouse). Figure 13.13 shows a screenshot after the translation.

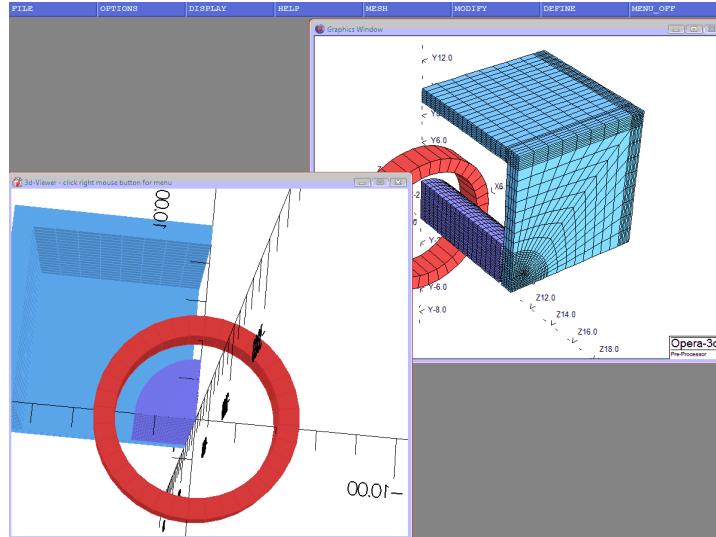


Figure 13.13 3d Viewer after translation

Next click the right button and select the **Zoom** option. Enlarge the model by dragging with the left mouse button pressed. Figure 13.14 shows a screenshot after zooming in.

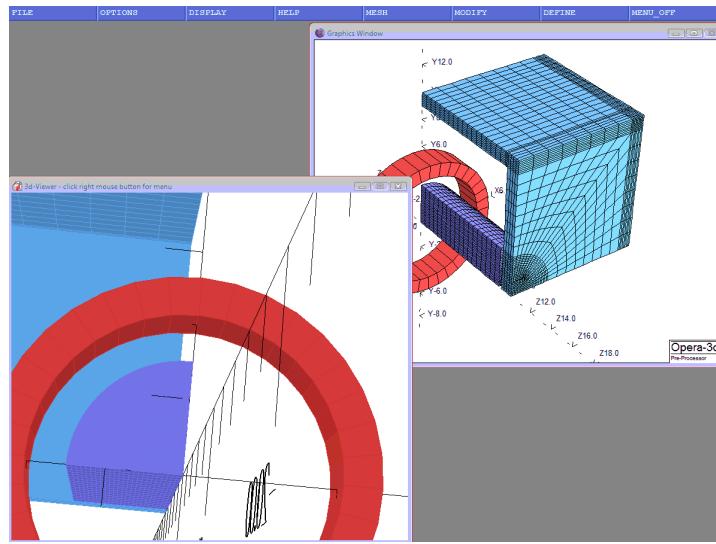


Figure 13.14 3d Viewer after zooming in

This view can also be copied to the Graphics Window by selecting

**DISPLAY -> Display Command... ...copy 3d view**

Figure 13.15 shows the results.

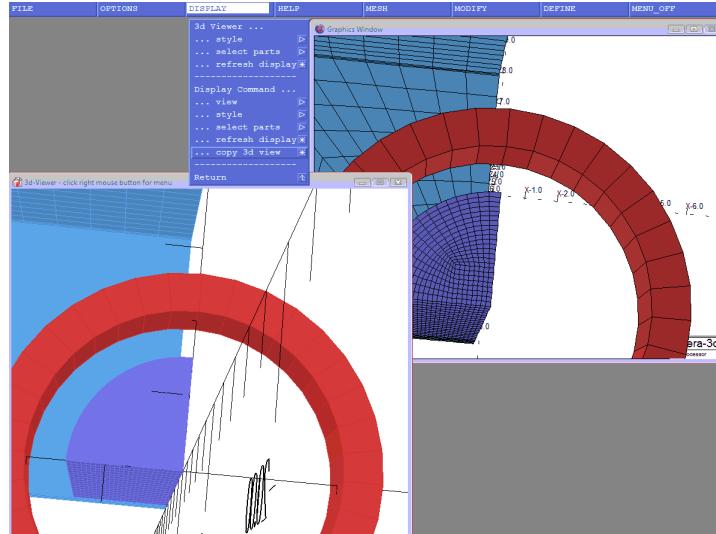


Figure 13.15 Screenshot after copying the 3d view to the graphics window

## Saving TOSCA Data

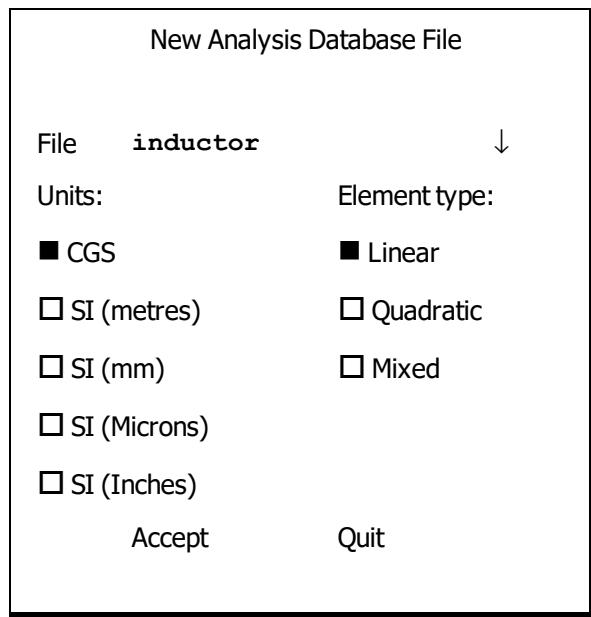
The instructions issued to create the model should be saved so that the pre-processing may be repeated or modified as required. If this is done *after* creating the analysis data, the analysis options will be recorded as well.

### Creating the TOSCA Database File

If you have a license for TOSCA and wish to solve the solenoid model as a static solution, select

**FILE -> Analysis**  
**... create new database -> Statics (TOSCA)**  
**Magnetostatics**

and select the units and element type to be used by completing the parameter box as shown below:



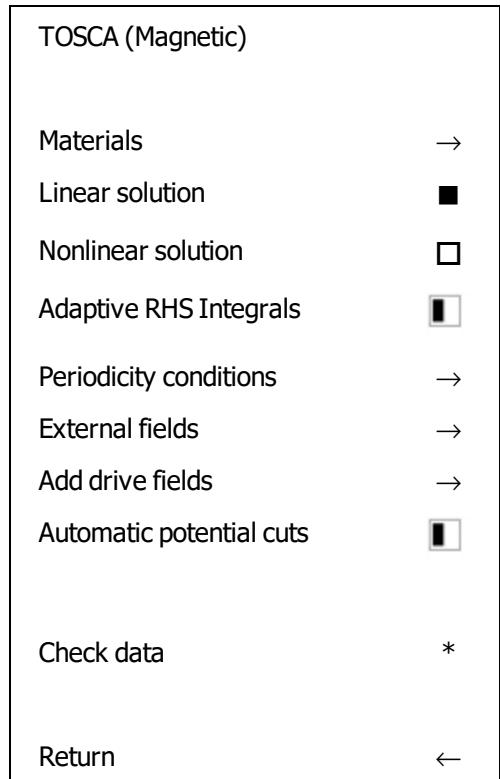
followed by **Accept**. If the box ↓ to the right of the file name is selected, a file selection box is raised, allowing all existing files to be shown/selected.

The element type is selected to be linear. This forces all elements in the model to have a linear interpolation in the solver. If the element type is set to mixed, the linear/quadratic information, which was given through the material definition stage, will be used. If the element type is set to quadratic, all elements in the model will be quadratic, which will give a greater accuracy than using linear elements. However the overall number of equations will increase, so this option must be used with care. It is recommended that quadratic hexahedra are used wherever possible, to ensure the greatest accuracy.

After a short pause whilst the main database is prepared, a further parameter box is presented, allowing a title to be specified for the model. Any number of lines of text can be input, and the title is terminated by typing a single \* character on a line on its own.

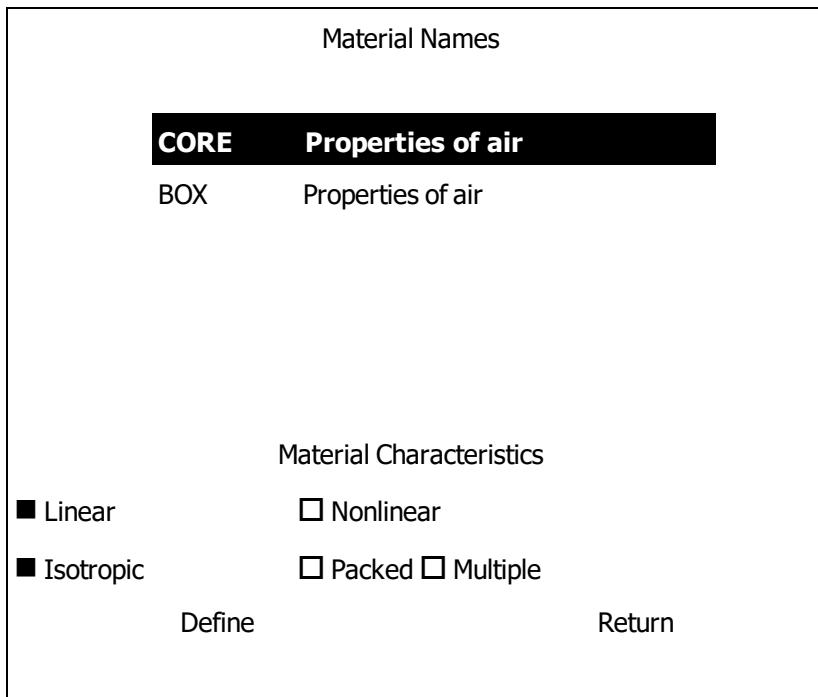
A prompt is then given for a scaling factor. This is used to scale all coil current densities. A value of 1 should be given.

Having cleared the message box, the **TOSCA (Magnetic)** data menu appears



The material properties of the different media in the model are first defined, by selecting **Materials**. For each material in the model, the linear or nonlinear characteristics can be defined.

In the present model, a dialog appears:



Selecting **CORE** first, the material properties can be defined. These are to be linear isotropic. Selecting **Define** allows the properties of the material for **CORE** to be defined. The parameter box that appears should be completed to set an isotropic permeability of 300, and coercive force remaining zero. Follow this by selecting **Accept**.

The same is done for material **BOX**, by high-lighting **BOX**, and selecting **Define**. Again, the isotropic permeability is set to 500, with zero coercive force, and select **Accept**. Close the material definition box by selecting **Return**.

We leave all other features in the **TOSCA (Magnetic)** data menu without changing; so we will perform a linear solution, and there are no external fields.

Select **Return** in the **TOSCA (Magnetic)** data menu, and check that the data is correct in the information box. Close the information box, to complete storing the analysis database.

```
Opening database inductor.op3, simulation number 1 ...

TOSCA Magnetostatic analysis


File: simulation: 1
Created on: 28/May/2012 14:58:43
In Directory:

By Machine:
Node: MyNode. Processor: Intel64/x64. System: Windows.
Log Files: Pre_201205281442299521.log/lp.
Simulation created using: Opera-3d/Pre-processor Ve...

User title:
inductor

Mixed CGS units


1 Biot-Savart conductor (current densities in A cm^2):
1 Solenoid
Current Densities: 1000.0
Adaptive RHS integrals
Drive sets and functions
ONE      :      Coil drive type Constant

Boundary Conditions: NORMMAGN TANGMAGN
```

```
Linear solution
No periodicity conditions have been defined

Materials defined:
CORE
    Isotropic permeability: 300.0
BOX
    Isotropic permeability: 500.0

17425 nodes in the model
50128 edges in the model
Only linear elements exist within the model
15360 linear hexahedra
```

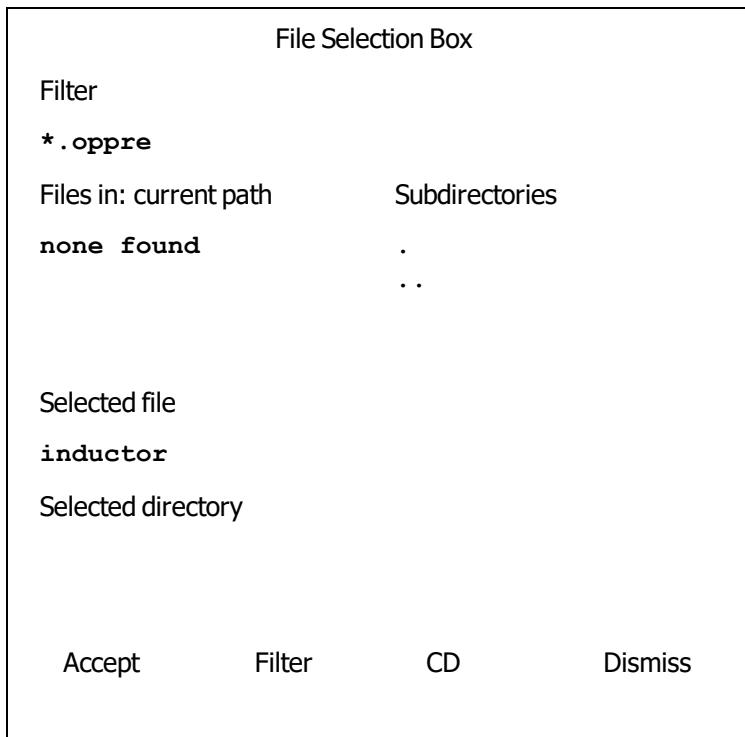


## Writing the Pre-Processor Data File

To save the Pre-Processor commands, select

**FILE -> Write pre-processor file**

This brings up a File Selection Box.



The desired filename can be included (for example **inductor**) in the box labelled "Selected File". Finish by selecting **Accept**.

## Leaving the Pre-Processor

The data has now been stored correctly. Exit the Pre-Processor so that the analysis module may be used. Do this by selecting

**FILE -> End Opera-3d/Pre**

and confirm with **YES**.

## Running TOSCA and Starting the Post-Processor

---

Use the Opera Manager to start the analysis. Right-click on the file **inductor.op3** in the Opera Manager. Select **Solve with TOSCA** from the menu.

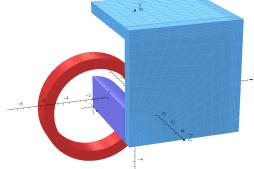
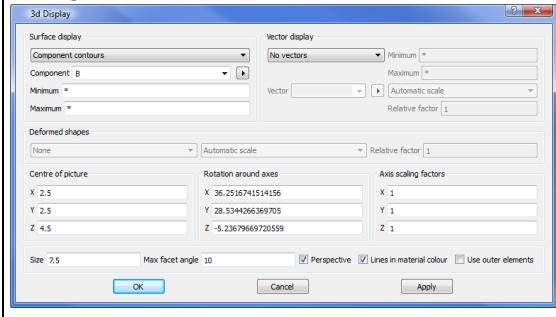
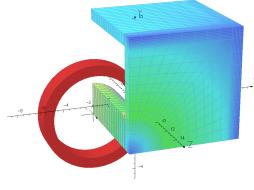
When the analysis has completed, start the Post-Processor by clicking on the **Post-Process** button on the solver window if it has been displayed. Otherwise the Post-Processor can be started from the Opera Manager by double-clicking on the database file, **inductor.op3** or by right-clicking on the analysis in the completed job queue and selecting **View Solution File**.

## Examining the Basic Solution from TOSCA

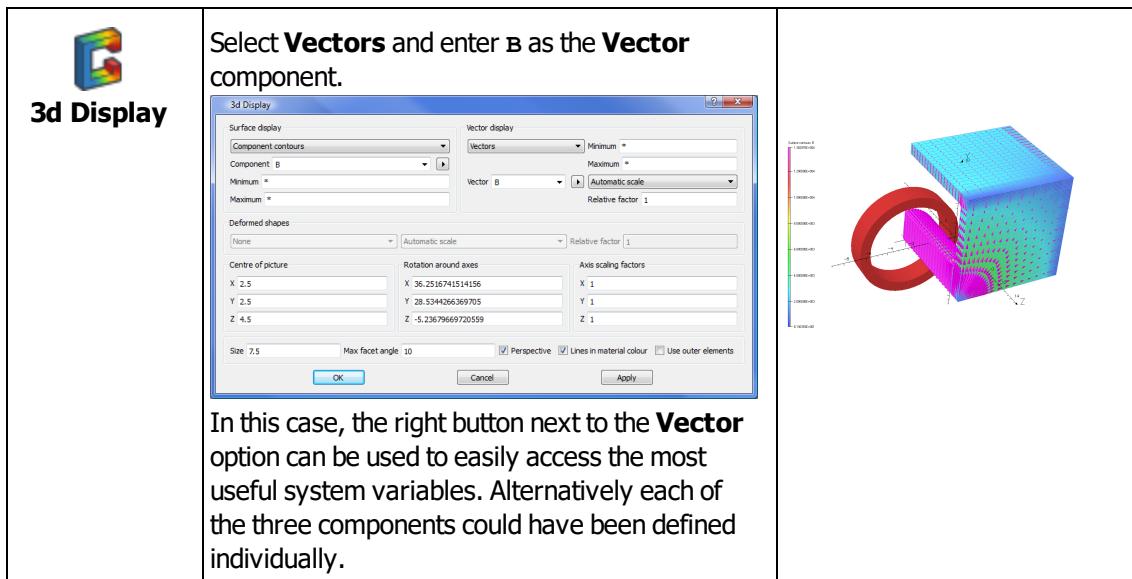
The modelling procedure has made use of the symmetry of the device. The basic solution is without reflection or rotation symmetries shown.

### Viewing the Model

Starting the Post-Processor from the TOSCA solver window or by double-clicking on **inductor.op3** in the Opera Manager, automatically activates and loads the model and displays the geometry. Now a display of the flux density on the surface of the model can be examined.

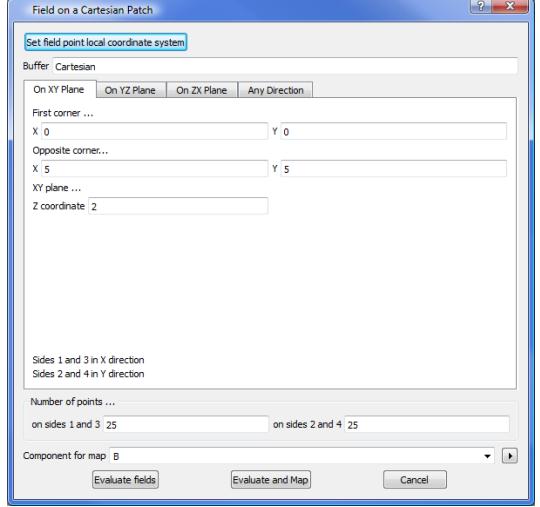
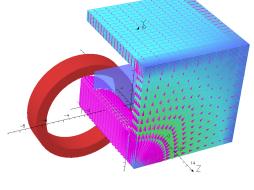
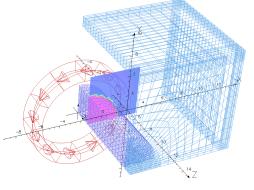
	<p>Start the Post-Processor from the TOSCA solver window or by double-clicking on <b>inductor.op3</b> in the Opera Manager.</p>	
 <b>3d Display</b>	<p>Select the <b>Component contours</b> option for the <b>Surface display</b> and set field <b>Component</b> to <b>B</b>.</p>  <p>Note that the right button on the <b>Component</b> can be used to easily access the most useful system variables</p>	

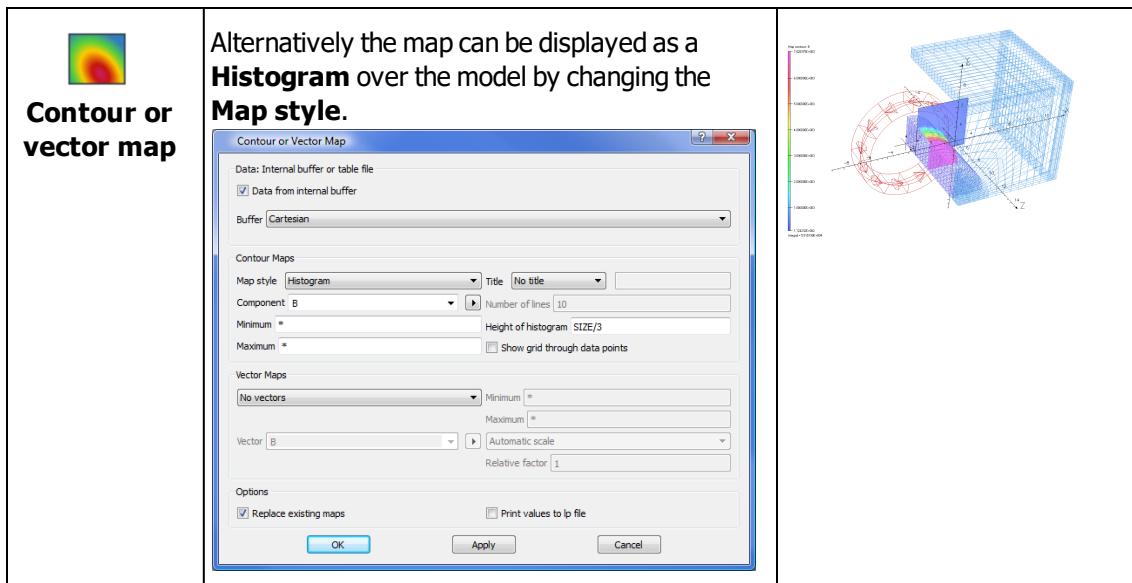
In addition to viewing the field contours it is possible to display field vectors on the surface elements as well.



## 2D Surfaces in the 3D Model Space

To examine a 2D surface in the model, it is possible to specify a surface or "patch".

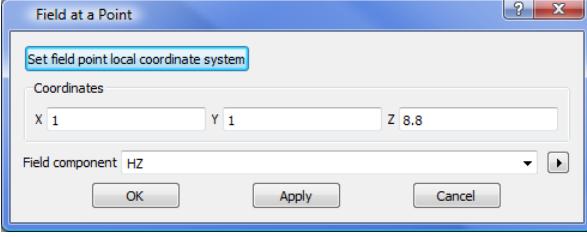
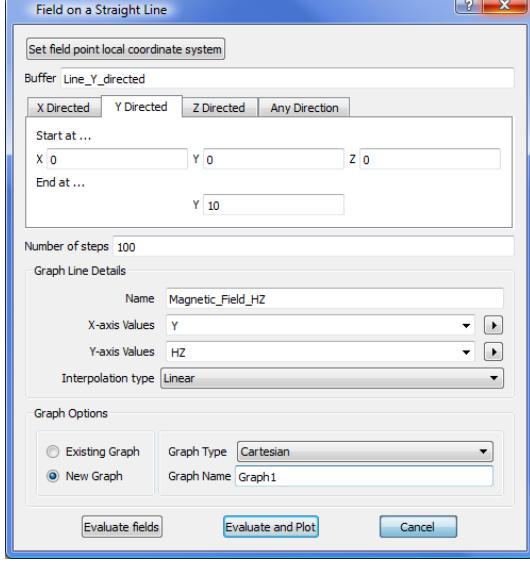
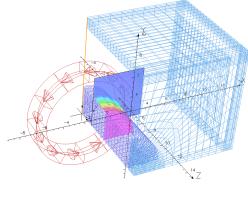
 <b>Fields on a cartesian patch</b>	<p>Select the <b>On XY Plane</b> tab, complete the dialog with corners at (0,0) and (5,5), <b>Z coordinate 2</b> and 25 points on each side of the patch. Select <b>Evaluate and Map</b>.</p> 	
 Toggle the <b>Solid view of model</b>   Toggle the <b>Vectors on surface</b>	<p>This can be better seen by turning off the surface elements in order to see a wire-frame view of the model. Also turn off the display of the vectors.</p>	



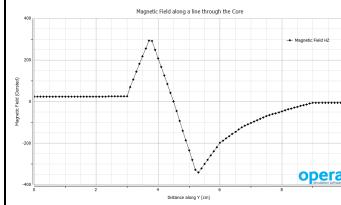
## Evaluation of Solution at Points and Along Lines

It is possible to examine the solution at points and along lines.

To examine the value of the magnetic field in the z-direction at a point and along a line the following actions can be taken.

 <b>Fields at a Point</b>	<p>Find the field strength in the gap between the core and the box. Enter coordinates: 1, 1, 8.8 and <b>Field component HZ</b>.</p>  <p>Results in a field of around 3800 oersted.</p>
 <b>Fields on a straight line</b>	<p>In <b>Buffer</b>, give a name <b>Line_Y_directed</b>. Click on the <b>Y Directed</b> tab. Start at <b>(0, 0, 0)</b> and end at <b>Y = 10</b>. Set the <b>Number of steps</b> = 100.</p> <p>In the <b>Graph Line Details</b> section, set the <b>Name</b> to <b>Magnetic_Field_HZ</b>. For the <b>X-axis Values</b>, type <b>Y</b>, and for the <b>Y-axis Values</b>, type <b>Hz</b>.</p> <p>In the <b>Graph Options</b>, select <b>New Graph</b>, set the <b>Graph Type</b> to <b>Cartesian</b>, and give a <b>Graph Name</b>, for example <b>Graph1</b>.</p>   <p>Click on <b>Evaluate and Plot</b>.</p>

Then the line graph is plotted on the 2d graphics tab as shown on the right.

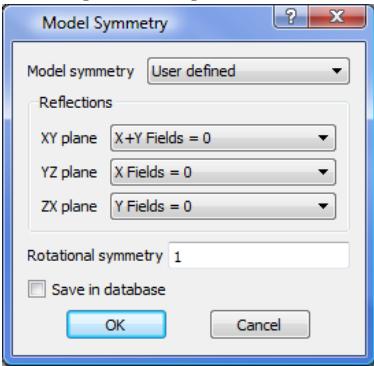
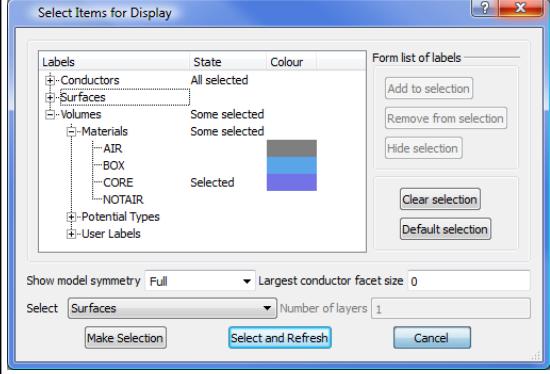


## Examining the Complete Device from TOSCA

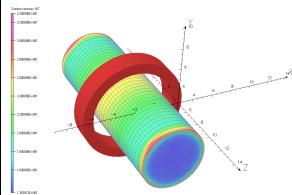
---

### Loading the Model

To examine the results of the complete device, the symmetry should be set.

 <b>Model Symmetry</b>	<p>Make the following choices:</p> <p><b>Model symmetry: User defined</b></p> <p><b>Reflections:</b></p> <p><b>XY plane: X+Y Fields = 0</b></p> <p><b>YZ plane: X Fields = 0</b></p> <p><b>ZX plane: Y Fields = 0</b></p> <p><b>Rotational symmetry: 1</b></p>  <p>If <b>Save in database</b> is selected, this operation need not be repeated if the database is opened again.</p>	
 <b>Select components for display and calculation</b>	<p>Expand options <b>Volumes</b> and <b>Materials</b>, and select the material named <b>BOX</b> and then choose <b>Remove from selection</b> so that only the <b>CORE</b> is selected for display. Set <b>Show model symmetry</b> to <b>Full</b> and run the command with <b>Select and Refresh</b>.</p> 	

Reinstate the solid view of the model  and remove the histogram .



This completes the post-processing and the Post-Processor can be closed by selecting

**File -> Exit**

## ELEKTRA Worked Example

The data prepared in the previous model can also be used as the basis of an ELEKTRA analysis, calculating the eddy currents in the non-air volumes. The ELEKTRA analysis module must be licensed.

If the ELEKTRA analysis module is to be used the eddy current effects must be modelled using total and reduced **VECTOR** potentials.

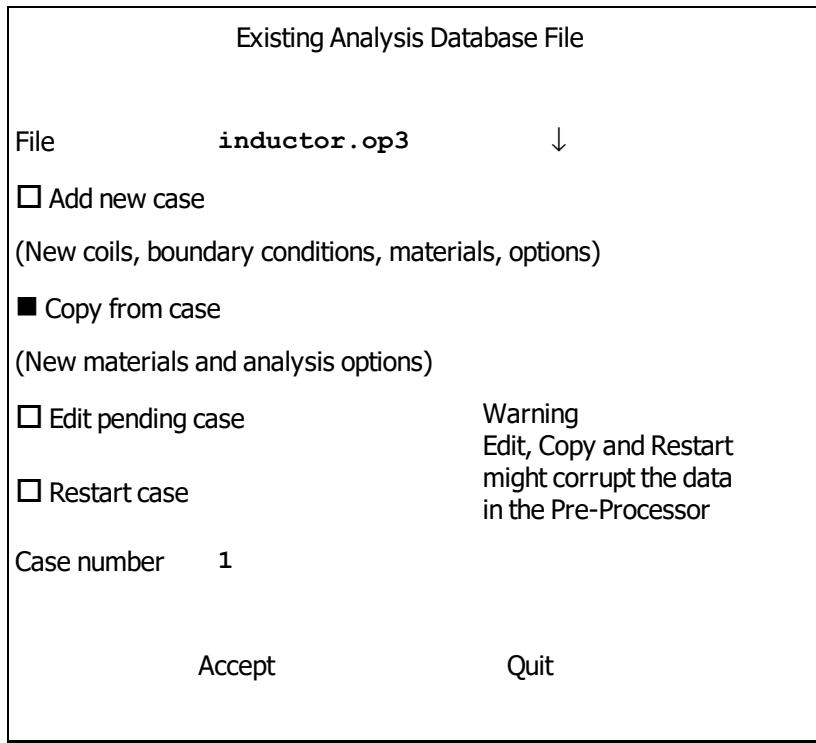
No changes are required to the model and the mesh. In this case it is sufficient to add a new simulation to the existing database.

### Writing the ELEKTRA Database File

Start the Pre-Processor in the usual way and select

**FILE -> ... use existing database**

and under the section **Low Frequency (ELEKTRA)**, select **Steady-state AC**. The following parameter box is then completed, using the previous file name:

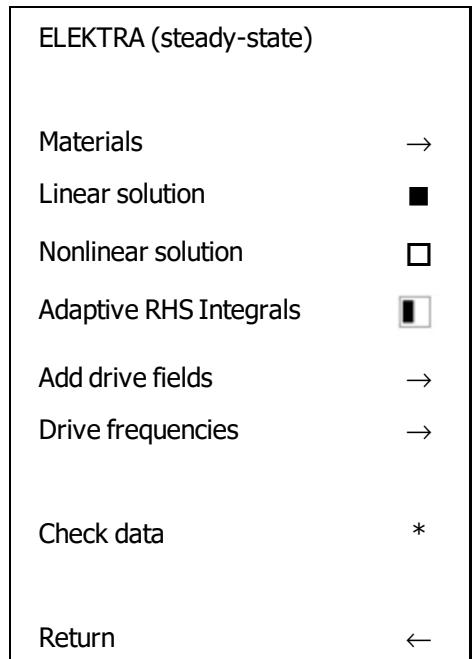


If the box ↓ alongside the file name is selected, a file selection box is raised, allowing all existing files to be shown/selected.

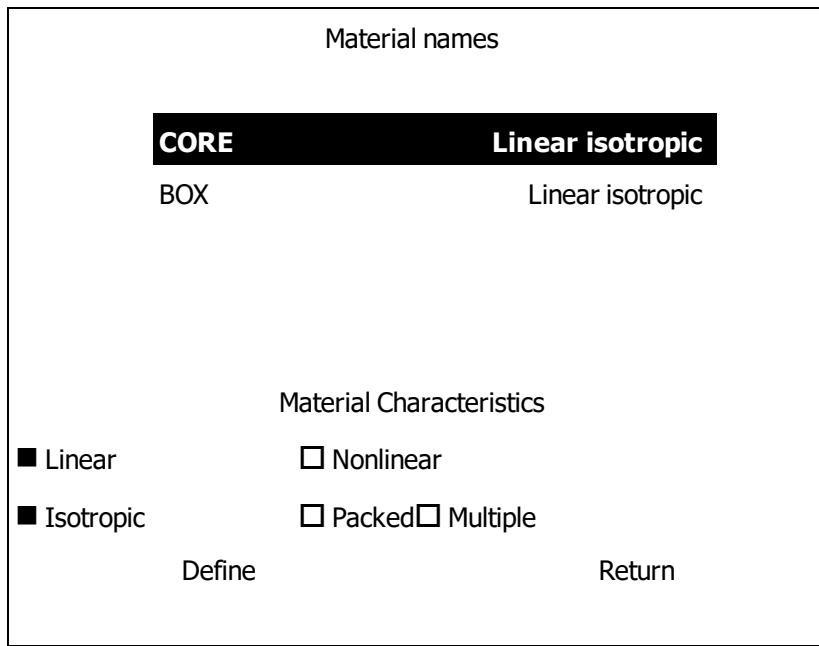
Select **Accept**, clear the message box that appears.

The first action is to give a title for this simulation, terminated with an \* character on a line of its own. A prompt is then given for the drive phase, and drive scale factor. Since this is an ELEKTRA/SS solution, it is possible to set the phase of the drive (assumed to be Cosine). There is only a single drive, so in this case set the phase to zero, and drive scale factor to 1, and **Accept**.

Having cleared the message box, the **ELEKTRA (steady-state)** data menu appears. Now it is necessary to specify the analysis specific quantities.

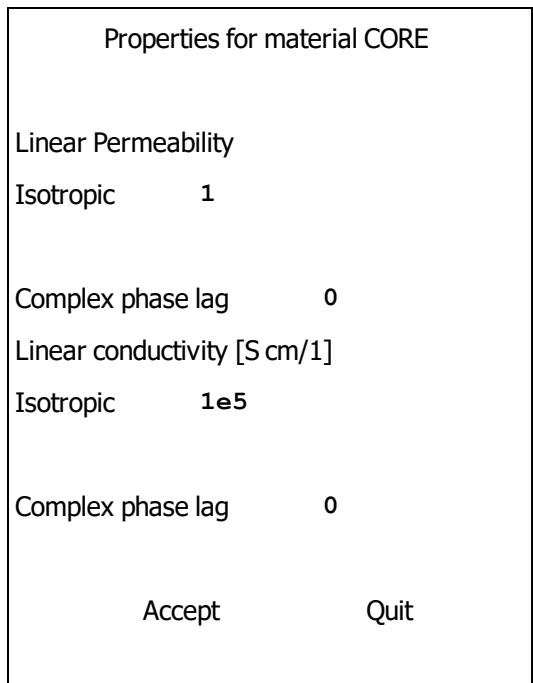


First select **Materials**. For each material in the model, the linear or nonlinear characteristics can be defined. For example, in the present model, the parameter box will appear, and **CORE** is selected as follows:



Selecting **CORE** first, the material properties can be defined. These are to be linear isotropic. Selecting **Define** allows the properties of the material for **CORE** to be defined.

The parameter box that appears should be completed to set an isotropic permeability of 1 (previously set at 300). The isotropic conductivity should be set to 1.0e5 (with zero complex phase lag). Follow this by selecting **Accept**.



The same is done for material **BOX**, by high-lighting **BOX**, and selecting **Define**. Again, the isotropic permeability should be changed to 1 (from 500), with conductivity of 1.0e4, and select **Accept**.

Close the **Material names** box by selecting **Return**.

The frequency of the solution must be given as well. Select **Drive frequencies** and enter 50 for the new frequency. Press **Add** to add this frequency to the case list, and close the menu with **Return**.

Close the **ELEKTRA (steady-state)** data menu with **Return**, and check that the data is correct in the information box. This should be:

```
Opening database inductor.op3, simulation number 2 ...

ELEKTRA Steady-state harmonic analysis


File: simulation: 2
Created on: 28/May/2012 15:36:28
In Directory:

By Machine:
Node: MyNode. Processor: Intel64/x64. System: Windows.
Log Files: Pre_201205281442299521.log/lp.
Simulation created using: Opera-3d Pre-processor Ve...

No user title given.

CGS units

1 Biot-Savart conductor (current densities in A cm^2):
1 Solenoid
Current Densities: 1000.0
Adaptive RHS integrals
Drive sets and functions
ONE : Coil drive type Cosine (Frequency by case, Phase 0

Boundary Conditions: NORMMAGN TANGMAGN

Linear solution
No periodicity conditions have been defined

Materials defined:
AIR
BOX
```

```
Isotropic permeability: 1  
Isotropic conductivity: 10000.0 [S cm/1]  
CORE  
Isotropic permeability: 1  
Isotropic conductivity: 100000.0 [S cm/1]
```

```
1 solution frequency defined: 50.0
```

```
17425 nodes in the model
```

```
50128 edges in the model
```

```
Only linear elements exist within the model
```

```
15360 linear hexahedra
```



Continue

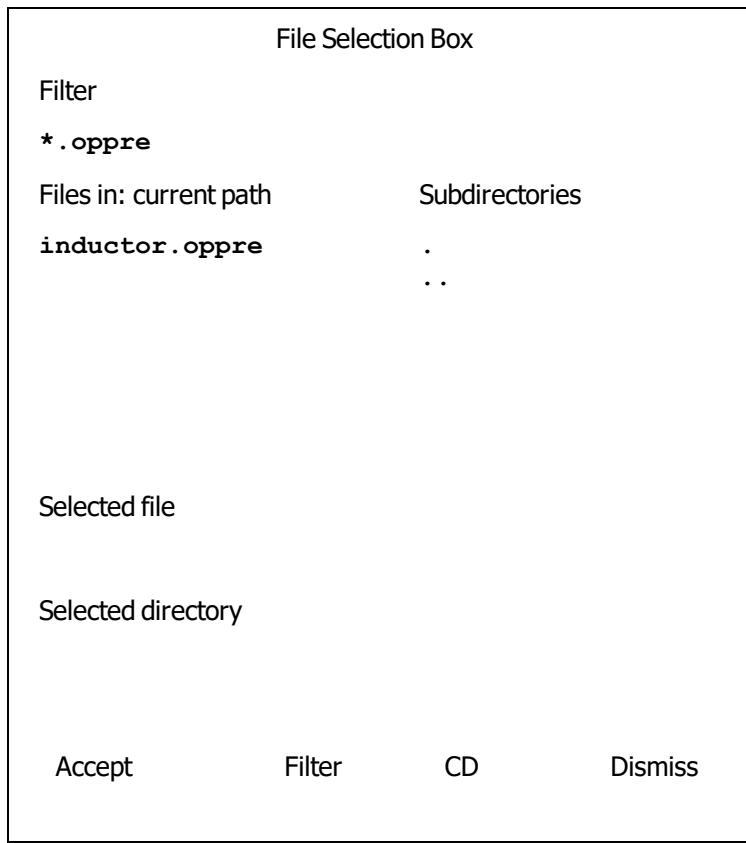
Close the information box, to complete storing the analysis database.

## Writing the Pre-Processor Data File

To save the Pre-Processor commands, select

**FILE -> Write pre-processor file**

This brings up a File Selection Box.



into which the filename can be included. The existing file can be updated by double-clicking this filename, and allowing it to be over-written.

## Leaving the Pre-Processor

The data has now been stored correctly. Exit the Pre-Processor so that the analysis module may be used. Do this by selecting

**FILE -> End Opera-3d/Pre**

and confirm with **YES**.

## Running ELEKTRA and Starting the Post-Processor

---

Use the Opera Manager to start the analysis. Right-click on the file **inductor.op3** in the Opera Manager. Select **Solve with ELEKTRA/SS** from the menu.

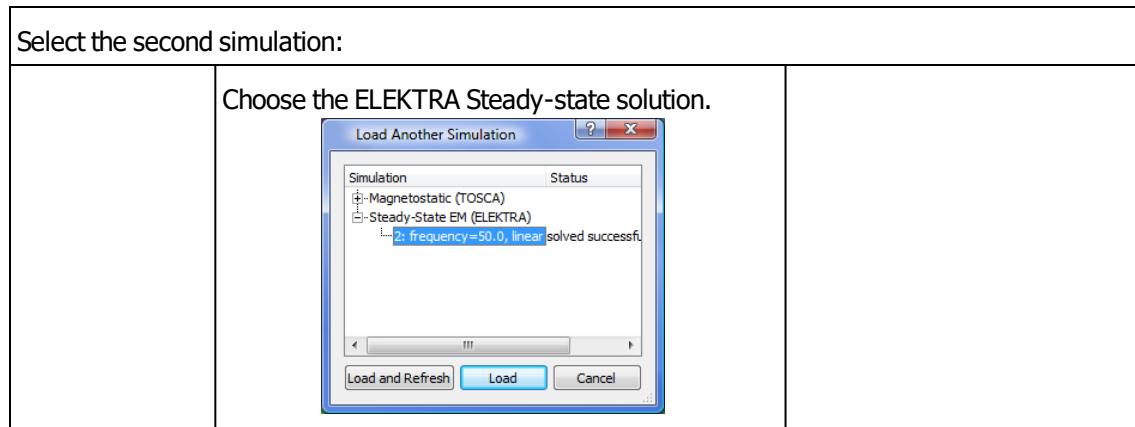
When the analysis is complete, start the Post-Processor by clicking on the **Post-Process** button on the solver window.

## Examining the Basic Solution from ELEKTRA

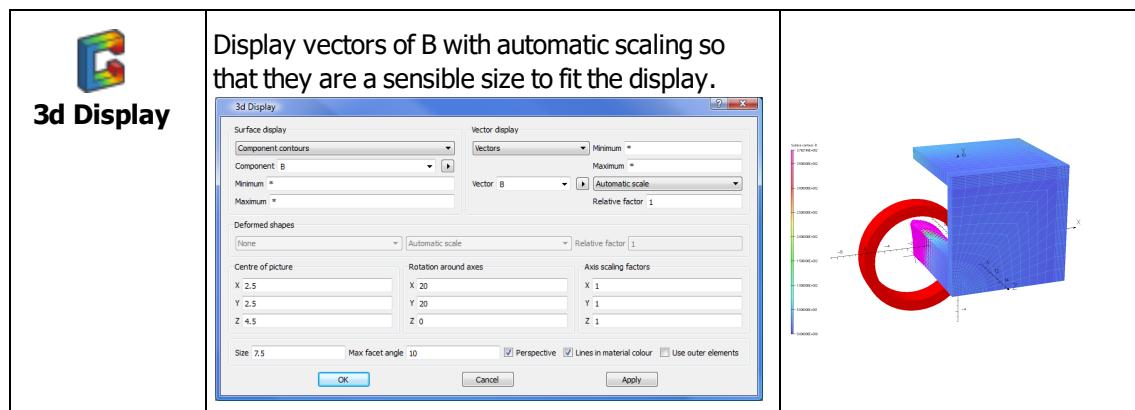
All the commands and techniques used in the post-processing of the TOSCA solution may be used to examine the ELEKTRA solution once the model has been read into the Post-Processor.

### Loading and Displaying the Model in the Post-Processor

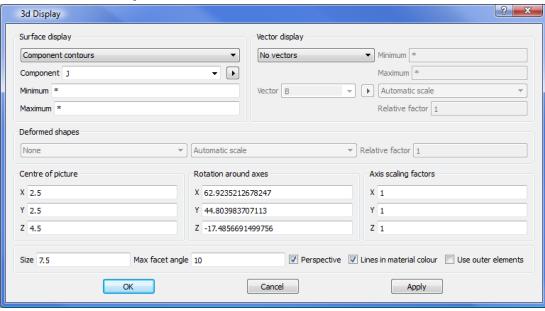
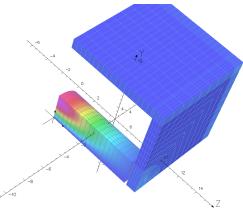
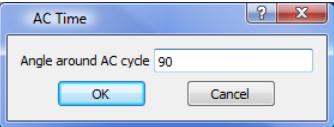
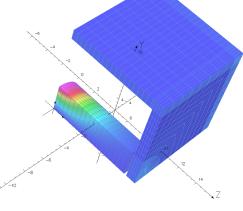
Starting the Post-Processor from the ELEKTRA solver window or by double-clicking on **inductor.op3** in the Opera Manager, automatically activates and loads the model and displays the geometry. Now there are two simulations in the database, simulation 2 should be chosen.



Note that because the permeability of the **CORE** and **BOX** have both now been changed to 1 the magnetic flux density will be much smaller and so the vector display of the flux density will automatically be rescaled in order to see the vectors.



The other post-processing commands that were used for the TOSCA solution can now be used on the ELEKTRA solution. In addition, the eddy current effects can be seen. In this case, however we will now display the model with the conductor hidden from view.

 <b>View conductors</b>	Remove the conductors from the display	
 <b>3d Display</b>	Set the component to: J 	
 <b>Set AC Time</b>	Change the time to 90: 	
 <b>Refresh</b>	Show the currents flowing at 90° around the AC cycle.	

Exit the Post-Processor by selecting

**File -> Exit**

# **Chapter 14**

# **Application Notes**

## **Introduction**

---

This chapter contains a set of Application Notes which are provided to help users with specific types of applications of Opera-3d. The examples given do not contain complete instructions but should be used in conjunction with other sections of the User Guide and the Opera-3d Reference Manual to obtain details of the commands.

The application notes are:

### **Basic Functionality**

- Meshing in the Modeller [page 541]
- Periodicity in the Modeller [page 557]
- Analysis of Parts of a Geometric Model [page 565]
- Using Packing Factor to Represent Laminations [page 570]
- Complex Material Properties [page 572]
- Tabular Functions [page 576]
- Simulation of Magnetization [page 581]
- Using Hysteretic Materials [page 596]
- Boundary Conditions for Thin Gaps [page 603]
- Using Surface Impedance Boundary Conditions in Harmonic Electromagnetic [page 610]
- Using Surface Impedance Boundary Conditions in Transient Electromagnetic [page 613]
- Using Surface Impedance Boundary Conditions in Multiphysics (Electromagnetic and Thermal) Simulations [page 621]
- Thin Plate Approximation for Magnetostatic Simulations [page 625]
- Using Current Source Boundary Conditions [page 635]
- Power and Energy Calculation in Harmonic solutions [page 639]
- Inductance Calculations in Opera-3d [page 641]
- Legendre Polynomials [page 645]

- Circuits in Harmonic and Transient EM simulations [page 651]
- Example Circuits [page 659]
- Bulk Conductors [page 670]
- Restarting Simulations [page 681]

## Electrical Machines

- Machine Analysis using Motional EM solver [page 684]
- Example Control File [page 705]
- Opera in Simulink® System Analysis [page 713]
- Estimating Iron Loss in Opera-3d [page 717]

## Other Applications

- Modelling DC Current flow [page 726]
- Multiphysics Analysis [page 739]
- Combined Magnetic and Electric Fields [page 753]
- Q and g (R/Q) Factors from a Modal HF Solution [page 755]
- Coupled EM and Thermal Analysis in Opera-3d [page 758]
- Particle Collision and Scattering in the Charged Particle solver [page 762]
- Plasma Free Surface - Type 103 Emitter [page 772]
- Plasma Emitter [page 776]
- Parameterized Models in the Pre-Processor [page 785]
- Single Phase Inductor, featuring 3 types of losses [page 787]

## Python Applications

- Supplied Python Functionality [page 791]
  - Database Index [page 793]
  - Database Extraction [page 796]
  - Database Update [page 799]
  - Steady-State Detection [page 801]
  - Loss Calculation [page 806]

# Meshing in the Modeller

## Introduction

The mesh generation in the Modeller is a two stage process.

- Surfaces of volumes (cells) are initially discretized into triangles or quadrangles.
- Starting from the surface mesh, each cell is subdivided into tetrahedra, hexahedra or prismatic elements. This process is called volume meshing.



To prepare the model for meshing the user must first execute the step **Create Model Body**. This selects all entities as bodies, and performs a Boolean **Union, without regularization**. The result is one body with all cells retained.

## Mesh Generator

Two different mesh generators are available. **Mesh Generator Type I** can produce mosaic meshes that consist of tetrahedral, prismatic and hexahedral elements. It can handle all special cases like filamentary and meshed conductors in circuits.

**Mesh Generator Type II** creates tetrahedral meshes in a wide variety of geometries except for the special cases mentioned above, where Mesh Type I has to be used.

When the mesh generator is set to **Automatic Choice**, generator Type II will be the default choice, except for the special cases described above.

## Surface mesh

When generating the surface mesh, a choice has to be made which mesh generator type is to be used. The software issues a warning when the user requests a combination that is not possible, for example filamentary conductors in mesh Type II. In this case the software reverts automatically to Type I.

## ACIS surface mesh

The ACIS kernel generates an initial surface mesh which is normally hidden from the user. This mesh is used as the basis for the normal surface mesh. If there are problems in creating either the surface or volume mesh, it is advised to first check the quality of the ACIS surface mesh. This can be viewed for Mesh Type I only, by setting a negative **Maximum angle between elements** in the surface mesh generator.

## Volume mesh

The volume mesh is based on the surface mesh. Sometimes the volume mesh generator can fail. In most of these cases the surface meshes of adjacent faces are very different so that the volume mesh generator cannot connect up the surfaces. Later in this Application Note it will be discussed how to prevent volume meshing errors.

## Element Types

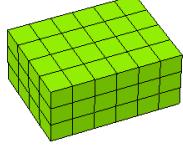
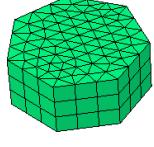
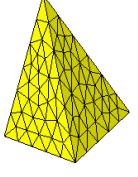
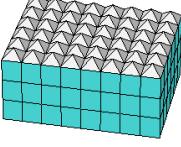
A finite element mesh created with the command **MESH GENERATOR=ONE** (referred to in the previous section as Mesh Type I) can consist of 4 different element types. **Table 141 below** gives an overview. Mosaic meshing allows a mixture of hexahedra, prisms and tetrahedra, with pyramids forming an interface between elements with quadrilateral facets and elements with triangular facets. Pyramids are created automatically without user interaction as the first layer of elements in a cell which is otherwise meshed with tetrahedra.

When creating the surface mesh with the **MESH** command, a preferred mesh type has to be selected using the **TYPE** parameter. There is a choice of 4 options: **MOSAIC**, **PREFERMOSAIC**, **PREFERTETRA**, **TETRAHEDRAL**. In summary:

- **TETRAHEDRAL**: use tetrahedral meshing in all cells;
- **PREFERTETRA**: use tetrahedral unless a cell has its preference set to **HEXOPRISM** and that preference can be satisfied;
- **PREFERMOSAIC**: use hexahedra or prisms wherever possible unless a cell has its preference set to **TETRAHEDRA**;
- **MOSAIC**: use hexahedra or prisms wherever possible.

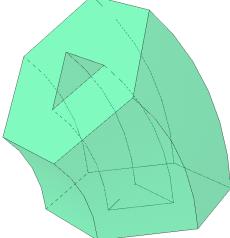
For a detailed description of the options and the implications, please see the ***Opera-3d Reference Manual***, "The **MESH** Command".

**Table 141: Mesh Element Types**

			
Hexahedra	Prisms	Tetrahedra	Pyramids

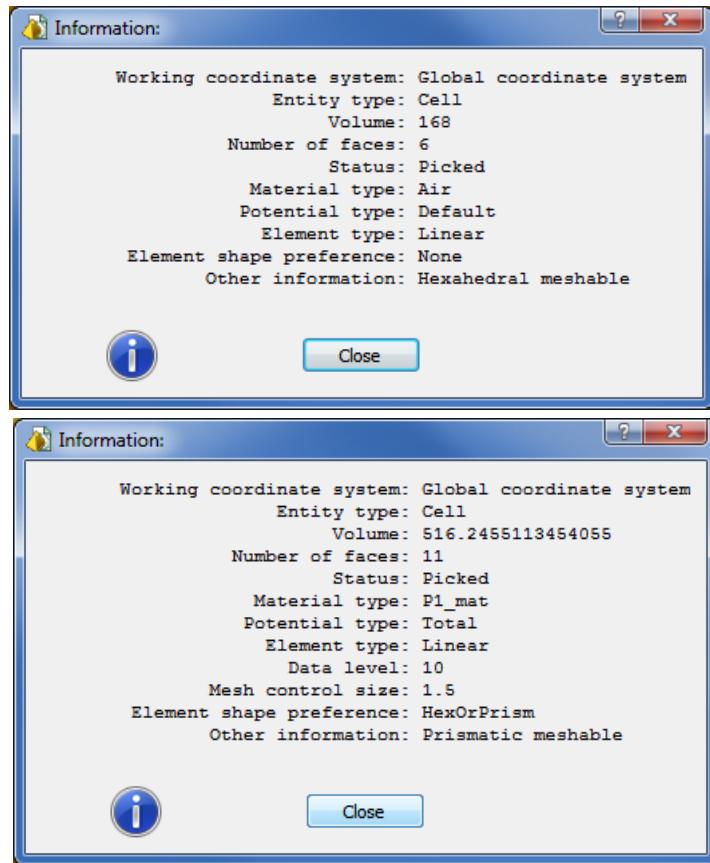
## Cell topology

Any cell can be meshed with tetrahedra elements. To qualify for hexahedral or prism elements, the topology of a cell has to meet the following requirements.

	Criteria for a hex-meshable cell: 6 quadrilateral faces, 8 vertices, and 12 edges.
	Criteria for a prism-meshable cell: one pair of faces with the same topology, connected by quadrilateral faces.

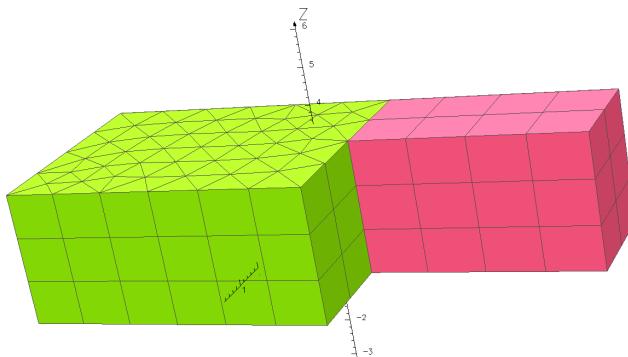
At each stage of building a geometry, the **List Properties** information box for a picked cell will tell the user whether the cell meets the criteria for hexahedral elements (strongest), or for prism elements (second strongest), see [Figure 14.1](#).

For cells which cannot be meshed with hexahedra or prisms this additional information is not included when displaying the cell properties.



*Figure 14.1 Cell information windows, showing whether a cell qualifies for hexahedral or prism elements*

Note that when creating the model body, adjacent cells can add additional edges to a cell, downgrading a hexahedral meshable cell to a prismatic meshable cell (see [Figure 14.2](#)), or even to a cell which can only be meshed with tetrahedral elements.



*Figure 14.2 As a result of creating the model body, an additional edge has been added to the larger block, forcing a prismatic mesh*

## Mesh Control

### Mesh parameters

Parameters to control the mesh size can be set globally in the **Generate Surface Mesh** dialog, or locally on faces and cells. The **Maximum element size** can also be set on edges and vertices. There will often be cases where different element sizes apply to an entity, e.g the global element size, different sizes on adjacent cells with shared faces and edges, etc. When this happens the smallest of the element sizes is used.

Element size means element edge length, which is controlled in 2 ways:

- **Mesh size**

The element size can be controlled by properties of vertices, edges, faces or cells. This allows the mesh to be refined in areas of interest where high accuracy is required or where the field is changing rapidly.

The most commonly used method is to set the **Maximum element size** (**SIZE** parameter of the **MESH** command) to a suitable number.

- **Curved faces**

When a curved surface is to be meshed with a set of planar facets, the element size can be controlled with additional parameters.

- The **Maximum angle between elements** defines the maximum angle between the normal vectors to adjacent faces on a curved surface (**NORMALTOL** parameter of the **MESH** command).
- The **Maximum deviation from surface** defines the maximum distance between the centroid of the planar surface element and the real curved surface it represents (**SURFACETOL** parameter of the **MESH** command).

In cells with a tetrahedral mesh, applying an element size to a face or edge and a smaller size to one of its vertices will lead smaller elements near to that vertex. Most of the face or edge will use the larger mesh size.

In cells with a mosaic mesh, applying a small element size on an edge or on a face can have quite significant implications on mesh sizes in other cells of a model. This will be discussed later in [Mesh-Size Transition \[page 548\]](#).

Setting the mesh size on a vertex only effects neighbouring triangular facets (not 4-sided facets).

Entities with a particular element size can be identified using the **Pick Entities by Property** dialog.

## Layering

Layering can help improve the quality of the mesh generated in thin layers, and makes the modelling of thin surface regions easier. Layering affects the distribution of elements adjacent to a face. Two methods are available: geometric and mesh.

### Geometric layering

In geometric layering, the layers are created during **Create Model Body**. The layers form additional cells with the surface mesh of the original face copied onto the layers. The method should only be used where necessary, because the additional operations when creating the model body will increase the time and complexity of operations needed to form the model body.

Geometric layering can be used with any element type.

### Mesh layering

The mesh layering method can be applied to any face of a cell which can be meshed using hexahedra or to the "end" faces of a cell which can be meshed with prisms.

The layers are created during mesh generation. With this method, the mesh generator forms layers of elements without introducing additional cells or faces.

### Layering data

The parameters for layering are specified in the **FACEDATA** command. First it has to be explained that a face has a forward and backward sense with respect to the surface normal. The surface normal can be viewed as shown in [Figure 14.3](#), and if needed, the direction can be changed (by a double-click on the vectors).

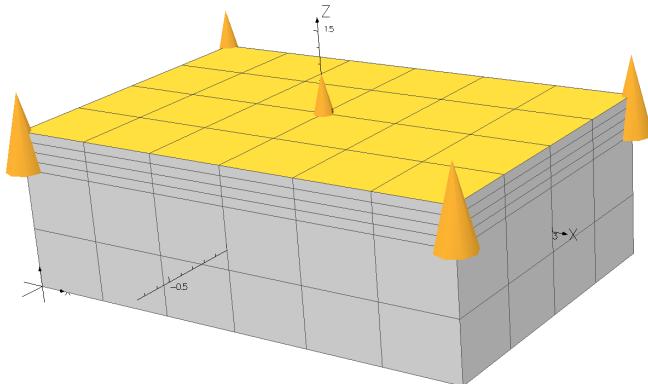


Figure 14.3 Backlayering of the selected face

A face can be layered in both the forward and backward direction simultaneously (but only if the same layering method is used). The example in Figure 14.3 shows a simple block with a thickness of 1 length units. The selected surface has been given a face property of 4 backward layers, with a **BACKOFFSET** of 0 . 06.

The layering information for a face is defined by setting the following parameters:

- **FORLAYER**s defines the number of layers in the forward direction from the starting face.
- **FOROFFSET** defines the thickness of each forward layer - a functional offset can be specified.
- **BACKLAYER**s defines the number of layers in the backwards direction.
- **BACKOFFSET** defines the thickness of each backward layer - a functional offset can be applied.

## Functional layering

Functional layering offsets can be defined. The distance between each subsequent layer can be defined as an expression involving the variable **LAYER**. For example, the expression **2\*layer** will make the thickness of the first layer 2 length units, and the second layer 4, etc.

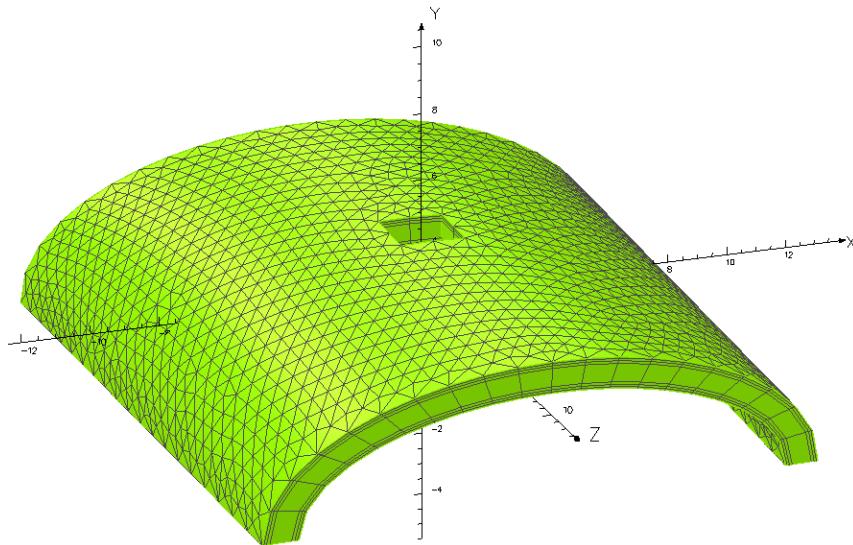


Figure 14.4 Prismatic mesh with mesh layering in an elliptical, conical shell with a hole

## Mesh-Size Transition

In a model, which qualifies for a mosaic mesh, a modification to the mesh size of one cell may have a huge implication on the mesh sizes in other cells of the model. For example, the mesh size on a face is changed from 4 to 1 length units. [Figure 14.5](#) shows the model with a general mesh size of 4 length units, and [Figure 14.6](#) shows the mesh after reducing the mesh size to 1 length unit on the selected face.

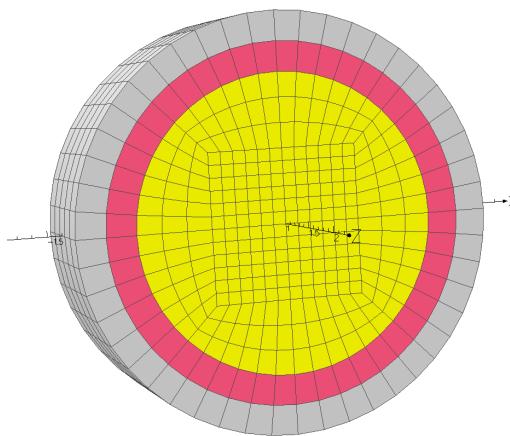


Figure 14.5 Hexahedral mesh with an element size of 4

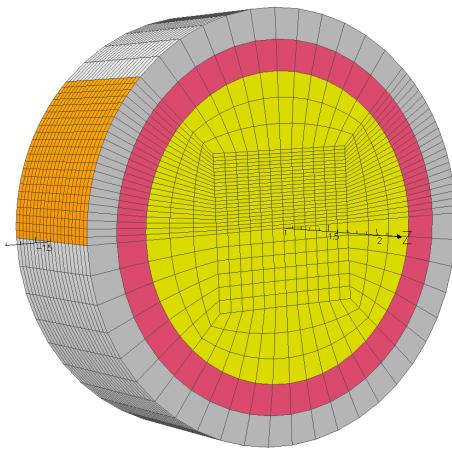


Figure 14.6 The mesh size on the marked face has been changed to 1

By picking a face as shown in Figure 14.6, the fine mesh has propagated in 2 directions: not only from left to right, but also in the axial direction of the pipe structure. Note that by picking an edge on the outer radius of the mesh, the fine mesh would have propagated only from left to right.

### How to stop transition

By introducing an additional cell between two hex-meshable cells, and forcing the mesh to be tetrahedral in this cell, the propagation of small mesh sizes can be stopped, as shown in Figure 14.7.

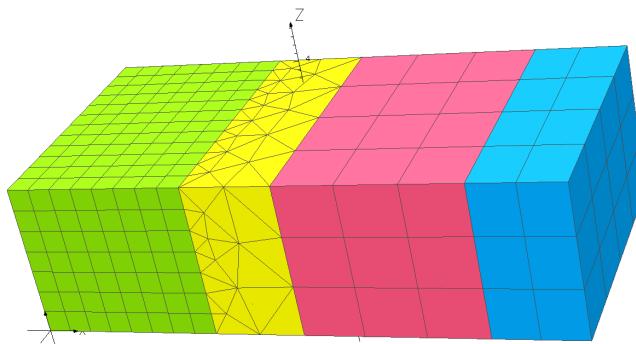


Figure 14.7 By forcing a tetrahedral mesh in the yellow block, the propagation of small element sizes can be stopped

Figure 14.8 shows the mesh around a dielectric rod in a waveguide. Here a cylindrical shell has been used to stop the transition of a fine mesh in the rod.

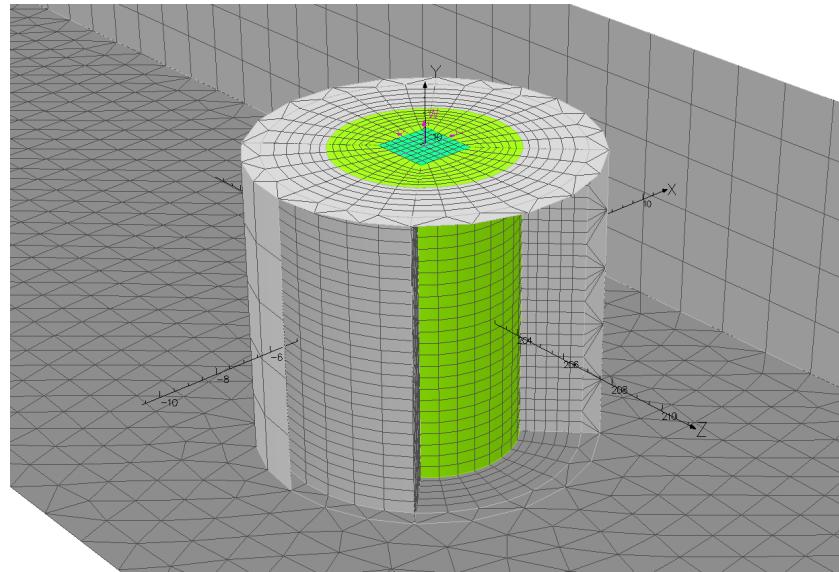


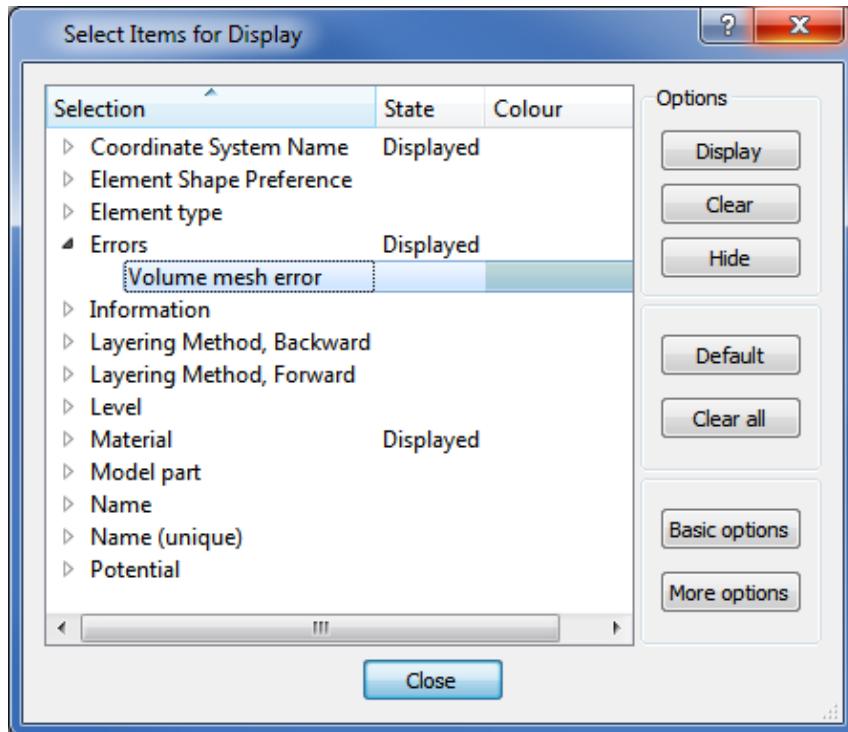
Figure 14.8 A fine mesh in the rod connects to a coarser mesh in the waveguide

## Identifying Errors

### Mesh error labels

When the surface mesh generator stops with an error message, the face which has failed to mesh is assigned an additional label **Surface mesh error**. This label can be selected for display. In most cases it is obvious why the surface mesh generator has failed, for example if the face has a complicated shape or an extreme aspect ratio.

Similarly in the volume mesh generator, an error will assign the label **Volume mesh error** to the volume that caused the error. Initially this volume will be displayed in blue colour. The volume can also be selected for display:



## Meshing - Typical Problems

### Background too complicated

A common cause of meshing problems is when the background region is too complicated, and the mesh generator cannot connect surfaces that are close together and have very different surface meshes. What is often required is to add extra regions to connect the problem surfaces together. This ensures a better surface mesh which the volume mesh generator can then use.

This is done by including some air regions in the model to connect small model components and leaving the background region to mesh the larger parts around the device.

Essentially, do not build a model that requires the background region to fill thin gaps between materials. Instead insert additional air regions to model thin gaps.

### Smaller volumes mesh faster

It is recommended to build a model using a larger number of small volumes rather than a smaller number of large volumes. The reasons are:

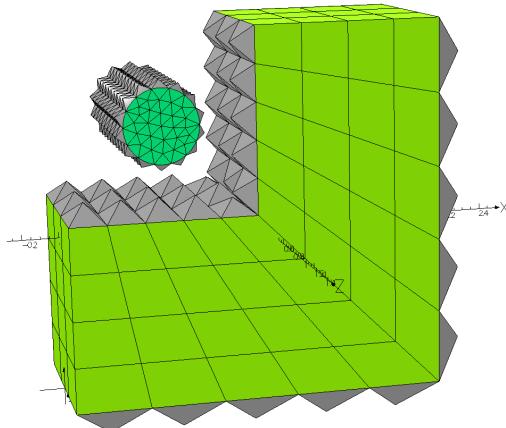
- The volume mesh generator creates the mesh for each separate volume one by one. In a large volume there are more degrees of freedom to generate a mesh, whereas small volumes restrict the degrees of freedom significantly.

- Comparing two models, both with the same number of elements but with a different number of cells, it is found that the model containing a larger number of smaller cells will mesh faster. Smaller volumes are easier to mesh, resulting in:
  - fewer mesh generator errors,
  - shorter times to mesh the whole model.

Large cells can be cut into smaller volumes as described later in [Cutting sheets \[page 554\]](#).

## Pyramid elements on inside corners

When using mosaic meshes, i.e. a combination of cells meshed with tetrahedral and hexahedral elements, the pyramid elements on the interface may cause problems on inside corners. [Figure 14.9](#) illustrates this situation.



*Figure 14.9 Pyramid elements may intersect at inside corners*

It is recommended to avoid inside corners with less than 90 degrees in this case, or to try filling the inner region with a cell which qualifies for a prism mesh.

## Fixing a Meshing Error

Once the cell which refuses to mesh is identified, measures can be taken to solve the problem. One method can be to change the cell properties and adjust the maximum element size.

If a problem persists, a copy of the cell can be made which can be placed into a separate file. This new file can be loaded into a new instance of the Modeller and different combinations of **Target maximum mesh element size** and **Maximum angle between elements** can be tried until the cell can be successfully meshed. Using this method cell after cell can be fixed until the whole model meshes.

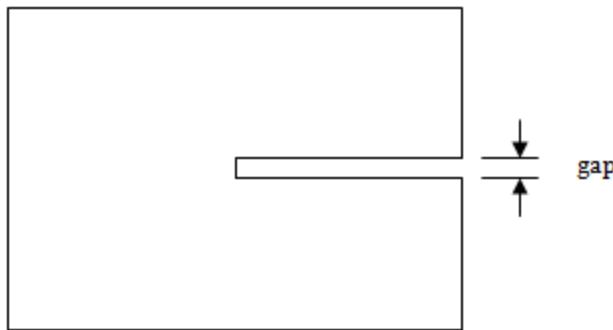
The order in which cells are meshed is based on the order the cells have been created, except that cells which are picked will be meshed before other cells. Following a mesh error, a subsequent volume mesh will start with the cell which failed last time.

## Guidelines to Improve Mesh Generation

In this section some guidelines for reliable meshing are given.

### Simplify the shape of cells

A typical configuration that can lead to volume meshing problems with tetrahedral meshes is shown in [Figure 14.10](#).



*Figure 14.10 Section through a U-shaped region with a thin gap*

The figure shows a section through a region that is cut by a thin plate (thickness = gap). The Modeler's tetrahedral mesh generator becomes unreliable when the gap dimension is much smaller than the size of the region it is cut into. [Figure 14.11](#) shows how to avoid this problem:

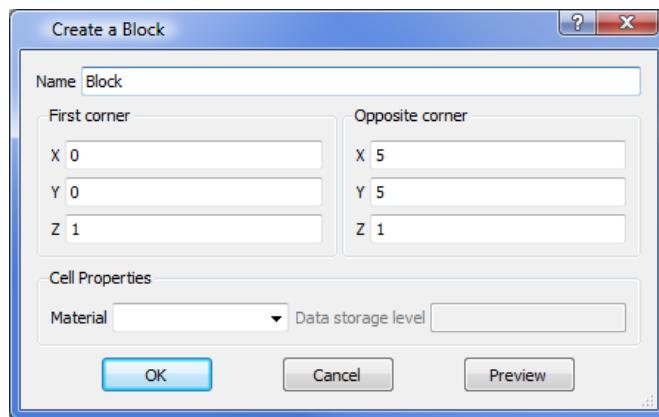


*Figure 14.11 Region cut into two parts*

Cutting the region into two regions makes the mesh generator much less sensitive to rounding errors during the calculation of a tetrahedral mesh. There are various methods of achieving this goal. For example the two regions can be built separately using two entities, or a cutting sheet can be used to subdivide the U-shaped region.

## Cutting sheets

In order to have a better control over the meshing, especially volume meshing, it can be useful to cut volumes into smaller regions using a cutting sheet with zero thickness. Such a cutting sheet can be a block with zero thickness for example as defined in [Figure 14.12](#).



*Figure 14.12 A block with zero thickness in one direction creates a thin sheet*

[Figure 14.13](#) shows an example with three cutting planes - one in each direction.

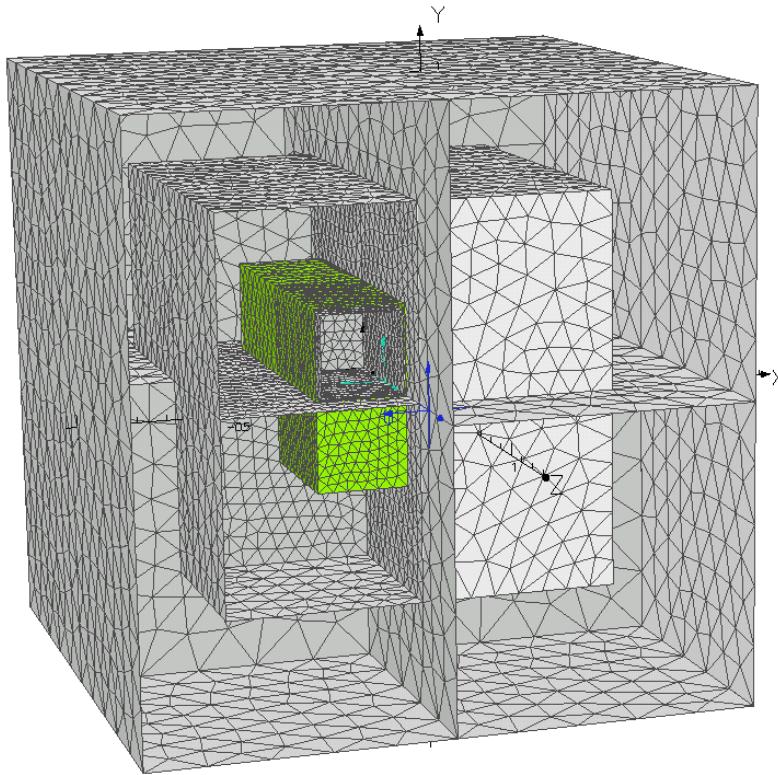


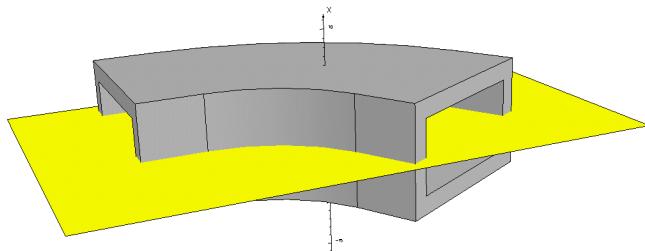
Figure 14.13 Model space cut into smaller cells using 3 cutting planes

Rather than using a block as shown above, any planar **face** of a **cell** can be selected and copied. The copy process results in a new **body**, which then can be used for further Boolean operations.

The three cutting planes shown in Figure 14.13 can be created with a **comi**-script just before creating the model body. No further Boolean operations have to be done and no material properties or mesh control sizes have to be specified.

```
/ cutting.comi
/ Generate 3 cutting sheets to improve volume meshing
/
BLOCK Name=CuttingSheet X0=-1 Y0=-1 Z0=0 X1=1 Y1=1 Z1=0
PICK OPTION=ADD PROPERTY=Name LABEL='CuttingSheet'
/
TRANSFORM OPTION=COPY KEEP=YES TYPE=ROTATE COUNT=1 ROTU=0 ROTV=1,
TRANSFORM ROTW=0 ANGLE=90
TRANSFORM OPTION=COPY KEEP=NO TYPE=ROTATE COUNT=1 ROTU=1 ROTV=0,
TRANSFORM ROTW=0 ANGLE=90
/
MODEL CREATE
MESH SIZE=1 NORMALTOL=20 SURFACETOL=0 TOLERANCE=1.0E-06
FILL TOL=1.0E-06
/
/ end of comi-file
```

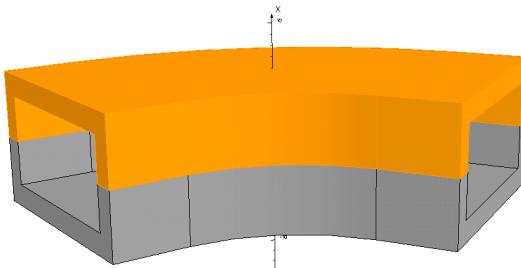
In case the outer boundaries of a volume to be cut are not planar, a Boolean subtraction between the volume and the cutting sheet can be done, see [Figure 14.14](#). A command script to generate this model is given below.



*Figure 14.14 An iron core is to be cut into 2 cells*

```
/ Kicker_Bend_Cut.comi
/
BLOCK Name=IRON X0=-6 Y0=-10 Z0=-20 X1=6 Y1=10 Z1=20
BLOCK Name=CUTOUT X0=-4 Y0=-8 Z0=-20 X1=4 Y1=8 Z1=20
/
PICK OPTION=ADD PROPERTY=Name LABEL='IRON'
PICK OPTION=ADD PROPERTY=Name LABEL='CUTOUT'
COMBINE OPERATION=SUBTRACT +REGULAR
PICK OPTION=ADD PROPERTY=Name LABEL='IRON'
BEND LCSNAME='Global coordinate system',
BEND RADIUS=25 ANGLE=60 CENTRE=YES
/
BLOCK Name=cutting X0=-30 Y0=-12 Z0=0 X1=30 Y1=25 Z1=0
PICK OPTION=ADD PROPERTY=Name LABEL='cutting'
TRANSFORM OPTION=APPLY TYPE=ROTATE ROTU=0 ROTV=1 ROTW=0 ANGLE=90
PICK OPTION=ADD PROPERTY=Name LABEL='IRON'
PICK OPTION=ADD PROPERTY=Name LABEL='cutting'
COMBINE OPERATION=SUBTRACT +REGULAR
/ end of comi-file
```

The Boolean operation in the script above is a **Subtraction with regularization**. The volume to be cut has to be selected first, then the cutting sheet. The Boolean operation leaves the volume called **IRON** cut into two cells with the cutting sheet removed. [Figure 14.15](#) shows the result.



*Figure 14.15 The body cut into 2 cells*

## Periodicity in the Modeller

In many cases it is possible to solve only a part of a model and the symmetry in the field can be ensured by applying either normal or tangential boundary conditions (that is the field is either normal or tangential to the plane of symmetry).

However there are some systems where the combination of fields makes it difficult or impossible to apply such a boundary condition (for example in rotating machines where there are planes of symmetry, but the field is not normal or tangential to those planes). In this case it is possible to use a periodicity boundary condition. This matches the fields on one periodic boundary to the field on another corresponding periodic boundary.

The following simple example shows a quarter of a full model.

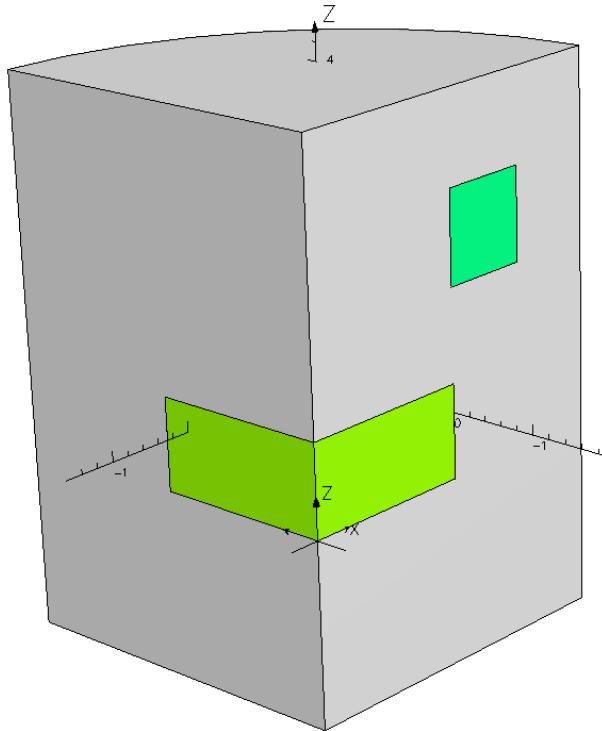
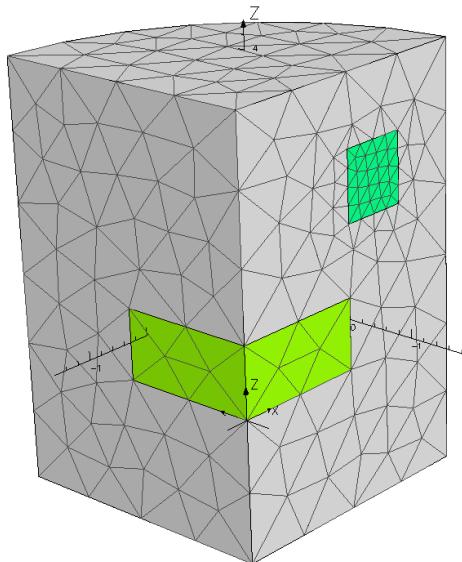


Figure 14.16 Quarter model used in demonstration of periodicity

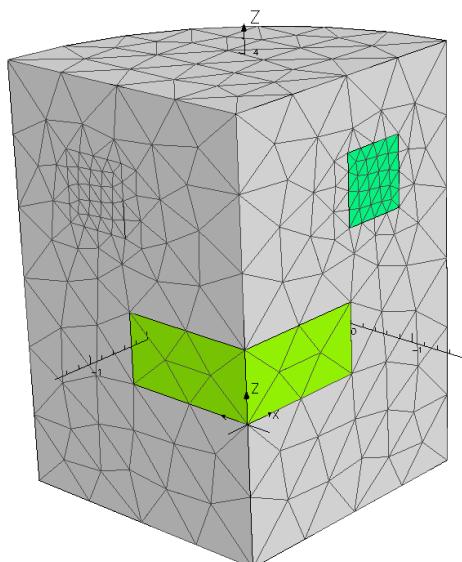
The periodicity option requires the same mesh on the faces to be paired. In case of geometric features on one face not appearing on the other face (as in the example above) the mesh will be copied over without changing the material properties.

## Viewing the mesh

When the mesh is created, all nodes on one face will be matched to all nodes on the corresponding face. [Figure 14.17](#) and [Figure 14.18](#) show the meshes created with and without periodicity applied



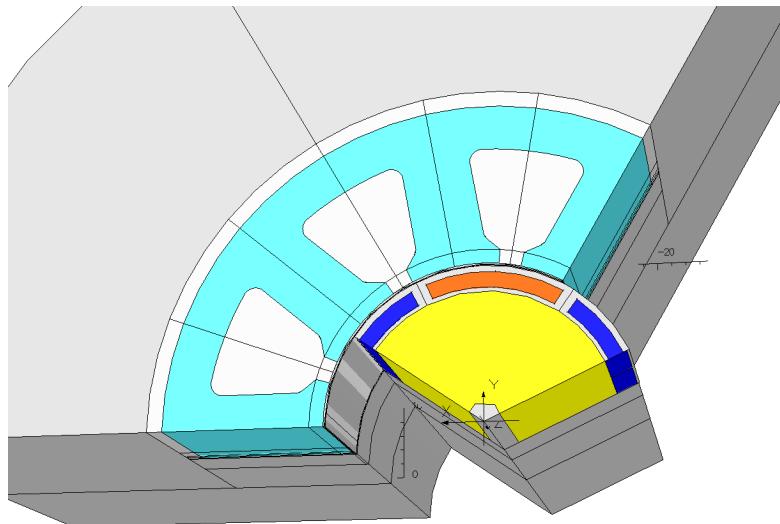
*Figure 14.17 Different surface mesh on each face with no periodicity set*



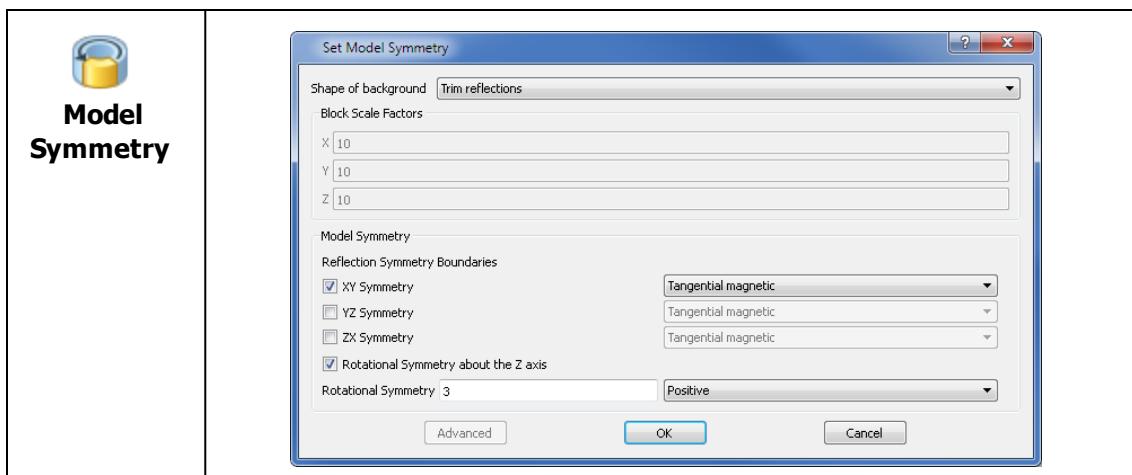
*Figure 14.18 Identical surface mesh on the symmetry planes after periodicity has been applied*

## Rotational Symmetry around Z

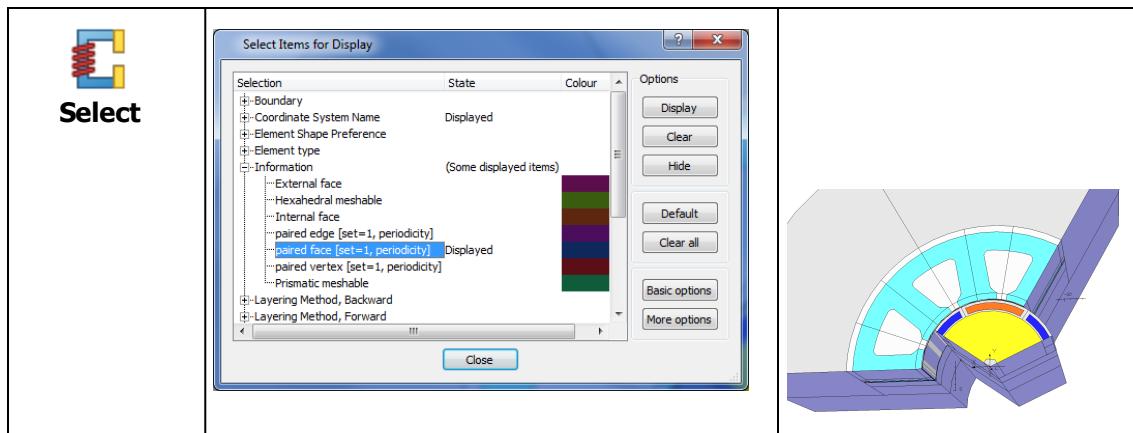
There is an easy way of specifying the rotational symmetry of an electrical machine. For a 120 degrees section as shown in [Figure 14.19](#), a positive symmetry of 3 should be defined.



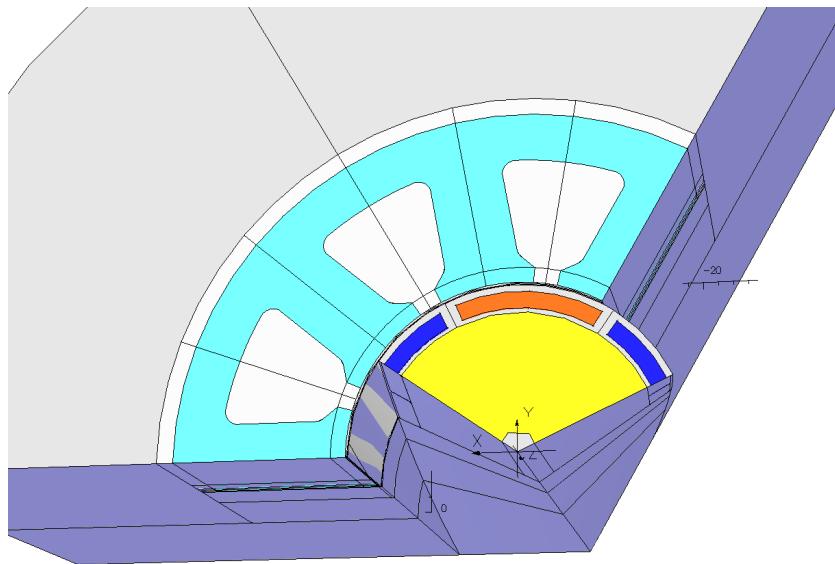
*Figure 14.19 Model of a permanent magnet motor prepared for a magnetostatic analysis*



Having created the Model Body, there are new **information** labels which can be used to show the **paired faces**.



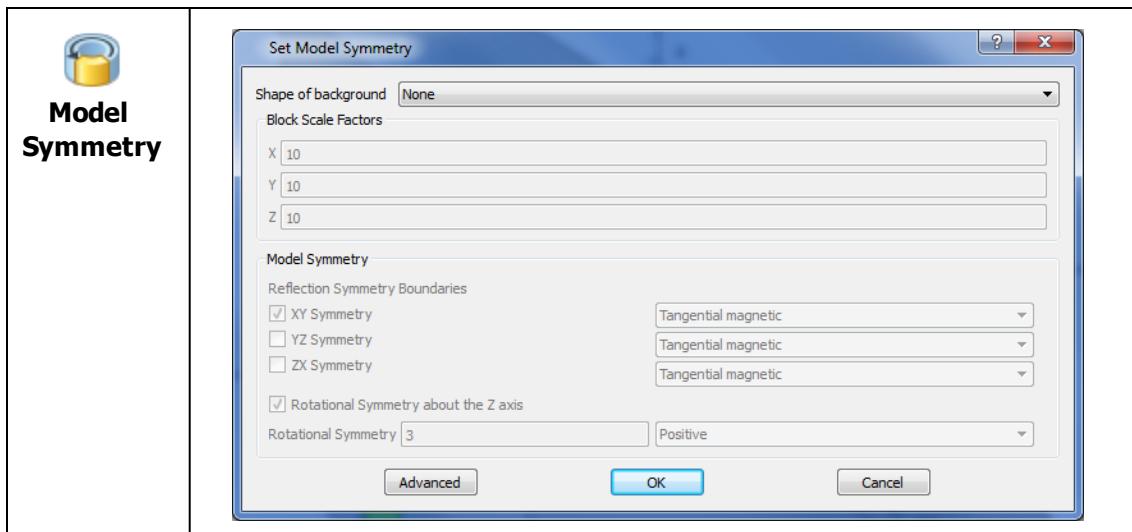
Using the Trim Reflection option, even the faces of a skewed rotor can be paired as shown in [Figure 14.20](#).



*Figure 14.20 Paired symmetry faces with a skewed rotor*

## Advanced Methods for Periodicity

For more complex geometries, advanced options for periodicity are available. Set the **Shape of background** to **None**, then press the **Advanced** button at the bottom of the dialog.



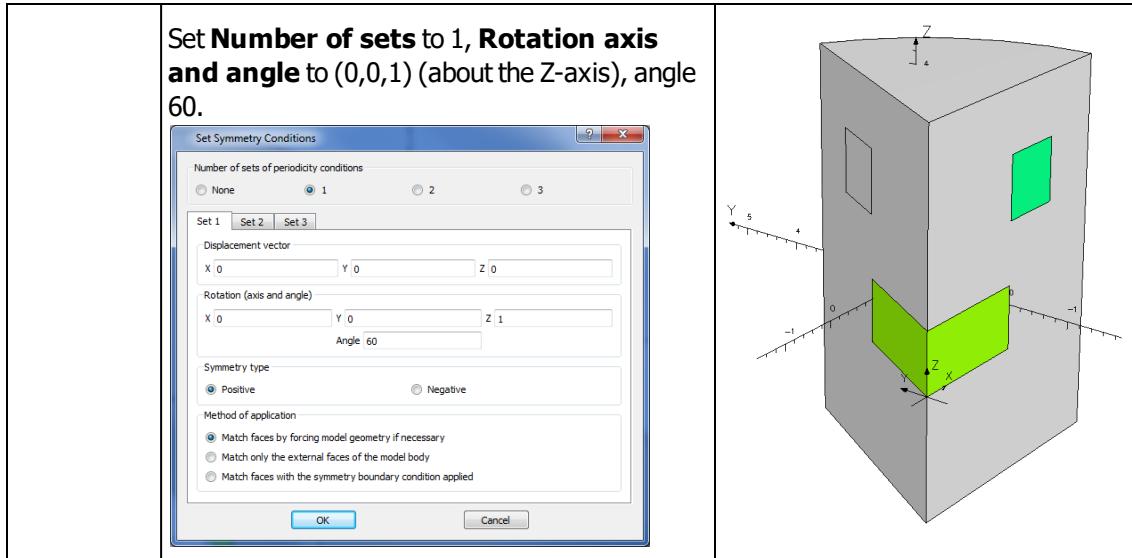
Beside rotational symmetry, a displacement vector for linear periodicity can be defined. Up to 3 sets of periodicity boundary conditions can be given.

For each, three different methods of application can be used:

1. Forcing model geometry
2. External faces
3. Match labelled faces

### Forcing model geometry

The first method is the most powerful and the easiest to use, as no faces have to be labelled. It can actually modify the geometry in order to accommodate the symmetry settings which have been chosen. The example shown in [Figure 14.16](#) would need a 90 degrees rotational symmetry around the Z-axis. If a 60 degrees symmetry is entered instead, the model will be cut back to 60 degrees.

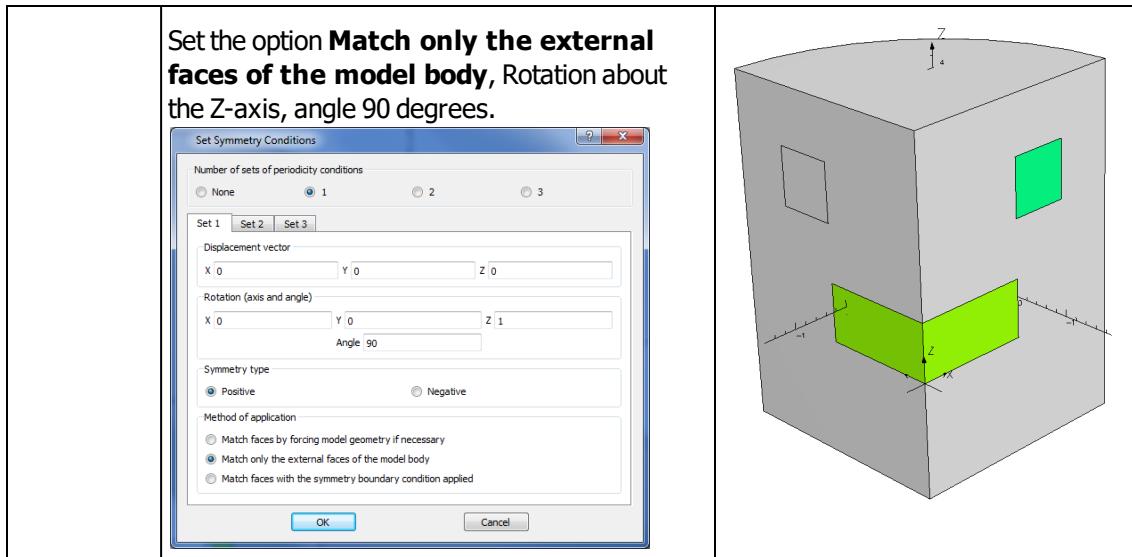


The periodicity condition has to be defined before the model is created and the mesh generated. Note that the actual transformation (forcing the model geometry) is applied during the stage of creating

 the model body via **Create Model Body**.

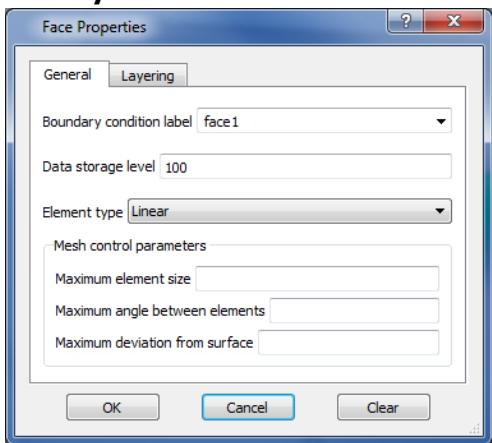
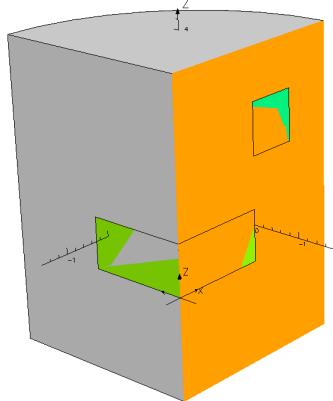
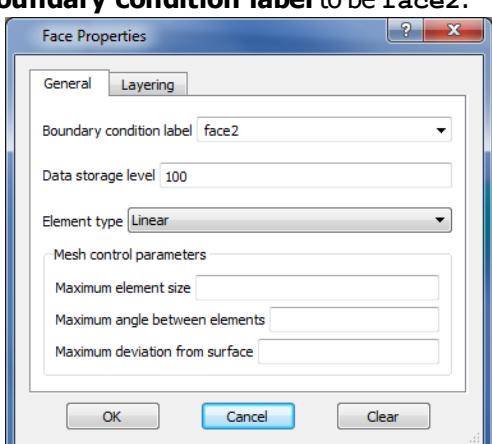
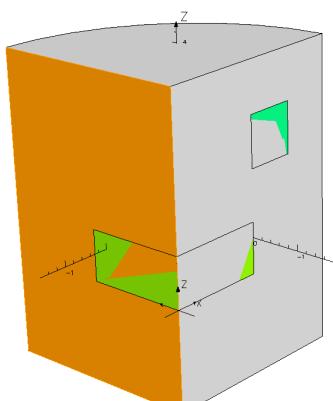
## External faces

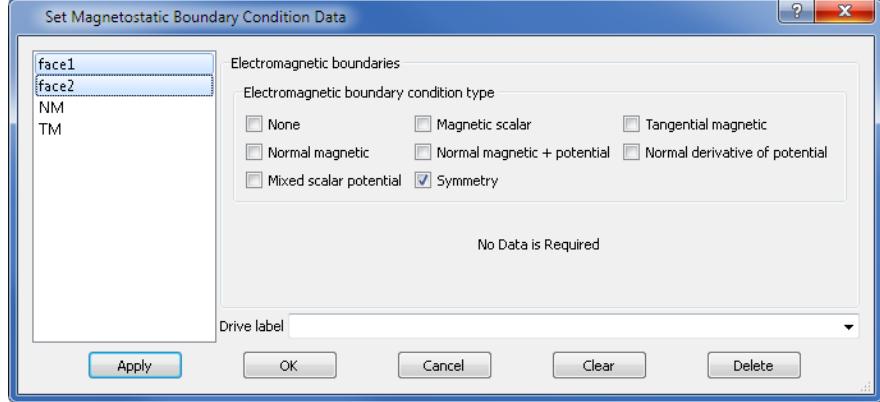
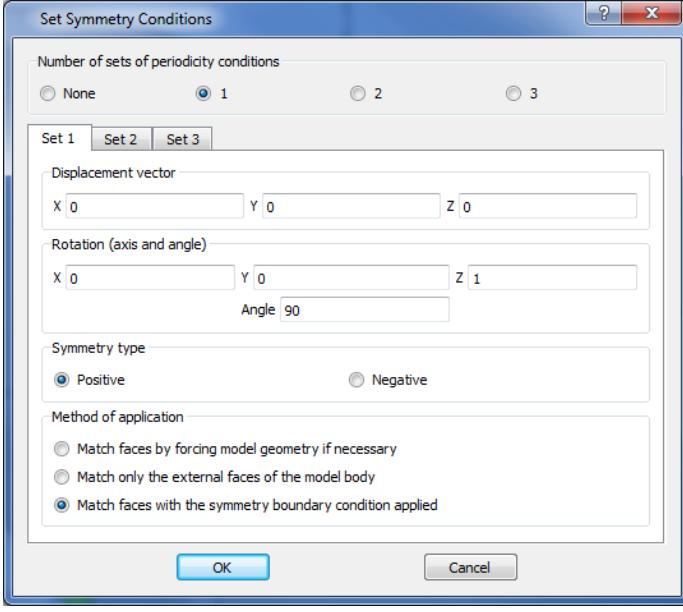
The second choice is matching external faces. This method will use the **Displacement vector** and **Rotation** to define which external faces are to be matched. It is not necessary to set face labels when this option is selected.



## Match labelled faces

The third method requires boundary condition labels to be set on the periodic surfaces. In a first step the faces are picked and labelled; in a second step the labels are assigned a symmetry boundary condition; in the third step the pairing is defined. As before, the displacement and rotation vectors are used to match the faces that are periodic.

 <b>Pick Faces</b>	<p>Pick face 1 as shown. Right-click &gt; <b>Face Properties</b>, and set <b>Boundary condition label</b> to be <b>face1</b>.</p> 	
 <b>Pick Faces</b>	<p>Pick face 2 as shown. Right-click &gt; <b>Face Properties</b> and set <b>Boundary condition label</b> to be <b>face2</b>.</p> 	

 <b>Boundary Conditions</b>	<p>By holding down the shift key, both labels can be selected at the same time. Set the option <b>Symmetry</b>.</p>  <p>The dialog shows two faces selected: face1 and face2. Under Electromagnetic boundary condition type, the Symmetry option is checked.</p>
 <b>Model Symmetry &gt; Advanced</b>	
<p>Set the option <b>Match faces with the symmetry boundary condition applied</b>, Rotation about the Z-axis, angle 90 degrees.</p>  <p>The dialog shows Set 1 selected. Under Displacement vector, X, Y, and Z are all 0. Under Rotation (axis and angle), X, Y, and Z are 0, and Angle is 90. Under Symmetry type, Positive is selected. Under Method of application, Match faces with the symmetry boundary condition applied is selected.</p>	

This setting will use the labelled boundary conditions that have been set on the chosen faces and will match all nodes and facets.

## Analysis of Parts of a Geometric Model

Some electromagnetic and coupled models require several different analyses to be performed on different parts of the structure. Good examples of this are:

- cathode ray tubes where both electrostatic (possibly including space charge) and magnetostatic analyses may be needed;
- eddy current heating where the air regions from the electromagnetic model are not needed in the thermal analysis but the coil volumes must be meshed because they are also heat sources.

It is often desirable to construct one model to cover the different analyses to ensure that the relative positioning of the components for the two analyses is correct. However it can be very inefficient to import all of the structure and mesh from one analysis into the second analysis when it does not form part of the calculation. The following example shows how the Opera-3d Modeller, analysis programs and Post-Processor can be used efficiently in such cases.

## Static Current Flow and Magnetostatic Analyses

### Example model

Figure 14.21 shows the geometry of a spiral coil surrounding a magnetic insert. The objectives are:

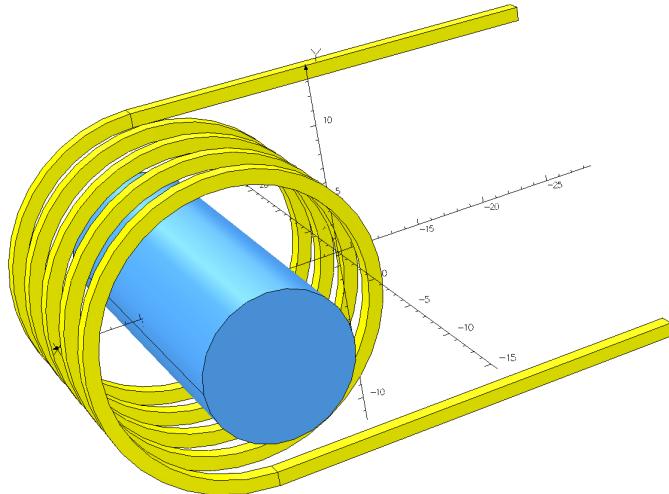


Figure 14.21 Spiral coil and magnetic insert

1. to find the force exerted on the magnetic insert when DC current flows in the coil, and
2. to find the DC resistance of the coil.

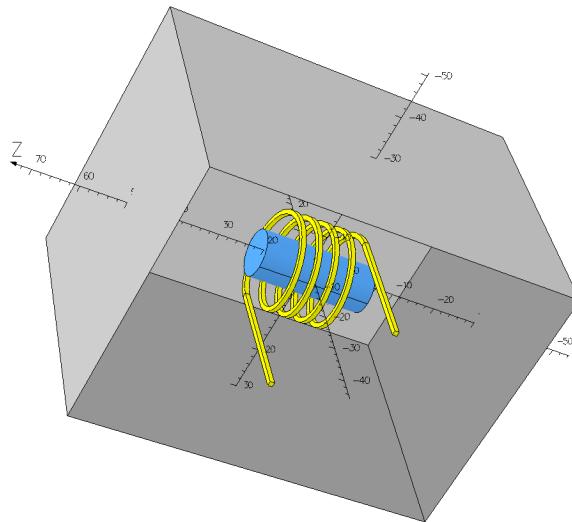
For the force calculation, the spiral coil could be modelled with reasonable accuracy using standard Opera conductors. However these assume that the current density is uniform. To evaluate the coil's

resistance it is necessary to solve for the true current distribution. Consequently two analyses are needed:

1. **Static Current Flow** and
2. **Magnetostatic**, using fields from the current flow analysis as sources.

## Setting up the current flow model

The coil and magnetic insert are shown in [Figure 14.21](#). A reduced potential background region is added to the model as shown in [Figure 14.22](#). All the geometry for both the magnetostatic and current flow analysis is now in place.



*Figure 14.22 Background region added*

The subset of the model for the current flow analysis can be produced by picking the bodies that make up the spiral coil. This is achieved most easily by first picking all bodies using the **Pick All Filter Type Entities** button and then unpicking the bodies which form the background region and

**Create Model Body** will now select only the geometry of the spiral coil for the analysis. A database containing a **Static Current Flow** simulation can now be created.

After analysis, the results can be seen in the Post-Processor. [Figure 14.23](#) shows the current density in the spiral coil and demonstrates the non-uniformity that makes the current flow analysis necessary.

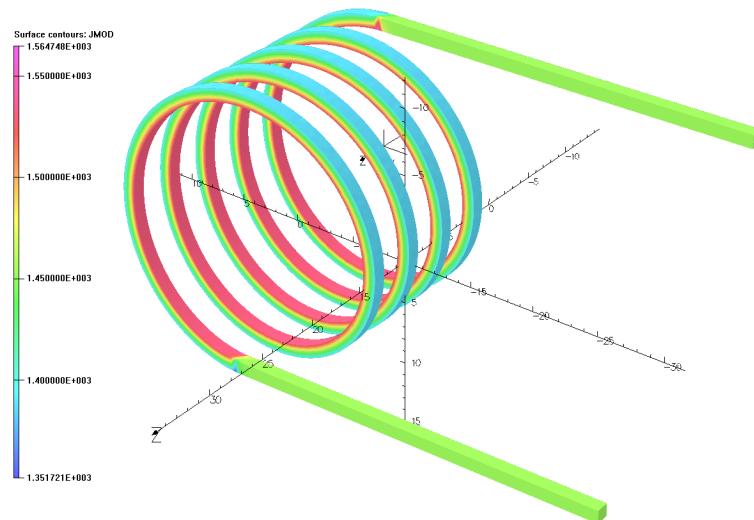


Figure 14.23 Current flow solution

## Setting up the Magnetostatics model

The same technique is now used to create the magnetostatic simulation. In this case only the background region and magnetic insert bodies are picked before the **Create Model Body** instruction is issued. The model body created in this way contains no detail of the coil and is consequently a considerably simpler model to analyse.

Generate surface mesh, and volume mesh. Before creating an **op3** database, note that to use source fields imported from another program into a **Magnetostatic** analysis, it is necessary to select the **Simple** option for **Conductor line integrals** when specifying the **Analysis Settings**. Create the **op3** database, but do not solve it yet, because the source field from the spiral coil is required and must be added first.

## Calculate the source field from the spiral coil

The source field from the spiral coil is calculated by the Post-Processor using the **TABLE** command. This is a three stage process as shown below.

1. Load the unsolved **Magnetostatic** simulation into the Post-Processor and create a table file of the node coordinates:



**Create Table Files > All Node Coordinates...**

and specify a filename, for example **Spiral\_Nodes.table**.

2. Load the **Static Current Flow** simulation into the Post-Processor and create a second table file containing the magnetic field from the currents evaluated at all the coordinates in the **Magnetostatic** simulation, i.e. at the coordinates in the table file created in step 1.

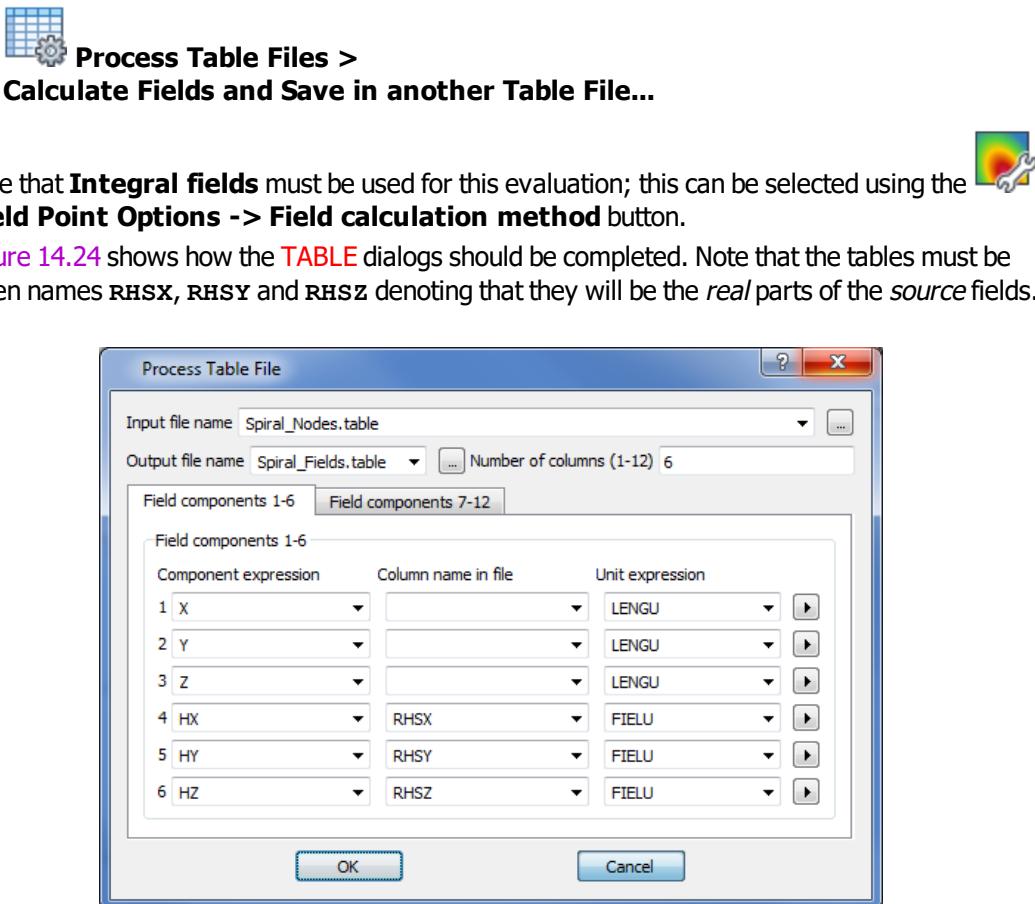


Figure 14.24 Creating the table file of field values

3. Re-load the **Magnetostatic** database into the Post-Processor. Then add the table of source fields created in step 2 to the database as follows:

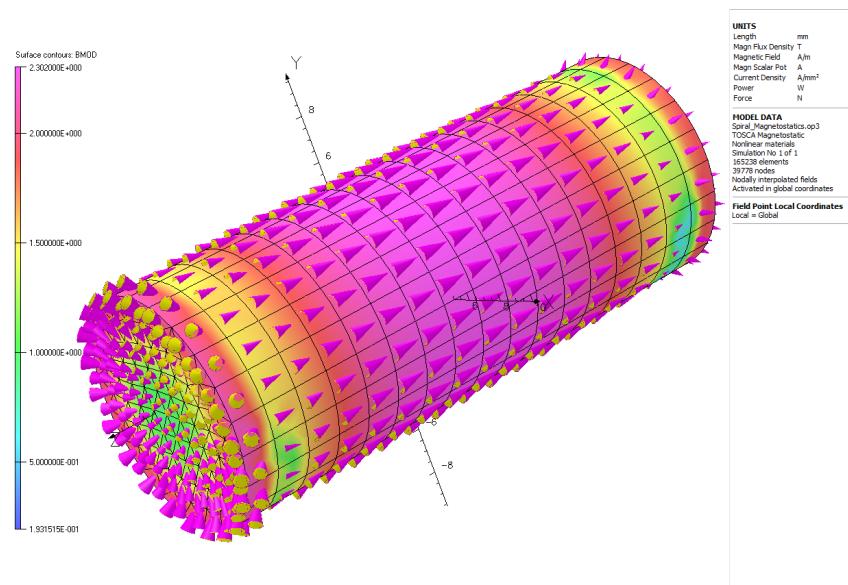


#### Process Table Files > Import Fields at all Nodes or Elements...

and select the name of the table file containing the magnetic field, in this example **Spiral\_Fields.table**.

The **Magnetostatic** simulation is now ready for analysis.

After analysis, the results can be viewed in the Post-Processor. Figure 14.25 shows the magnetic flux density on the surface of the magnetic insert.



For further tests, there are 2 files available in the examples directory:

**Spiral\_CurrentFlow\_3dExample.opc**  
**Spiral\_Magnetostatics\_3dExample.opc.**

## Using Packing Factor to Represent Laminations

In both static and dynamic electromagnetic simulations there is an option for modelling laminated materials by means of single BH characteristic, a packing factor and the orientation of the laminations.

This note describes a method of using the anisotropy feature to model lamination stacks in bulk. The method requires the modification of the solid material BH characteristics by a packing factor ( $f$ ). Two modified characteristics are required: one in the plane of the laminations and one normal to the plane of the laminations.

### In the plane of the lamination

The component of the magnetic field strength tangential to the plane of the lamination,  $\mathbf{H}_t$ , is continuous at the steel/air (insulation) interface, i.e.,

$$\mathbf{H}_t = \frac{\mathbf{B}_{air}}{\mu_0} = \frac{\mathbf{B}_{steel}}{\mu_{steel}} \quad (14.1)$$

so that:

$$\mathbf{B}_{steel} = \frac{\mu_{steel}}{\mu_0} \mathbf{B}_{air} \quad (14.2)$$

The average flux density in the tangential direction,  $\mathbf{B}_t$ , is given by:

$$\mathbf{B}_t = f\mathbf{B}_{steel} + (1-f)\mathbf{B}_{air} = \left[ f\frac{\mu_{steel}}{\mu_0} + (1-f) \right] \mathbf{B}_{air} \quad (14.3)$$

The effective permeability in the plane of the lamination (tangential to the lamination) is given by:

$$\mu_t = \frac{\mathbf{B}_t}{\mathbf{H}_t} = \frac{\left[ f\frac{\mu_{steel}}{\mu_0} + (1-f) \right] \mathbf{B}_{air}}{\frac{\mathbf{B}_{air}}{\mu_0}} = f\mu_{steel} + (1-f)\mu_0 \quad (14.4)$$

### In the direction normal to the lamination

The component of flux density normal to the plane of the lamination,  $\mathbf{B}_n$ , is continuous at the steel/air (insulation) interface, i.e.,

$$\mathbf{B}_n = \mu_0 \mathbf{H}_{air} = \mu_{steel} \mathbf{H}_{steel} \quad (14.5)$$

Hence the average field strength in the normal direction to the lamination is given by:

$$\mathbf{H}_n = f\mathbf{H}_{steel} + (1-f)\mathbf{H}_{air} = \left[ f\frac{\mu_0}{\mu_{steel}} + (1-f) \right] \mathbf{H}_{air} \quad (14.6)$$

Hence the effective permeability in the normal direction is given by:

$$\mu_n = \frac{B_n}{H_n} = \frac{\mu_0 H_{air}}{\left[f \frac{\mu_0}{\mu_{steel}} + (1-f)\right] H_{air}} = \frac{\mu_{steel}\mu_0}{f\mu_0 + (1-f)\mu_{steel}}. \quad (14.7)$$

## Grain Oriented Materials

It is also possible to model laminated or grain-oriented materials using the more general anisotropic material definitions, where 3 different permeabilities for local X, Y and Z directions can be specified. In this case, "functional material properties" have to be used, including the **\$EQUATION** command. More detailed instructions can be found in the Modeller chapter of the ***Opera-3d Reference Manual*** under "The **MATERIALS** Command".

## Complex Material Properties

---

### Introduction

A simple model of hysteresis can be represented by a complex material permeability in steady-state AC analyses. Hysteretic materials can also be modelled using the transient analysis programs (see [Using Hysteretic Materials \[page 596\]](#)).

This note reviews the AC approximation and shows how the losses can be calculated.

### Complex Permeability Model

The harmonic solution programs use complex variables to model time harmonic fields. For example, the magnetic flux density can be expressed as function of time as follows:

$$B(t) = B_0 \cdot \cos(\omega t + \varphi) = \operatorname{Re}(B_0 \cdot e^{i(\omega t + \varphi)}), \quad (14.8)$$

where  $B_0$  is the amplitude of the flux density.

An elliptical B-H curve can be represented by a phase change  $\alpha$  between the field intensity and the flux density:

$$H(t) = \frac{1}{\mu} \cdot B_0 \cdot \cos(\omega t + \varphi + \alpha) = \operatorname{Re}\left(\frac{1}{\tilde{\mu}} \cdot B_0 \cdot e^{i(\omega t + \varphi)}\right) \quad (14.9)$$

with

$$\tilde{\mu} = \mu \cdot e^{-i\alpha}. \quad (14.10)$$

The equation of the ellipse in the B-H plane is:

$$B^2 - 2\mu \cos(\alpha) BH + \mu^2 H^2 = \mu^2 \sin(\alpha)^2 H_0^2 \quad (14.11)$$

or

$$H = \frac{1}{\mu} \left( B \cos(\alpha) \pm \sqrt{\mu^2 H_0^2 - B^2 \sin(\alpha)^2} \right). \quad (14.12)$$

For example a material with a relative permeability of 100 and a phase angle of 20 degrees ( $\mu_r = 100 e^{-i(\pi/9)}$ ) gives a B-H curve as shown in [Figure 14.26](#).

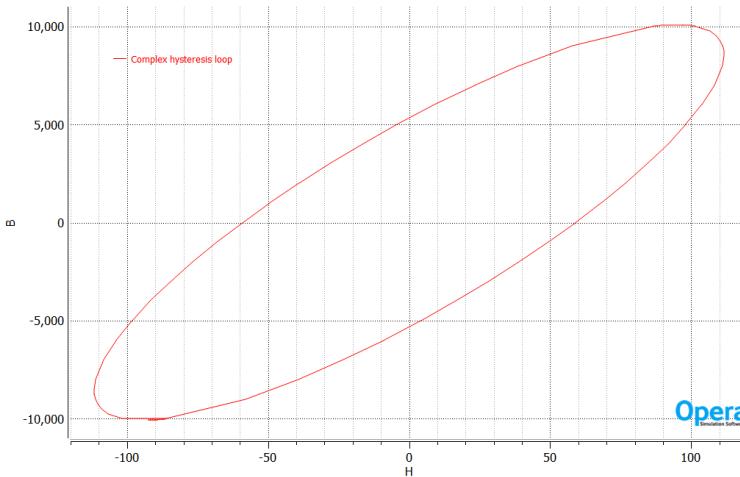


Figure 14.26 Typical hysteresis loop

## Hysteresis Losses

The complex permeability described above provides a phase shift between B and H and can be used to model hysteresis.

$$H = H_0 \cos(\omega t + \varphi), \quad (14.13)$$

$$B = \mu H_0 \cos(\omega t + \varphi - \alpha). \quad (14.14)$$

The hysteresis energy loss per cycle is given by the area of the ellipse:

$$W = \pi \sin(\alpha) \int \mu H_0^2 d\Omega \quad (14.15)$$

so that the average hysteresis power loss per cycle is:

$$P = \frac{\omega}{2} \sin(\alpha) \int \mu H_0^2 d\Omega. \quad (14.16)$$

In the Post-Processor the average power loss (equation 14.16) is available as a separate system variable called **HYSPOWER** and is evaluated in the **ENERGY** command.

## Restrictions of the Model

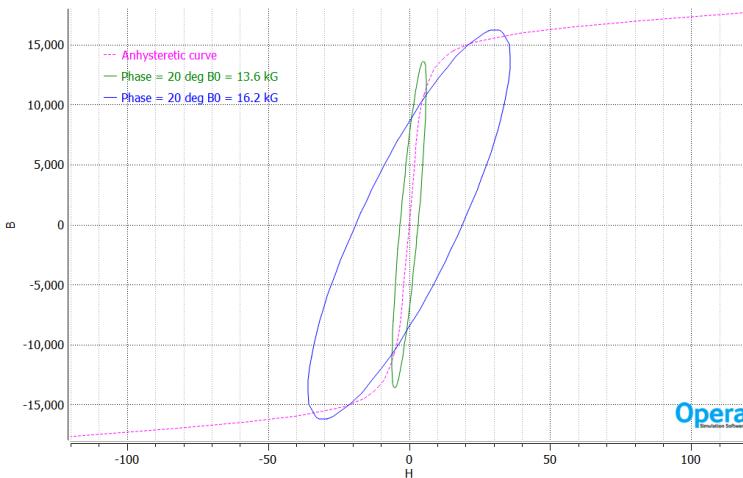
The introduction of complex permeability corresponds to a simple model for hysteresis. Although it is straightforward to replace the real permeability by a complex value in the linear steady-state AC equations, the model is then restricted to steady-state AC applications. The minor loops in this model will be concentric similar ellipses, so there will be no harmonic distortion implied in the analysis.

## Complex Materials in Harmonic EM and Harmonic HF

When the database (**op3**) file is being created in the Modeller, the **Set Material Properties** of the **Model** menu allows the user to specify the properties for each material used in the model. For **Harmonic Electromagnetic** and **Harmonic High Frequency** solutions the **Phase lag** (in degrees) may be specified. Note that if the **Anisotropic** options are chosen, the same phase lag is used in all three local coordinate directions. The same is true for the Pre-Processor, using the **Materials** option on the **Analysis data** menu.

## Complex Permeability in Nonlinear Models

When complex permeability is used in nonlinear materials (**Harmonic EM** solution only) the choice of phase angle can become difficult. As shown in [Figure 14.27](#), choosing a single phase angle that is applied over the whole BH curve can give very different sized hysteresis loops. In fact the loop size, and hence the hysteresis losses, for a material operating at a peak flux density of 13.6 kilogauss and at 16.2 kilogauss should have been nearly identical.



*Figure 14.27 Using same phase angles for different parts of the BH curve gives poor representation*

To obtain a more accurate solution it is better to use several different phase angles for parts of the model at different regions of the curve. Suitable angles can be determined as follows.

1. Run the model nonlinearly without complex permeability.
2. Determine the variation of peak flux density in the magnetic materials and discretize into a number of ranges. The shape of the B-H curve will dictate this discretization but it is unlikely that all ranges will be equal. For example, the ranges (0-0.5), (0.5-1.0), (1.0-1.2), (1.2-1.3), (1.3-1.35) and (1.35-1.4) tesla may be suitable for a material that is not near saturation.
3. Re-assign the materials in the model so that a different material is used for the parts of the model that fall into each range of flux density or define a function for **PHASE** which depends on the magnitude of **B**.

4. Run the new model nonlinearly with complex permeability using the same B-H curve for all materials but assigning a different phase angle for each material name.

## Tabular Functions

Many physical properties in Opera models depend on the value of a physical field within the material. The most obvious is the material's magnetic characteristic, where the magnetic flux density is expressed as a function of the magnetic field strength in a table. These tables are stored as files with the extension **bh**. Because Opera is primarily aimed at electromagnetic analysis (although thermal and stress analysis modules are also available), tools have been provided to generate these **BH** tables.

Other physical properties can be expressed as functions of solution variables. For example, in a thermal analysis, it can be important to express the thermal conductivity as a function of the temperature. In some cases, the material properties can be expressed as functions using algebraic expressions. However, in the general case, finding a polynomial or other function that fits the behaviour can be difficult and it is better to express the behaviour as a "Tabular Function".

### Tabular Function Files

Tabular functions are stored in **table** files using the same format as the table files generated in Post-Processor for exchanging geometrically dependent data between models (for example, transferring losses computed by electromagnetic solvers to thermal solutions) or saving the results stored in the internal buffer from a **CARTESIAN**, **LINE** etc. command. Further details on the formats for **table** files can be found in the **Opera-3d Reference Manual**.

When the **table** files are used for physical property functions, the independent variables are not the geometric coordinates (**X**, **Y** and **Z**) but the values that can be computed by the Opera analysis program (flux density, temperature, etc.). The values of independent variables need not be equally spaced but must have a logical ordering:

- a one variable function must be defined using a table with increasing values of the independent variable;
- a two variable function must be defined using a table with a rectangular grid of independent variable values;
- a three variable function must be defined using a table with a 3-dimensional grid of independent variable values.

Two examples follow: a function of 2 variables used in **Quench** and a function of 3 variables used in the **Demagnetization** solver. The examples illustrate how the independent variable values should be ordered.

#### A function of 2 variables

In a **Quench** simulation, the critical current in a superconducting material is dependent on both the temperature and the magnetic flux density. The table file has 3 columns as shown below. The first line of the table indicates how many different values exist in the variable columns (11x3). Note that the values in column 1 ( $T$ ) vary most rapidly. When all 11 values of  $T$  have been given for one value

in column 2 (`BFIELD`), the values of `T` are repeated for the next value of `BFIELD`. The third column contains the function values corresponding to the values of `NBTI_JC (T;BFIELD)`.

11	3	1	1
1	T	[1.0]	
2	BFIELD	[FLUXU]	
3	NBTI_JC	[CURDU]	
0	0.00E+00	4.75E+06	
1	0.00E+00	4.66E+06	
2	0.00E+00	4.39E+06	
3	0.00E+00	3.93E+06	
4	0.00E+00	3.32E+06	
5	0.00E+00	2.68E+06	
6	0.00E+00	2.04E+06	
7	0.00E+00	1.40E+0	
8	0.00E+00	7.65E+05	
9	0.00E+00	1.28E+05	
10	0.00E+00	-5.10E+05	
0	5.00E+03	2.14E+06	
1	5.00E+03	2.10E+06	
2	5.00E+03	1.97E+06	
3	5.00E+03	1.76E+06	
4	5.00E+03	1.48E+06	
5	5.00E+03	1.18E+06	
6	5.00E+03	8.88E+05	
7	5.00E+03	5.93E+05	
8	5.00E+03	2.98E+05	
9	5.00E+03	3.55E+03	
10	5.00E+03	-2.91E+05	
0	1.50E+04	1.02E+06	
1	1.50E+04	9.99E+05	
2	1.50E+04	9.35E+05	
3	1.50E+04	8.29E+05	
4	1.50E+04	6.86E+05	
5	1.50E+04	5.38E+05	
6	1.50E+04	3.90E+05	
7	1.50E+04	2.41E+05	
8	1.50E+04	9.31E+04	

9	1.50E+04	-5.52E+04
10	1.50E+04	-2.04E+05

## A function of 3 variables

In demagnetization modelling, the magnetization is a function of the flux density, the peak flux density and the temperature. The table has 4 columns as shown. The table here has fewer entries than would normally be used and does not represent a real material. It is used to illustrate the table format.

The first line of the table indicates how many different values exist in the variable columns (3x3x3). Note that the values in column 1 ( $B$ ) vary most rapidly. When all 3 values of  $B$  have been given, the value in column 2 ( $B_{MAX}$ ) is incremented to its next value and the value of  $B$  goes back to its initial value. When all 9 (3x3) combinations of  $B$  and  $B_{MAX}$  have been given in this way, the value of column 3 ( $T$ ) is incremented and the sequence of  $B$  and  $B_{MAX}$  values is repeated. Column 4 holds the function values  $M_K1(B;B_{MAX};T)$ .

3	3	3	2
1	$B$	[TESLA]	
2	$B_{MAX}$	[TESLA]	
3	$T$	[1]	
4	$M_K1$	[TESLA]	
0			
0	0.39	20	5.91E-04
0.18	0.39	20	0.1604
0.36	0.39	20	0.3618
0	0.78	20	6.75E-04
0.18	0.78	20	0.1723
0.36	0.78	20	0.3804
0	1.17	20	6.96E-04
0.18	1.17	20	0.1794
0.36	1.17	20	0.3975
0	0.39	30	5.92E-04
0.18	0.39	30	0.1614
0.36	0.39	30	0.3628
0	0.78	30	6.75E-04
0.18	0.78	30	0.1743
0.36	0.78	30	0.3834
0	1.17	30	6.97E-04
0.18	1.17	30	0.1823
0.36	1.17	30	0.3994

0	0.39	40	5.94E-04
0.18	0.39	40	0.1653
0.36	0.39	40	0.3626
0	0.78	40	6.85E-04
0.18	0.78	40	0.1763
0.36	0.78	40	0.3883
0	1.17	40	7.17E-04
0.18	1.17	40	0.1842
0.36	1.17	40	0.4343

## Units in tables

There are two versions of table indicated by the last number on the first line of each of the above tables:

1. Values in version 1 tables are in cgs units and the text inside square brackets gives the conversion factors as algebraic expressions. These will be evaluated to obtain the conversion factors when other units are required. The table in the example of a function of 2 variables is version 1.
2. Values in version 2 tables are in any choice of units and the text inside square brackets gives the names of the units used as an algebraic expression. The programs decode the unit expressions to calculate conversion factors if unit conversion is necessary. The table in the example of a function of 3 variables is version 2.

## Using Tabular Function Files to Create Functional Variables



The tables are imported into the Opera programs by selecting **Define and list tabular functions**. In a command file or from the console, the command is **\$FUNCTION** with the syntax:

```
$ FUNCTION name.table
```

The **\$FUNCTION** command will create a function that can be used in expressions from the name of the final column of the table file. For example, reading in the functional **table** files above would create the following functions:

```
NbTi_Jc(T;B)
M_K1(B;BMAX;T)
```

which could be used in any expression in a similar way that the built-in functions such as **SIN**, **EXP**, **ATAN2** are used.

Note that the separator between the arguments of the functions is a semi-colon.

## Interpolation of Tabular Functions

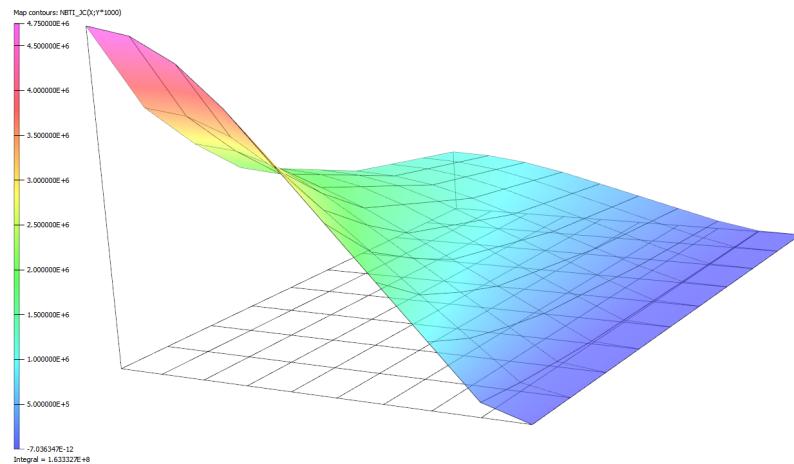
Tabular functions are defined at a set of discrete points. When a function is used in the software, values might be required at any values of the independent variables. The software interpolates between the function values in the table in 2 or 3-dimensional space to find the value corresponding to the independent variables.

If the independent variables are outside the range of the table, the software will substitute the closest values in the range of the table. It is therefore important that the table covers the range of independent variables which will be needed by the software.

## Display of Tabular Functions

Tabular functions can be displayed using the **CARTESIAN** and **MAP** commands in the Post-Processor. The coordinates given to the **CARTESIAN** command should correspond to the independent variables. For example, using the function of 2 variables described above:

- the range of X should be 0 to 10, corresponding to T;
- the range of Y should be 0 to 15, corresponding to BFIELD/1000;
- Z can be a constant;
- the **COMPONENT** should be **NBTI\_JC (X;Y\*1000)** .



*Figure 14.28 Histogram of a Tabular Function*

Note: with version 1 tables it is best to use CGS units when displaying the function values to avoid all unit scaling.

# Simulation of Magnetization

---

## Introduction

This note describes the facilities that have been included in Opera-3d to allow the orientation and strength of a permanent magnet to be related to a magnetizing field. The use of the results in an application device will also be demonstrated, using an example of a Permanent Magnet DC motor (PMDC).

This class of model can be simulated in two ways.

- In the first method, simulation of magnetization is implemented using functional Volume Properties (local coordinates and packing factor) of a material. The functions can reference element values, which can be imported into the Opera database from table files. This method only uses the **Magnetostatic** solver and has some limitations:
  - During a magnetizing cycle the permanent magnet material can only be described by its virgin magnetization curve. The option to return down a demagnetization curve is not included.
  - In the application device the different levels of magnetization are represented by adjusting the packing factor of the material in each element, to correspond to the maximum magnetizing field experienced by that element. The packing factor is simply applied as a scaling factor to the BH properties of an element.
- The second method uses the much fuller representation of demagnetization available in the **Demagnetization** solver. Analysis with this solver includes eddy current effects during the magnetization process. In addition, the material properties also include a recoil permeability. This makes it possible to determine the "in-service" performance of a device in which the magnets are used.

## Using Magnetostatic solver: Packing Factor and Material Orientation

In order to use the first approach, it is necessary to describe the packing factor as a function of the magnetizing flux density. A polynomial of up to 5th order is normally sufficiently accurate, and this can be easily calculated using a spreadsheet.

Let  $\mathbf{B}$  be the calculated magnetizing flux density in a finite element. The material orientation angles required to define the magnetization in the direction of  $\mathbf{B}$  are

$$\theta = \text{atan} \left( \frac{\sqrt{B_x^2 + B_y^2}}{B_z} \right) \quad (14.17)$$

$$\varphi = \text{atan} \left( \frac{B_y}{B_x} \right) \quad (14.18)$$

$$\psi = 0 \quad (14.19)$$

If  $H_{cmax}$  is the maximum coercive field strength of a permanent magnet material, which is defined as a packed material in Opera, with a packing factor of  $\gamma$ , the coercive field strength used by Opera,  $H_c$ , will be

$$H_c = \gamma H_{cmax}. \quad (14.20)$$

Since the coercive field strength varies with the magnetizing field, the packing factor should be calculated using

$$\gamma = \frac{H_c(B)}{H_{cmax}}. \quad (14.21)$$

As discussed above, it is convenient to represent  $H_c(B)$  as a polynomial in terms of the magnitude of the magnetizing flux density  $B$

$$H_c(B) = a_1 B + a_2 B^2 + a_3 B^3 + \dots \quad (14.22)$$

The coefficients can be easily determined by fitting a polynomial trend line to measured values of  $H_c$  as a function of the magnetizing flux density  $B$ .

## Limited Approach using Magnetostatic solver

### Modelling the magnetizing device

Figure 14.29 shows a model of a magnetizing device, used for a pair of Permanent Magnet DC (PMDC) motor magnet sections. The magnetizing model was used to calculate the magnetizing flux density in the permanent magnet material, before using this in the application device model.

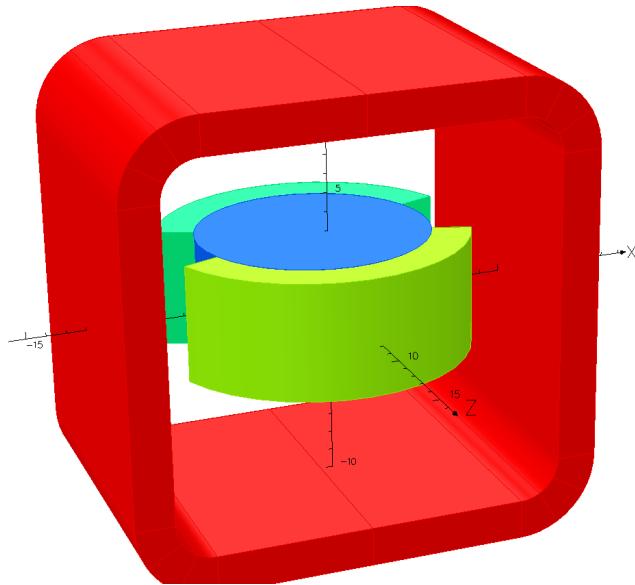


Figure 14.29 A model of a magnetizing device

When preparing the model, the magnetizable components were given virgin BH characteristics, as shown in Figure 14.30.

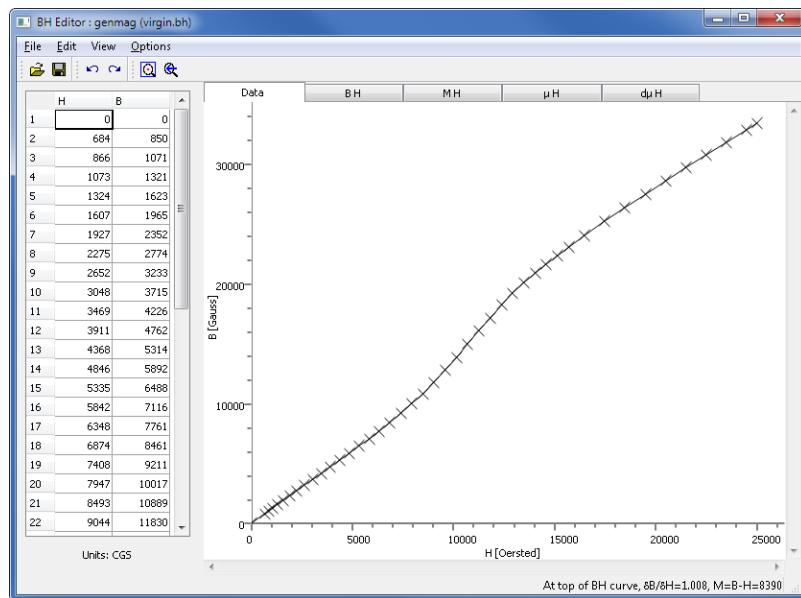
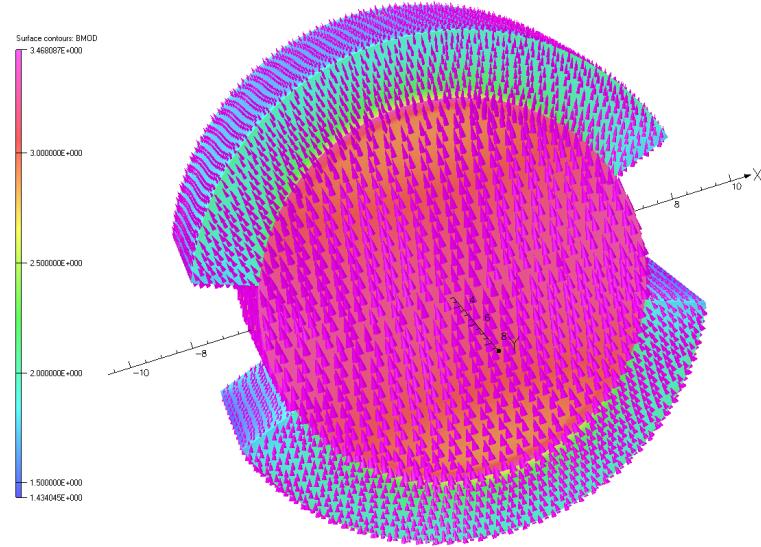


Figure 14.30 Virgin BH characteristic given to the permanent magnet material

Note that no other special modelling considerations exist. The procedure may be implemented using either static or time varying fields. However, if a time varying field is used, the maximum field in the magnetizable material will need to be extracted from the solutions that have been stored.

[Figure 14.31](#) shows the resulting magnetic field in the magnetizing device, solved in **Magnetostatic**.



*Figure 14.31 Magnetic field in the magnetizing device*

## Modelling the PMDC motor

The application device, the PMDC motor, was subsequently modelled using the magnetized permanent magnets. The following guidelines were observed when preparing the application device model in Opera-3d:

1. A Volume Label was specified for each magnet section
2. For each Volume Label, the orientation Euler angles and packing factor were defined as variables. Variable names must always begin with **r**, but there are no other restrictions on the choice. The values would be supplied later using tables. [Figure 14.32](#) shows an example definition of a Volume Label, **mslabe** with **Other orientation angles** set to **RTHETA**, **RHPI** and **RPSI** and **Packing factor** to **RPF**.

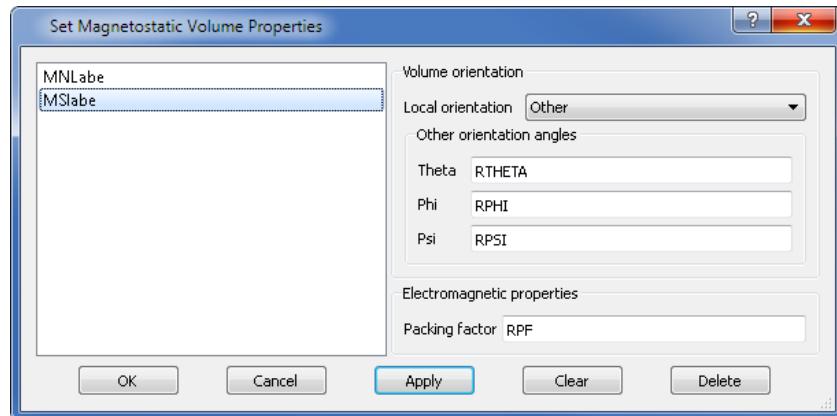


Figure 14.32 Volume label specification in the PMDC motor model

- The permanent magnet sections material properties were specified as packed, nonlinear, as shown in Figure 14.33.

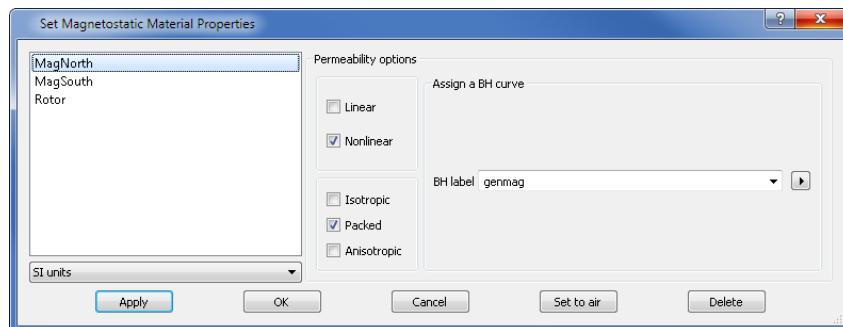


Figure 14.33 Permanent magnet material property specification

- The BH characteristic of the magnet was specified as a fully magnetized material. This would subsequently be degraded according to the packing factor that is supplied as a table.

## Analysis and post-processing

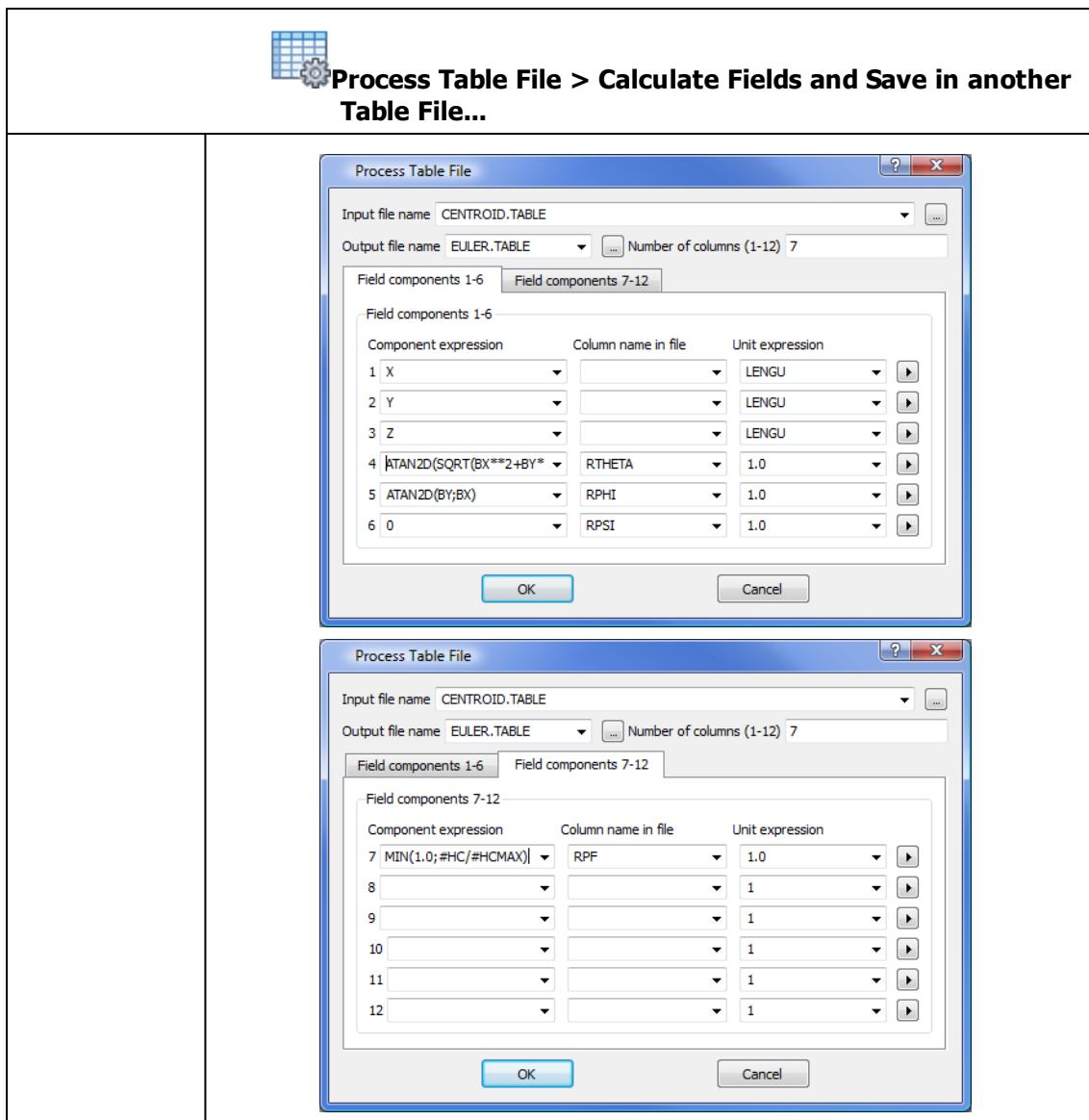
Having prepared the application device model according to the above guidelines, an Opera database was prepared for analysis. Before analysis, the Post-Processor was used to add tables defining the Euler angles and packing factors of the magnet materials. The following procedure was adopted:

- The application (PMDC motor) model was loaded into the Post-Processor.
- An element centroid table, **CENTROID.TABLE**, was created:



3.

- CENTROID.TABLE** contained 3 columns: X, Y, Z
4. The solved magnetizing device model was loaded. Parameter **#HC** was defined as a function of the flux density B and constant **#HCMAX**.
  5. The magnetizing field was used to calculate the orientation of the magnetization and its strength. To create **EULER.TABLE**:



6. The magnetization information was added to the application database.



**Process Table File > Import Fields at all Nodes or Elements...**

7.

For the filename, specify **EULER.TABLE** (the file created in step 4).

The application model (PMDM motor) was then analysed, and

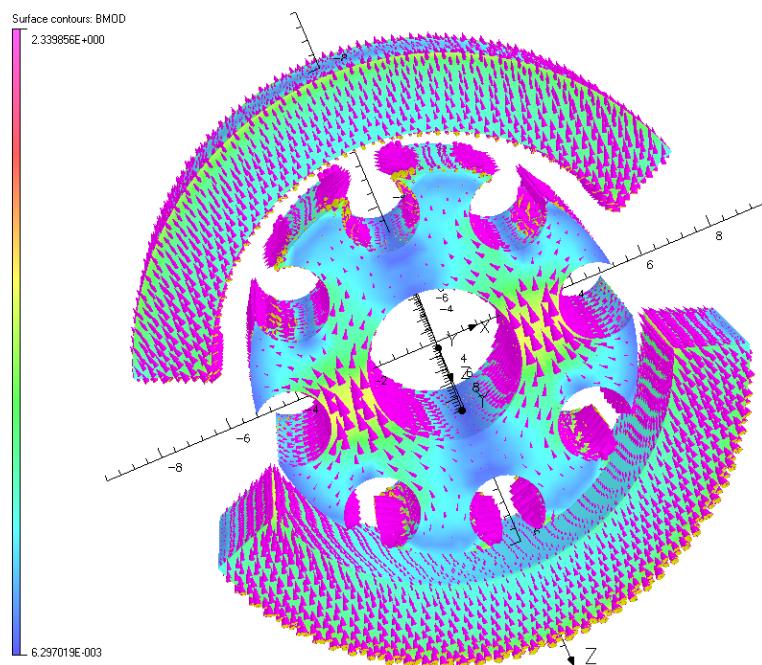


Figure 14.34 Flux Density in PMDC Motor

Figure 14.34 shows the resulting magnetic field distribution. It is observed that the field orientation in the magnet is neither radial, nor parallel, but an accurate representation of the field resulting from the magnetizing fixture set up.

## Command Script

A command script has been created which automates the transfer of magnetization information to the application model.

```
/ Command Script to evaluate the Magnetizing field at the
/ element centroids of the magnets in an application model
/ and add these tables to the application model to specify
/ the direction and strength of the magnets
$PROMPT MAGNETIZING 'Filename of the Magnetizing Database'
$ASK #MAGCASE 'Magnetizing Database Solution case number'
```

```

$PROMPT APPLICATION 'Filename of the Application Database'
$PROMPT MATERIAL 'Command file specifying the magnets HC vs Bmag'
/-----
/ WARNING - Uses and deletes temporary files
/ TEMP$$$APP.TABLE and TEMP$$$MAG.TABLE
$EXIST TEMP$$$APP.TABLE

$IF FILEEXISTS GT 0.5
$OS DEL TEMP$$$APP.TABLE
$END IF
$EXIST TEMP$$$MAG.TABLE
$IF FILEEXISTS GT 0.5
$OS DEL TEMP$$$MAG.TABLE
$END IF
/-----
/ Assumes case=1 in the application file
ACTIVATE FILE=&APPLICATION& CASE=1
LOAD
/ Output the element centroid coordinates
TABLE INFILE=ELEM OUTFILE=TEMP$$$APP.TABLE COLUMNS=3,
F1=X UNIT1=LENGU F2=Y UNIT2=LENGU F3=Z UNIT3=LENGU
/ Activate the MAGNETIZING model
ACTIVATE FILE=&MAGNETIZING& CASE=%INT(#MAGCASE)
LOAD
/ Evaluate the field at a point (in the model?) to set system variables
POINT 0 0 0 COMP=BX
/ run the command file to set up the material Hc function
/ THIS Should return parameter functions for #HC and #HCMAX
$COMI &MATERIAL&
/ Process the element table and output directions and packing factor
TABLE INFILE=TEMP$$$APP.TABLE OUTFILE=TEMP$$$MAG.TABLE COLUMNS=7,
F1=X UNIT1=LENGU F2=Y UNIT2=LENGU F3=Z UNIT3=LENGU,
F4=ATAN2D(SQRT(BX**2+BY**2);BZ) NAME4=RTHETA UNIT4=1.0,
F5=ATAN2D(BY;BX) NAME5=RPHI UNIT5=1.0, F6=0 NAME6=RPSI UNIT6=1.0,
F7=MIN(1.0;#HC/#HCMAX) NAME7=RPF UNIT7=1.0
/ Add The magnet orientation and strength information to the Database
ACTIVATE FILE=&APPLICATION& CASE=1
LOAD
TABLE INFILE=TEMP$$$MAG.TABLE OUTFILE=DBAS
LOAD -1
/* The application model can now be solved.
/* End of Script.

```

A typical script for the definition of the material properties could have the following form (as required above &MATERIAL&):

```

/-----
/ Define the relationship between magnetizing flux density
/ and Coercive field strength
/ This data is for material MATERIALDEN2 supplied by Erlangen
/ Note coefficients calculated using Gauss and Oersted, so change
/ units of Bmod
$PARA #b BMOD*FLUXU
$PARA #HC 0.0812*#b-1e-5*#b**2+4e-9*#b**3-2e-13*#b**4+3e-18*#b**5
$PARA #HCMAX 5420
/-----

```

## Using the Magnetization Analysis Module

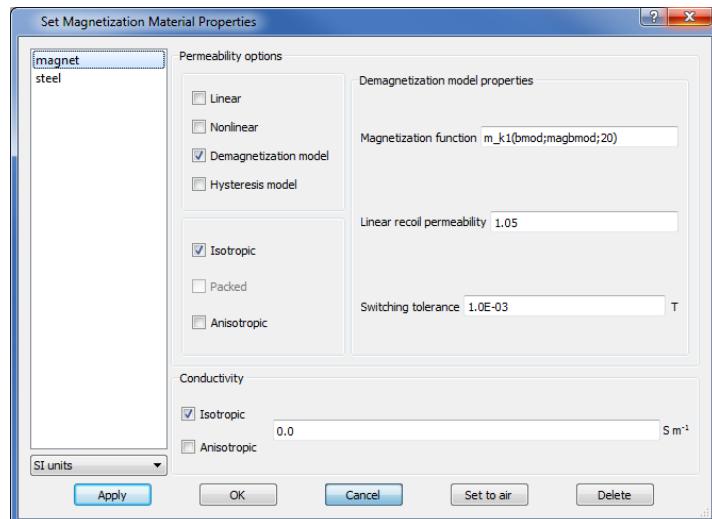
### Material models

In the **Magnetization** solver, hard magnetic material properties are represented by three pieces of information:

1. A magnetization function that the program will use to calculate the magnetization of the material during demagnetization (i.e. while the fields are decreasing). This function specifies the magnetization, in flux density units, and will typically be a function of the flux density ( $B$ ), the magnetizing or peak flux ( $B_{max}$ ) and the temperature ( $T$ ), for example  

$$M = M(B, B_{max}, T)$$
. The data is stored in table files that consist of 4 columns<sup>1</sup>. More details on functional tables can be found in [Tabular Functions \[page 576\]](#).
2. The recoil permeability which will be used if the applied field is greater than the minimum field during demagnetization.
3. A switching tolerance, on switching between magnetization and demagnetization. The change in flux density must reverse by more than this value before the state of the material will change from virgin to demagnetizing or from demagnetizing to recoil.

This information is provided to the Modeller using the Materials Properties dialog (with the Magnetization solver options enabled), see [Figure 14.35](#).



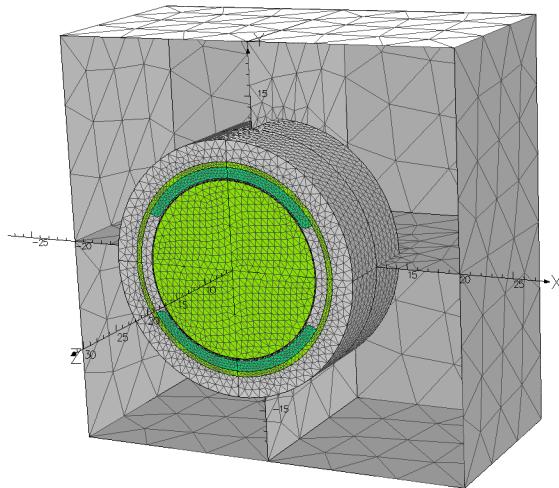
*Figure 14.35 Magnetization Material Properties Dialog*

<sup>1</sup>Note that if data is only available at a single temperature, the 3rd column of the table file can be omitted.

## Magnetizer example

The Magnetization solver is a time transient simulation with eddy currents. In general, permanent magnets are magnetized in a fixture with a pulse of field that is designed to completely saturate the magnet material and produce the desired orientation of residual magnetization.

In the example<sup>1</sup> shown in [Figure 14.36](#), the fixture consists of a steel cylinder between the magnet poles and a steel tube surrounding the magnets, all placed within a uniform field (external field option scaled with a peak pulse of 0.2 seconds). The method of applying the external field is similar to an example that is discussed in the **Additional Support Knowledgebase** under "External Fields in TOSCA & ELEKTRA".



*Figure 14.36 Model used for the simulation of the process of magnetization*

After the pulse has decayed, this configuration produces a uniform, Y-directed magnetization field as shown in [Figure 14.37](#). The screenshot shows the magnetization vectors at a simulation time of 1 second. Before the screenshot was taken, the following parameters were set up:

```
$PARAMETER NAME=#MX VALUE=BX/mu0-HX DESCRIPTION='Magnetization in X'  
$PARAMETER NAME=#MY VALUE=BY/mu0-HY DESCRIPTION='Magnetization in Y'  
$PARAMETER NAME=#MZ VALUE=BZ/mu0-HZ DESCRIPTION='Magnetization in Z'  
/  
$PARAMETER NAME=#M_MOD VALUE=SQRT (#MX^2+#MY^2+#MZ^2),  
$PARAMETER DESCRIPTION='Absolut value of Magnetization'
```

---

<sup>1</sup>The model **Demag\_DEMAG\_MagnetisingDevice.opc** is available in the examples folder.

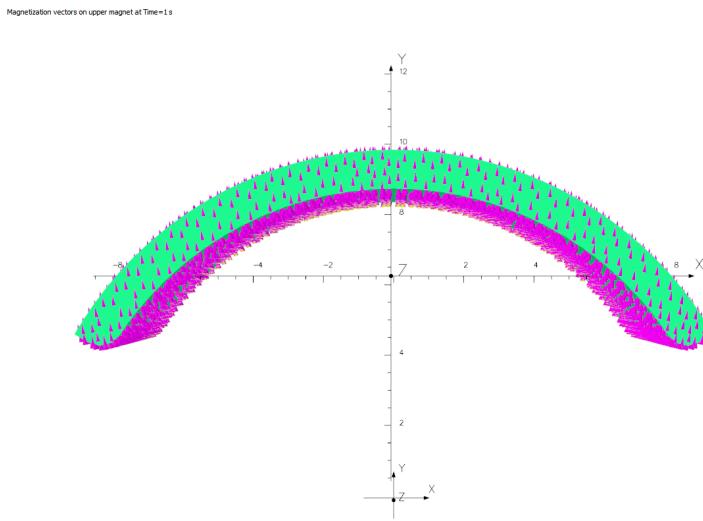


Figure 14.37 Magnetization vectors on upper magnet at Time=1 s

After the magnetization the magnet material operates in the second quadrant, therefore the magnetic field H points in the opposite direction as shown in Figure 14.38. This is the field distribution in the magnets and the steel at the end of the magnetizing sequence.

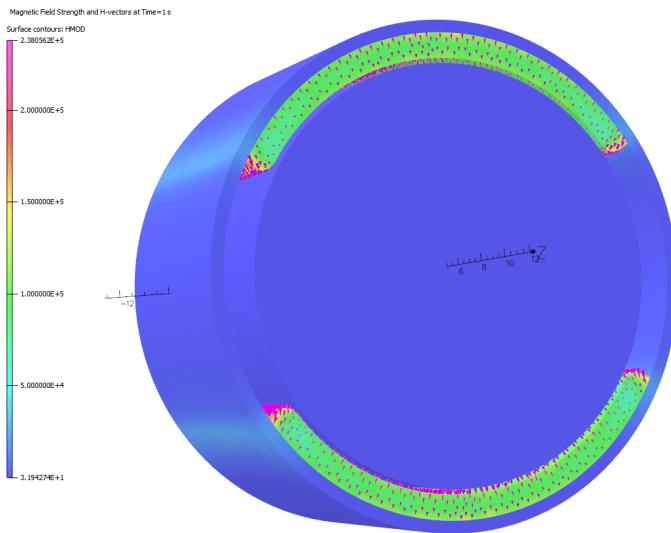


Figure 14.38 Residual field strength after magnetization

## Eddy currents

The presence of the eddy currents in the steel influences the distribution of magnetization in the magnets. Figure 14.39 shows the eddy currents induced as the magnetizing field reaches its peak value

computed by the Magnetization solver.

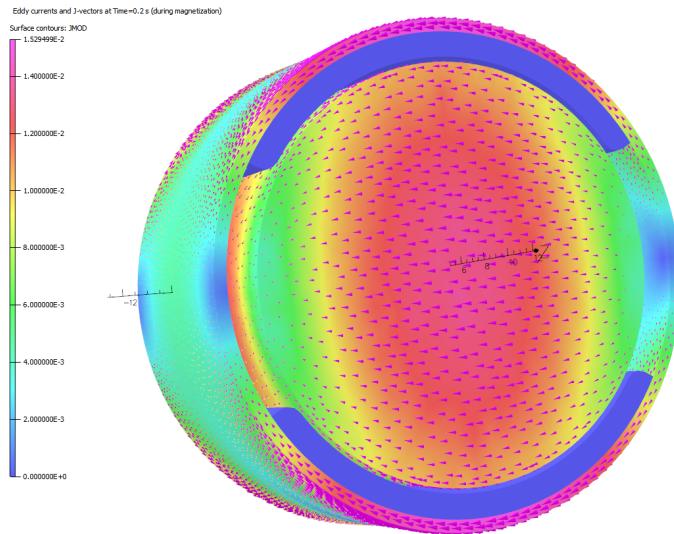


Figure 14.39 Eddy currents during magnetization process

## Export of magnetization into other Opera-3d analyses

The magnetization distribution can be imported into other 3D application models using tables in a similar way as described in the earlier part of this application note.

The magnetization data can be imported into Transient Electromagnetic, Transient Motion and Magnetostatic models. The three variables to be exported are **MAGBX**, **MAGBY** and **MAGBZ** which must be given the names **RMAGBX**, **RMAGBY** and **RMAGBZ** in the table file to identify them as real variables. These are the 3 components of the peak (magnitude) flux density reached in each element of the magnet during the DEMAG simulation.

Note that the application model must use a functional table for the material properties of the permanent magnet. Figure 14.40 shows the distribution of **MAGBMOD**, the magnitude of the peak flux density for the magnet elements, after it has been imported into a PMDC model. No other additional data or user variables are required and Transient Electromagnetic or Transient Motion can then be run in the normal way.

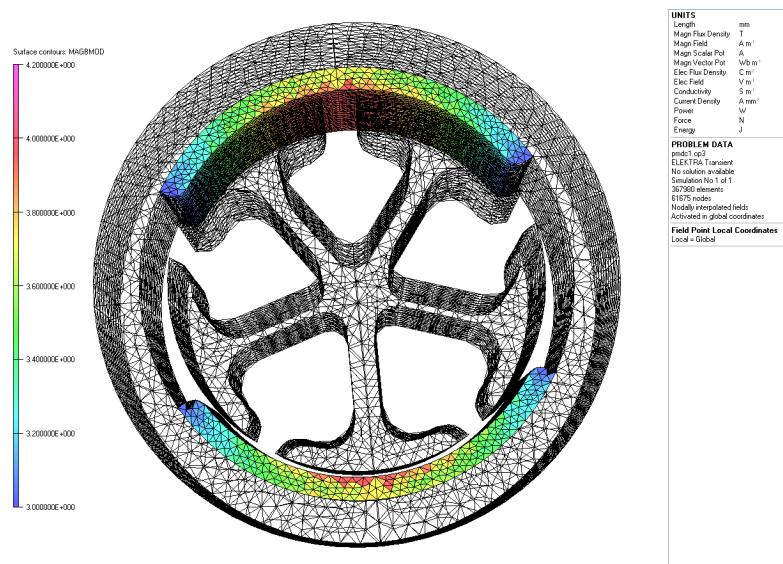


Figure 14.40 Distribution of MAGBMOD in PMDC model

## Demagnetization In Service

When the Magnetization material model is used in one of the transient electromagnetic solvers it is also possible to simulate further demagnetization of the materials. Additional tables, **MINBX**, **MINBY** and **MINBZ** are created which store the components of the minimum value of flux density that each element has reached during the transient process.

If the applied demagnetizing field is then reduced, the magnetization in the material will now follow a straight line reversible magnetization curve whose slope is given by the recoil (relative) permeability defined by the user. The value is defined in the material data for the magnet using the parameter **MU**.

### Running a model without a DEMAG simulation

Permanent magnets are often supplied by the manufacturer already magnetized. In simulation terms, this means that they are at the state they would be at the end of a Magnetization simulation.

Consequently, if the user wants to simulate demagnetization in service for one of these models using a transient simulation, it is necessary to load sufficient data into the model to correspond to the data that the magnetization solver would supply.

The material model requires a table of demagnetization curves. The manufacturer of the permanent magnet has usually supplied only a single demagnetization curve (although there may be different curves for different temperatures). To assemble a suitable material table from this data, the table should have two sets of data for each temperature - one for the **B<sub>max</sub>** value corresponding to where the manufacturer supplied curve would intersect the virgin curve and the other for **B<sub>max</sub> = 0**. The

magnetization will also be zero for this second set of data and the same values of **B** can be used as for the first set of data in the table. An example is shown below

```
18 2 3 1
1 B [FLUXU]
2 BMAX [FLUXU]
3 T [1.0]
4 M_PM1 [FLUXU]
0
-34440 55700 20 -2130
-29690 55700 20 1260
-25670 55700 20 3850
-22120 55700 20 5710
-19120 55700 20 6900
-16350 55700 20 7750
-13660 55700 20 8470
-10990 55700 20 9130
-8370 55700 20 9720
-5960 55700 20 10120
-3740 55700 20 10320
-1430 55700 20 10490
810 55700 20 10630
3080 55700 20 10760
7450 55700 20 11000
11530 55700 20 11120
36120 55700 20 11120
55700 55700 20 11120
-34440 0 20 0
-29690 0 20 0
-25670 0 20 0
-22120 0 20 0
-19120 0 20 0
-16350 0 20 0
-13660 0 20 0
-10990 0 20 0
-8370 0 20 0
-5960 0 20 0
-3740 0 20 0
-1430 0 20 0
810 0 20 0
3080 0 20 0
7450 0 20 0
11530 0 20 0
36120 0 20 0
55700 0 20 0
-10710 55700 150 7310
-9300 55700 150 7990
-7490 55700 150 8680
etc.
```

A virgin curve for the material is generated internally from an interpolation of the **table** file. The user does not have to provide a BH data file.

It is necessary to insert the tables of **RMAGBX**, **RMAGBY** and **RMAGBZ** for all the elements in the permanent magnet. The manufacturer supplied permanent magnet can be assumed to be uniformly magnetized to saturation in the required direction. Consequently, the following stages are needed:

- Create a table file of the element centroids of the magnet in the Post-processor. Select the permanent magnet volumes of the model and use  **Create Table File > Element Centroids in Selected Volumes.**
- Using a text editor, create a second table file with the required values of **RMAGBX** etc at each point. For example, if the magnet had been assumed to be uniformly magnetized in the Y-direction, the values of **RMAGBX** and **RMAGBZ** can be set to zero, while **RMAGBY** is set to  $B_{max}$ .  
 **Process Table File > Import Fields at Subset of Elements.**

The demagnetization in service simulation can now be run.

# Using Hysteretic Materials

## Introduction

All ferromagnetic materials exhibit hysteresis in some degree. Frequently, it is a reasonable assumption that hysteresis can be ignored and that the nonlinear behaviour of a material can be characterized simply by its anhysteretic curve, because the hysteresis loop is narrow. However, with the increasing use of power electronics in many applications - notably the use of PWM - hysteresis is becoming more important. Continual switching of operating point on the magnetic characteristic results in repeated traversal of minor loops. Consequently, it is becoming particularly important to obtain an assessment of the losses due to hysteresis when rapid switching is occurring.

Opera-3d includes the ability to model hysteretic materials under transient conditions using a B(H) trajectory-following algorithm. Hysteretic materials can be used in conjunction with the Transient Electromagnetic, Transient Motion and Magnetization simulation modules<sup>1</sup>.

The user needs to supply data for only the major hysteresis loop. The algorithm uses a reconstruction technique to determine minor loops and turning points of the trajectory and to erase turning points when the magnetization of a material exceeds the previous excursion. The algorithm also correctly transfers to the saturated material curve beyond the end of the user data, in the same way as for anhysteretic materials in Opera-3d.

Figure 14.41 shows the characteristics of the data that must be supplied in the BH curve for the hysteretic material. Several points are worth noting:

- The curve represents one half of the complete hysteresis loop in the first, second and third quadrants. The software reconstructs the other half of the loop.
- The first point on the curve is in the third quadrant (at  $(-B_1, -H_1)$  say). The last point on the curve must be in the first quadrant at  $(B_1, H_1)$  with the same magnitudes of  $B$  and  $H$ .
- The point at  $B=0$  between the 2nd and 3rd quadrants must be included in the table (as this defines the coercive field strength of the material).

Other than these conditions, the requirements for the data are identical to those of other BH curves used by Opera-3d.

---

<sup>1</sup>A license for the Magnetization solver is required in order to use hysteretic materials in any of the other analysis programs.

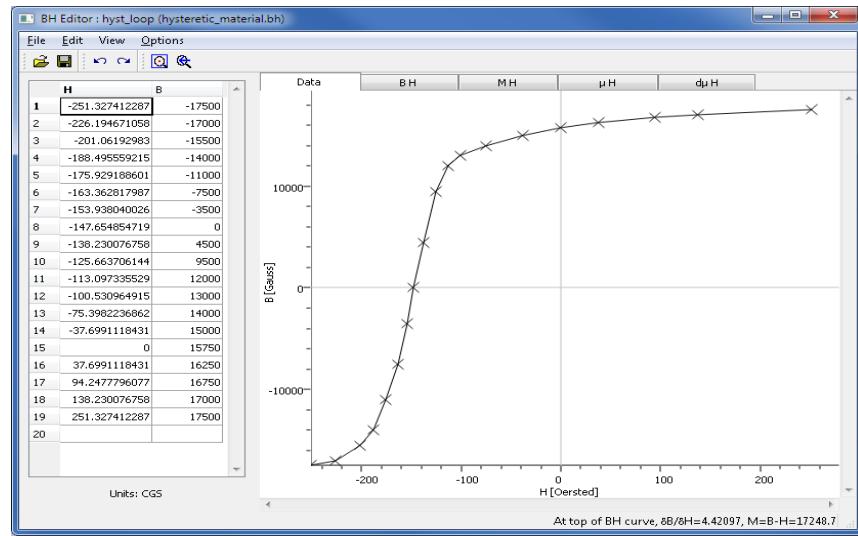


Figure 14.41 BH characteristic for hysteretic material

## Example: An H-frame dipole

To show the use of the hysteretic material option, an H-frame dipole magnet is subjected to a transient drive characteristic. Figure 14.42 shows the geometry of the magnet, while Figure 14.43 shows the transient drive characteristic specified through a timetable (**tt**) file. Note that the drive is a multiplying factor applied to the value of the defined source (a current density of 1800 amp/cm<sup>2</sup> in this case).

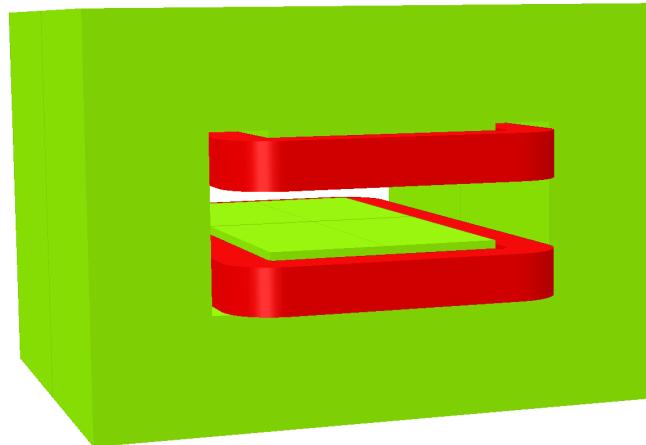


Figure 14.42 H-frame dipole magnet

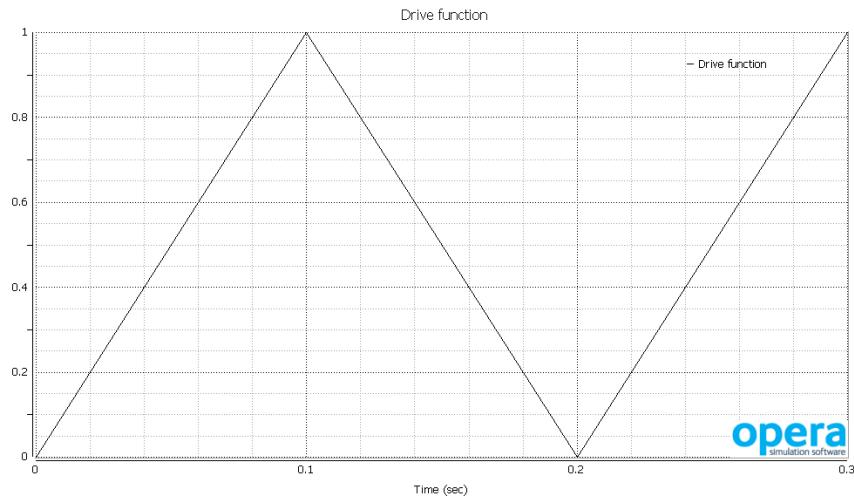


Figure 14.43 Drive function for transient analysis

Because of the symmetry of the magnet, it is sufficient to only solve 1/8 of the magnet using the Transient Electromagnetic solver, as shown in Figure 14.44.

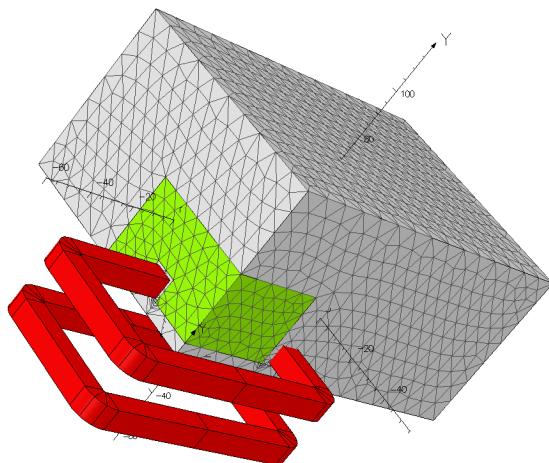


Figure 14.44 Symmetry section of the model for Transient Electromagnetic

When setting the material properties for the steel in the yoke, it is necessary to tell the analysis that

 **Set material properties** gives the dialog shown in Figure 14.45, in which the user must select **Hysteresis model** and give the **BH label** for the hysteresis curve. Note that the hysteresis model will be valid only if a nonlinear transient analysis is selected. The Modeller does not allow the **op3** database to be created if the setting is linear.

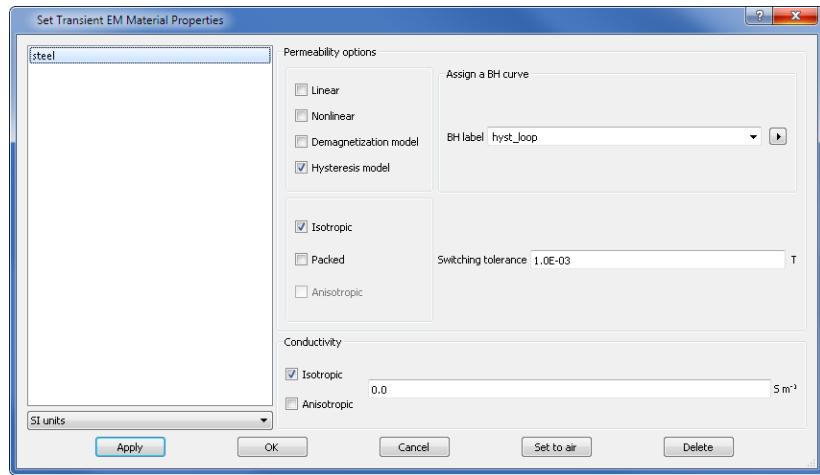


Figure 14.45 Materials dialog for hysteretic material model

## Results

The effect of using the hysteretic material characteristic can be clearly seen from the plot of the flux density at the centre of the air gap in the H-frame magnet, as shown in Figure 14.46.

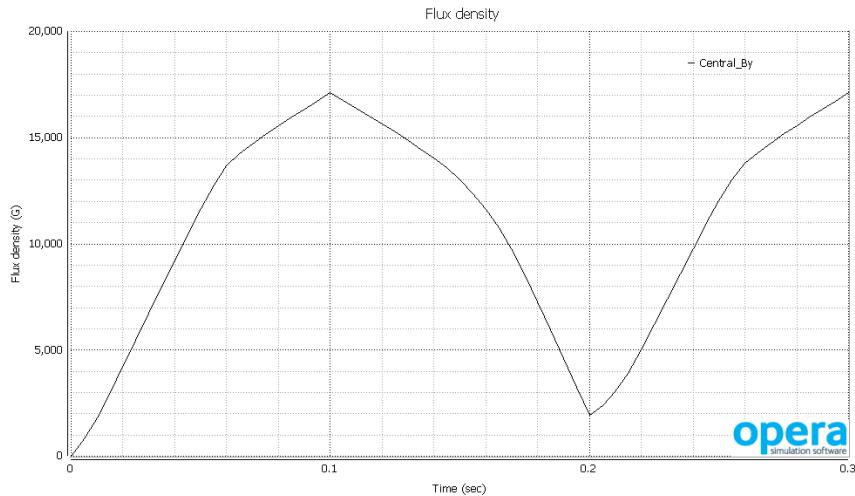
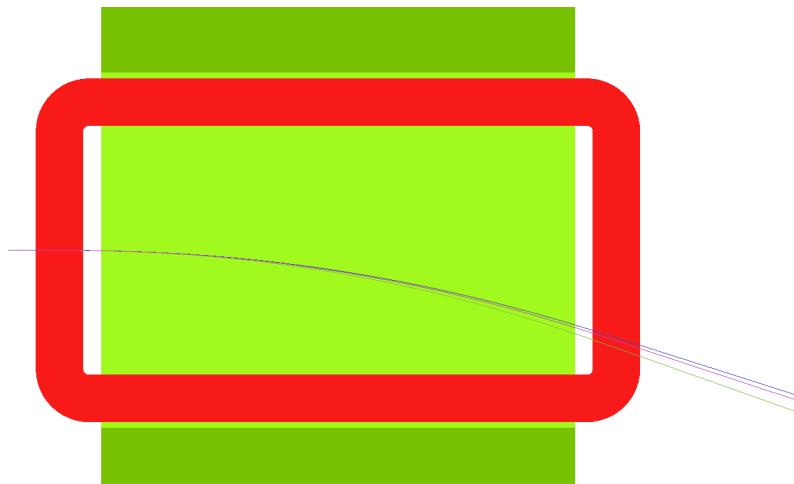


Figure 14.46 Flux density at centre of magnet

Comparison with Figure 14.43 shows that when the current has returned to zero at 0.2 seconds, the flux density in the gap is just over 1.9 kG. When the current is at 50% of its peak value (at 0.05, 0.15 and 0.25 seconds), the flux density has values 11.6, 13.0 and 12.0 kG respectively. However, when peak current has been applied (0.1 and 0.3 seconds), for both cases the central flux density is about 17.25 kG, because most of the material has gone beyond the end of the hysteresis loop and has transferred to the saturated material model.

The effect of the hysteresis can also be clearly seen when the trajectory of charged particles through the dipole field is calculated. Effects of hysteresis and remanent field in accelerator magnets has become an important topic, because the field strength in the aperture of a magnet dictates the trajectory followed by charged particles in the accelerator. If the trajectory cannot be predicted accurately, because hysteresis has not been included, magnet alignment becomes very difficult. [Figure 14.47](#) shows trajectories of a 1 GeV electron entering the magnet from the negative Z-direction for the 3 different 50% excitation cases. The blue track is at 0.05 seconds, when the current has first reached 50%; the green track is at 0.15 seconds, when the current has reduced to 50% following its peak at 0.1 seconds; the magenta track is at 0.25 seconds, when the current increases to its 50% value again, after being reduced to zero at 0.2 seconds.



*Figure 14.47 1 GeV electron tracks at 50% current level*

The same particle has also been tracked at time 0.2 seconds when the applied current is zero but the residual field still deflects the particle, as shown in [Figure 14.48](#).

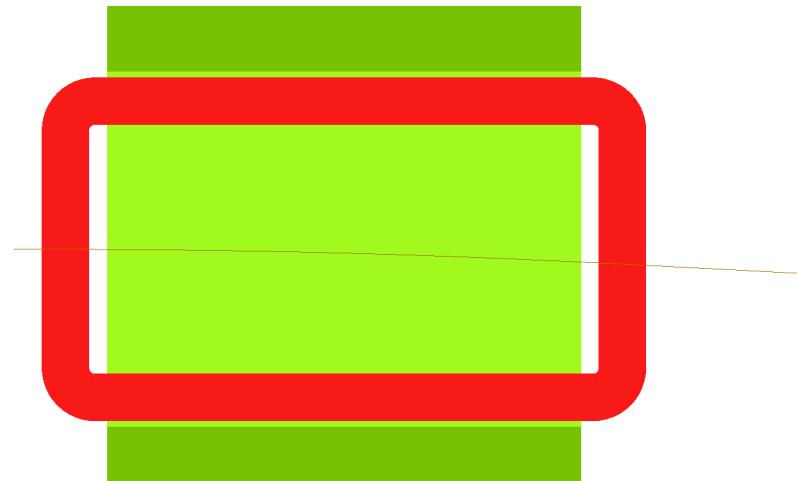


Figure 14.48 1 GeV particle track at 0.2 seconds

## Hysteresis losses

During the transient calculation, the loss due to hysteresis in the hysteretic material is calculated. At each output time, the energy density (the sum of the stored and dissipated energy since the beginning of the transient simulation) can be displayed. Values for each element are available as system variable **HLOSS**. This variable can be displayed in the same way as for example the magnetic flux density **BMOD**.

Figure 14.49 shows this energy density in the magnet at time 0.3 seconds.

The total power dissipated due to hysteresis and the change in the system stored energy can be found by taking the difference between the integral **HLOSS** over the volume of the material at two different times, and dividing by the time difference. Usually this calculation would be performed for material undergoing a repetitive cycle, such that the stored energy at a time interval of one cycle would be the same for both cases and the difference would be entirely power dissipation due to hysteresis.

In this model, due to the exaggerated loop (see Figure 14.41), the power dissipated due to hysteresis - computed between 0.1 and 0.3 seconds when the two values of total stored energy are almost identical - is about 69 kW. It is also possible to monitor the total energy for each material during analysis by including system variable **HLOSS\_name** as one of the variables recorded in the **log** file.

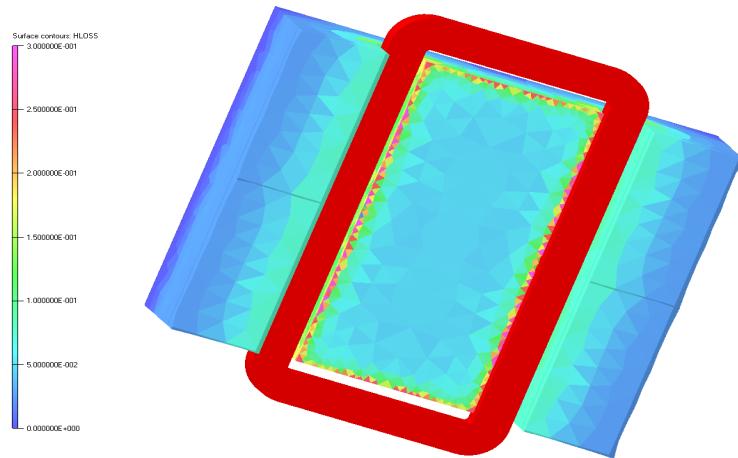


Figure 14.49 Stored and dissipated energy density after 0.3 seconds

## Boundary Conditions for Thin Gaps

### Introduction

Many electromagnetic devices and models contain thin gaps between components. These gaps may result as a consequence of the construction method or may be deliberately introduced to inhibit current flow from one component to the next.

Some examples of this are:

- Transformer yokes: In some 3-phase transformers, a laminated E-core is constructed and the pre-formed windings are added by sliding them onto the limbs of the E-core. The magnetic circuit is completed by a second laminated yoke that closes the open ends of the limbs. As far as the magnetic field is concerned, the gap between the two parts of the yoke is negligible. However, eddy currents induced in the plane of the laminations see the gap as a barrier that they cannot cross. This only slightly modifies the field, but changes the current distribution, and hence the losses, more significantly.
- Segmented surface mount permanent magnet machines: The "magnet" for a particular pole is often constructed from many small permanent magnets or segments. The segments have small gaps deliberately inserted between them to reduce the losses due to eddy currents.

Thin cracks can also occur due to a mechanical defect in a component. Electromagnetic non-destructive test (NDT) equipment may be used to find these cracks by detecting disruption to eddy currents induced in the surface of the component.

Creating a finite element model that explicitly incorporates these gaps or cracks as a volume of meshed space can be exacting - both in terms of model creation and accurate meshing. This application note introduces the user to a method available in Opera-3d which replaces the thin gap/crack by a boundary condition applied to a surface. The boundary condition inhibits the flow of current normally across the surface and is known as **ELECTRICINSULATOR**.

The boundary condition is available in the following analyses: Static Current Flow (but not if lossy dielectrics are used), Transient or Harmonic Electromagnetic, Transient Motion and Magnetization.

### Limitations

The Electric Insulator Boundary Condition (EIBC) represents an infinitely thin break between the materials. This break has zero conductivity. For a situation where the thin break (i.e. gap) is bridged with a low conductivity material as shown in [Figure 14.50](#) and [Figure 14.51](#), the final eddy current distribution can be different when comparing results from an EIBC model with results from the same model made with a finite gap.



*Figure 14.50 Effective behaviour of the EIBC*



*Figure 14.51 Situation with a real gap*

The EIBC works well when the surrounding material on the edges of the EIBC surface is (a) AIR or (b) similar conductivity to the material in which the EIBC has been applied.

However, if the electrical conductivity in the surrounding material is finite but much less than the conductivity of the other material, the answers will not mimic the same model made with a finite gap. The finite gap produces a substantial electrical resistance through the low conductivity material whereas the EIBC gives a value  $\rightarrow 0$ . Hence the eddy current distribution can be quite different.

## Specifying the Electric Insulator Boundary Condition

The boundary condition can be applied on any face that is shared by two different cells of a model. It cannot be applied to an internal face within a single cell and will be ignored.

If the user tries to use it on an internal face, a warning is issued that the boundary condition is not being enforced when the **op3** file is created.

#### ELECTRICINSULATOR

Warning: Defined on internal faces that do not separate cells. The boundary will have no effect.

The boundary condition may also be applied on an external face of a cell where a **Tangential magnetic / Normal electric** boundary condition is applied to all the other co-planar faces to imply symmetry in the model. The boundary condition enforces that currents cannot cross the symmetry plane at this face because there is a thin gap between the cell and its implied image. This will be illustrated in the example below.

## Segmented Surface Magnet Example

Figure 14.52 shows the rotor of a 4-pole surface mount permanent magnet rotor for a brushless motor. The surface mount magnet actually consists of 4 individual magnet segments in each pole separated by a very thin air space in the radial-azimuthal plane which inhibits flow of induced currents in the axial direction. At the gaps, eddy currents are forced to turn azimuthally thereby increasing the path length. The corresponding increased resistance reduces the eddy current density and, hence, the Ohmic loss.

In the Modeller, each magnet segment is constructed from a separate cell. The cells are touching where the thin air space exists - that is, the gap has been removed. On the common face between the cells, a boundary condition label insulation has been applied. Figure 14.53 shows the faces where the boundary condition is applied.

This boundary condition label is assigned to be an electric insulator in the **Model -> Set Boundary Conditions** dialog, as shown in Figure 14.54.

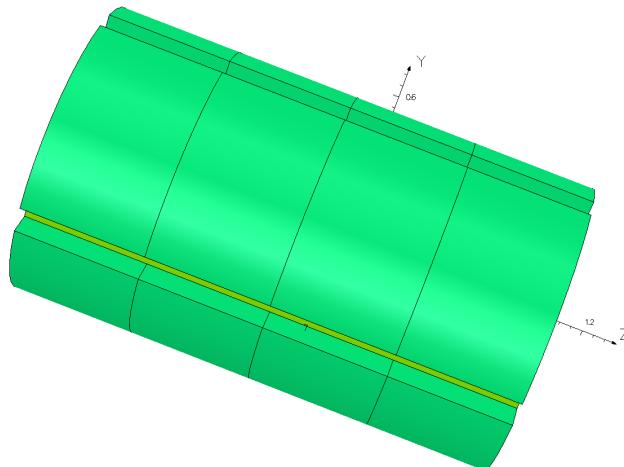


Figure 14.52 Geometry of rotor

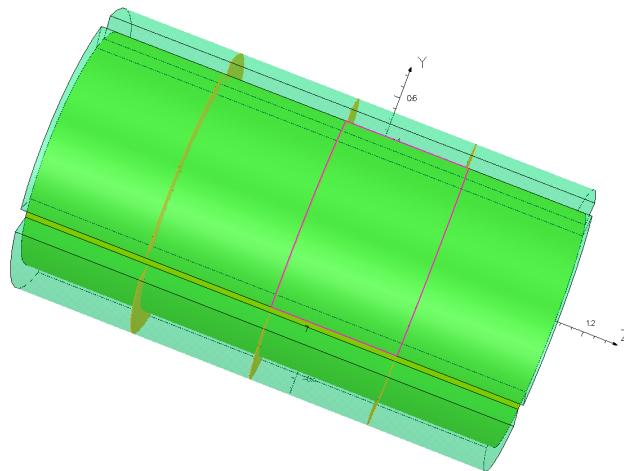


Figure 14.53 Model faces with the insulating boundary condition label applied

The rotor has been used in a Transient Motion simulation to calculate cogging torque. Symmetry in the model has been exploited, using 4-fold negative periodicity to reduce the model to a 90 degree section and only half the axial length. On the  $Z = 0$  plane, the magnetic field has been assigned to be only tangential using the **Model Symmetry** dialog. As can be seen in Figure 14.55, this is also a plane where the **Electric insulator** boundary condition is applied (shown in red). The boundary condition implies a gap between the magnet in the model space and the magnet in the image of the model space implied by the symmetry.

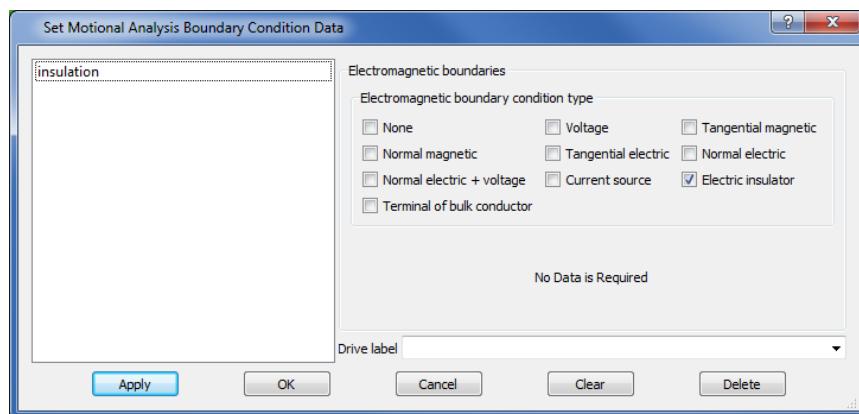


Figure 14.54 Boundary condition dialog

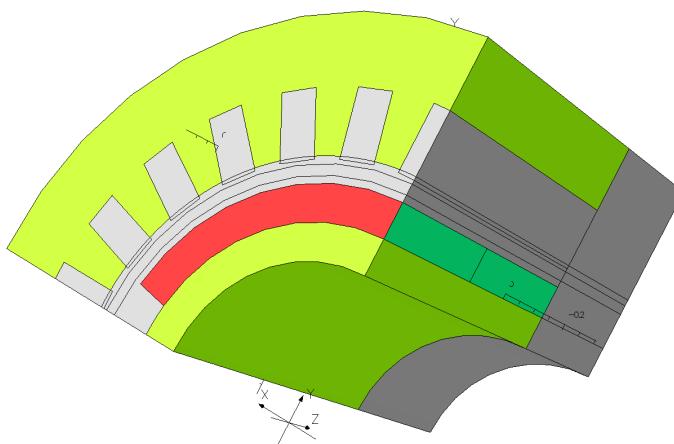


Figure 14.55 Model for Transient Motion simulation

The change in magnetic flux caused by the teeth as the rotor turns will induce eddy currents in the conducting magnets. These can be seen for a particular instance in Figure 14.56.

As can be seen, the eddy currents turn at the end of each magnet segment as the boundary condition prohibits them crossing to the neighbouring segment. This includes the symmetry plane at  $Z = 0$  where the **Tangential magnetic** boundary condition would otherwise imply that the current is only normal to the plane and that the normal component of current density is continuous.

The same machine has also been modelled with explicit spaces between the magnet segments. Because of the simplicity of this example, it is easy to produce a volume mesh that includes the thin gaps. The eddy current pattern induced in the magnets is very similar to that produced with the **Electric insulator** boundary condition and the resulting Ohmic losses agree within a few per cent.

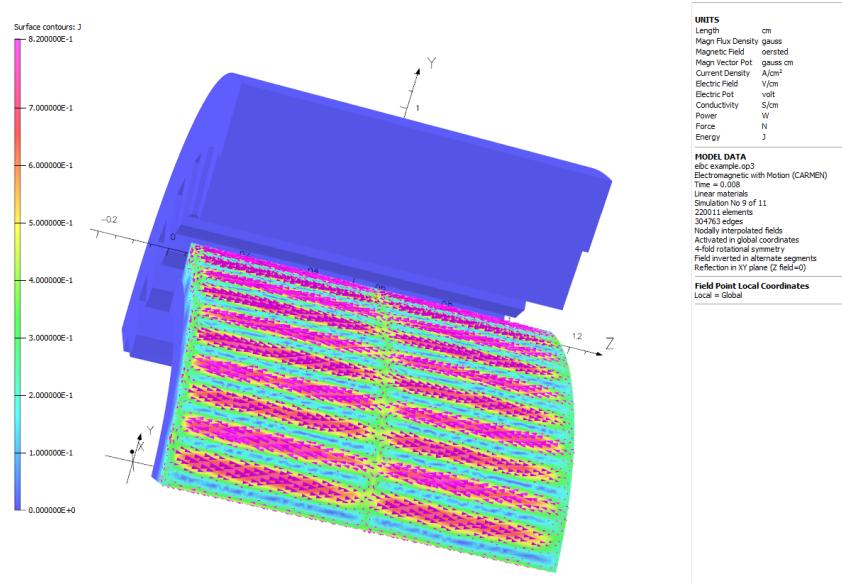


Figure 14.56 Eddy currents induced in magnet segments

## Thermal Contact Boundary Conditions

Thermal simulations also present comparable physical situations that require modelling. Sometimes electrically conducting materials are separated by a thin electrically insulating material which prevents current crossing between the materials but allows heat to pass. However, the insulating material has a low thermal conductivity and must be included in the thermal simulation to obtain realistic results. Similarly, bonding between two materials will also only be a very thin layer but have a low thermal conductivity. Equally, in many cases, two materials that are theoretically in perfect contact only touch imperfectly due to surface finish. This also presents resistance to heat transferring from one material to the other.

In Thermal and Quench simulations, it is now possible to specify a **Thermal contact** boundary condition. As with the **Electrical insulator** boundary condition, the thermal boundary condition must also be applied on a face between two cells and will be ignored on internal faces (i.e. "dangling" faces). A **Thermal contact** boundary condition also requires additional information. The dialog requires the user to give a value for the **Thermal contact conductance**, in units of W/(K length\_unit<sup>2</sup>), as shown in the dialog (Figure 14.57).

Figure 14.58 shows the temperature distribution in a model of a heated steel sphere in contact with a plastic block. The imperfect contact between the sphere and the plastic is represented using a thermal contact boundary condition. The resulting discontinuity in temperature at the boundary can be clearly seen.

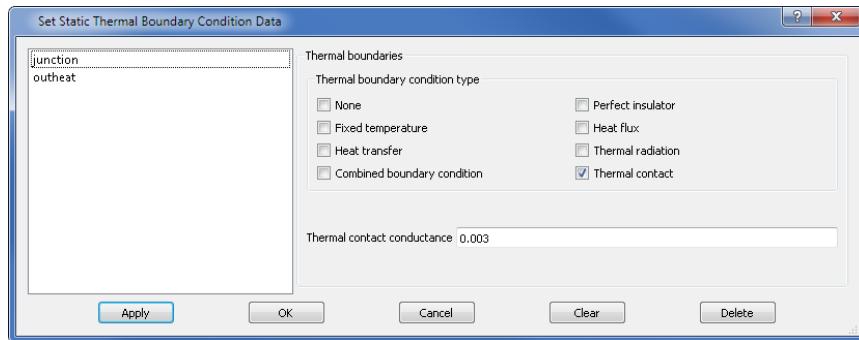


Figure 14.57 Boundary condition dialog for thermal models

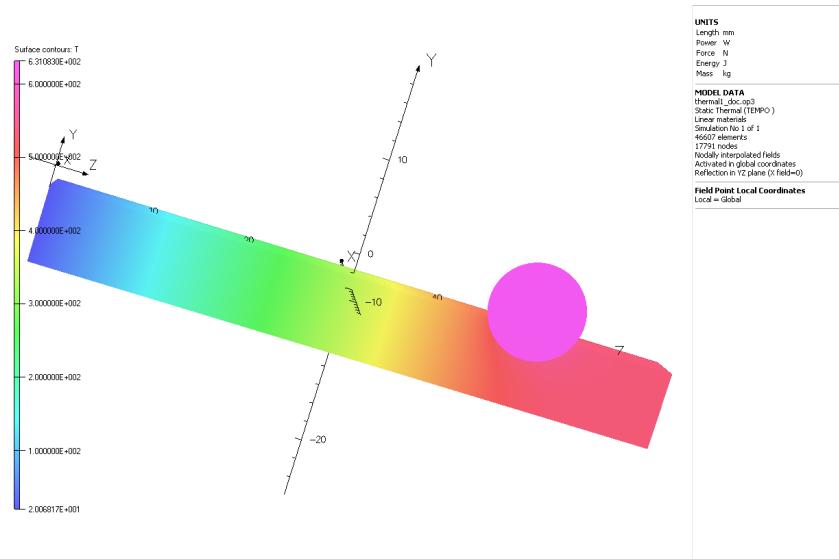


Figure 14.58 Temperature distribution with *Thermal contact* boundary condition

## Using Surface Impedance Boundary Conditions in Harmonic Electromagnetic

---

**Harmonic Electromagnetic** simulations calculate the electromagnetic field at a user defined frequency. Because the field is changing sinusoidally in time, eddy currents are induced in conducting media. The fields and eddy currents decay as they penetrate the conducting media at a rate which depends on the material properties and the frequency. Materials that have higher electrical conductivity exhibit greater attenuation of the field and eddy currents than lower conductivity materials. Magnetic materials also exhibit greater attenuation than non-magnetic (although this is also affected by magnetic saturation and conductivity). Higher frequency fields also penetrate significantly to less distance.

To capture the penetration of the fields and induced currents, it is necessary to use sufficient finite elements to represent the decay. As discussed in [Building layers \[page 276\]](#) of the Induction Heating worked example, a good way to assess the necessary discretization is to use one dimensional skin-effect theory. From a pragmatic viewpoint, using a minimum of 2 layers of finite elements in the first skin depth and 1 layer in the next skin depth will capture skin-effect sufficiently for reliable values of power dissipation etc.

However, in some applications, imposing such a mesh to capture skin-effect becomes very demanding. For example, the skin depth of copper at 10 kHz is about 0.5 mm, requiring that the first 2 layers are only 0.25 mm deep. The alternative is to impose a boundary condition at the surface of the conducting media which represents the decay of the field and eddy currents into the media. With this boundary condition, the software no longer needs to solve Maxwell's equations inside the media and building a suitable mesh to capture skin-effect is not required. The technique in Opera-3d **Harmonic Electromagnetic** simulation that achieves this is called a Surface Impedance Boundary Condition (usually abbreviated to SIBC). Opera's SIBC also relies on one dimensional skin-effect theory and is described in detail in the [Opera-3d Reference Manual](#). The limitations of the SIBC are also explained in the same document.

### Building a model with an SIBC

**Harmonic Electromagnetic** models using SIBC are built exactly the same way as models where the conducting media is discretized. The only difference is that the properties for a material label are specified to use the SIBC, as shown in [Figure 14.59](#).

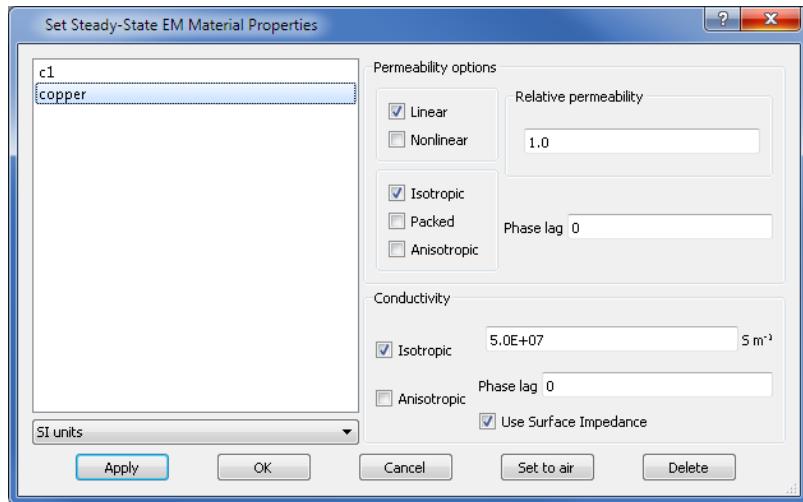


Figure 14.59 Material dialog with SIBC selected

Figure 14.60 shows a simple model where SIBC has been used.

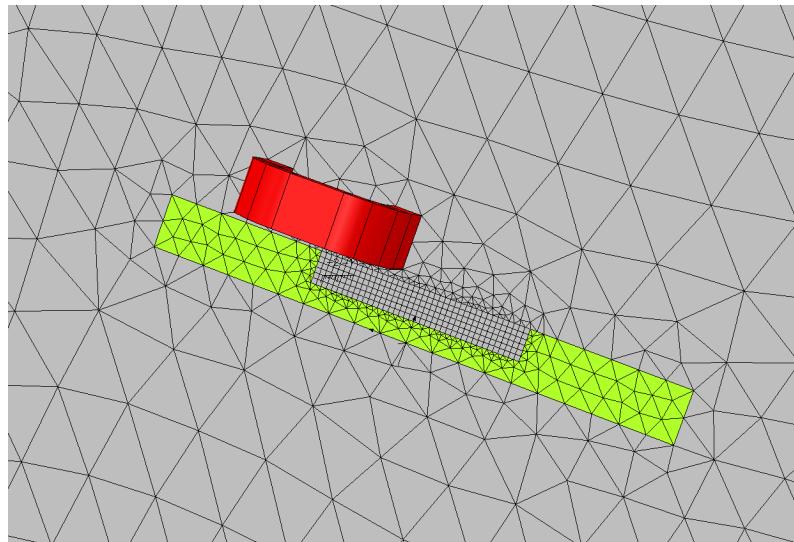


Figure 14.60 Model of NDT test

This is a typical NDT test scenario with a volume meshed coil connected to a 10 kHz voltage supply through a resistor. The 10 mm deep copper plate has a 1 mm wide x 40 mm long x 6 mm deep slit cut in it. The coil is positioned symmetrically such that the simulation may be confined to half of the model. As can be seen, the mesh in the copper is not adequate to model skin-effect. However, using the SIBC for the copper gives the expected result, as shown in Figure 14.61.

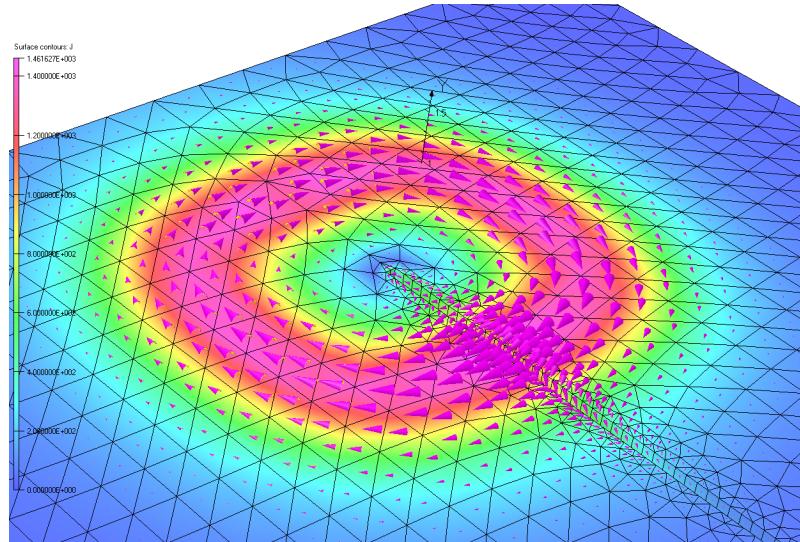


Figure 14.61 Real part of current density in copper

## Post-processing

Since the **Harmonic Electromagnetic** solver is solving for the vector potential at only the surface of materials modelled with SIBC, the solution inside the material volume is identically zero. If the user displays the field or currents inside the material, it will not be an accurate representation of the decay into the material. However, the Opera-3d Post-Processor can still perform correct calculations of global values such as **Energy**, **Power and Force** . The software assumes the same 1-dimensional skin-effect theory to evaluate the necessary field and current density values.

## Using Surface Impedance Boundary Conditions in Transient Electromagnetic

---

Surface impedance boundary conditions (SIBC) are available in all transient electromagnetic simulations and, similarly to Harmonic Electromagnetic, they are specified as a material property. They also serve the same purpose as in Harmonic simulations:

- They can be used when the skin depth is much smaller than the physical depth of the electrically conducting media which the fields and eddy currents are trying to penetrate and, hence, can be approximated accurately by one dimensional skin theory
- They avoid the use of complex layering of finite elements in the material by actual elements
- They should be used in simulations where the field and currents can be considered as time periodic. This may be a simple sine wave (time harmonic) or a periodic waveform containing multiple harmonics, such as a square wave or a saw tooth wave

They also have a further advantage that they can approximate the correct behaviour of non-linear magnetic materials by modifying the assumed skin depth during non-linear convergence, as opposed to Harmonic EM SIBC which is limited to linear materials.

There are a number of limitation when using transient SIBC which will be illustrated by the examples in the following sections and the description of SIBC in the ***Opera-3d Reference Manual***. However, in summary, these limitations are:

- The behaviour of the system when the applied field is (or is tending towards) DC will not be correctly modelled. Fields will penetrate the media once eddy currents in the media have decayed, but the SIBC will not model these fields.
- Two materials in electrical contact with different electromagnetic properties cannot both use SIBC.
- An SIBC material in electrical contact with a meshed electrically conducting material is allowed, but the SIBC approximation is not valid if there is a substantial normal component of induced current to the interface between the two materials.

### Setting up a Transient EM model with an SIBC material

As mentioned in the previous section, SIBC for Transient EM is specified as a material property. To use SIBC, the user must first select the tick box that switches it on (see the dialog shown in [Figure 14.62](#)) – similarly to Harmonic EM.

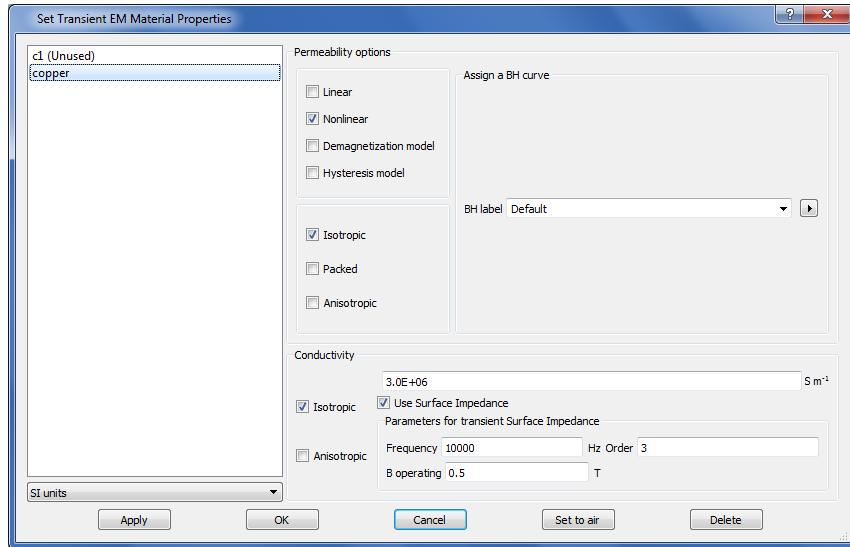


Figure 14.62 Material dialog with SIBC selected

Transient EM SIBC also requires a number of other parameters:

- Characteristic **Frequency** of the periodic function – usually the fundamental frequency of the drive field or currents in a model. A square wave with a period of 1 millisecond would have a characteristic frequency of 1 kHz for example.
- **Order** of 1-dimensional model. Transient simulations will usually be used when non-linear materials significantly affect the field, or the periodic function is non-sinusoidal. In either case, the field and current waveforms will contain harmonic components of the characteristic frequency. These higher order harmonic components can be captured by the SIBC by increasing the **Order** up to a value of 5. An order of 2 will include 3rd harmonic components, while order 3 adds the 5th harmonic etc. Order 1 is sufficient for purely sinusoidal fields / currents and linear materials.
- **B Operating** is the user's estimate of the surface flux density in the magnetic medium. Specifying this value is not mandatory but an intelligent estimate will improve the non-linear convergence.

The example model in the Application Note "[Using Surface Impedance Boundary Conditions in Harmonic Electromagnetic](#)" on page 610 has also been modelled using the Transient EM SIBC approximation with a 10 kHz sinusoidal drive for the coil. However, the plate is now constructed from steel using the **Default** BH curve supplied with Opera-3d. Consequently, order 3 has been chosen and the estimate for the flux density is 0.5 T. [Figure 14.63](#) shows the induced current after 200 microseconds.

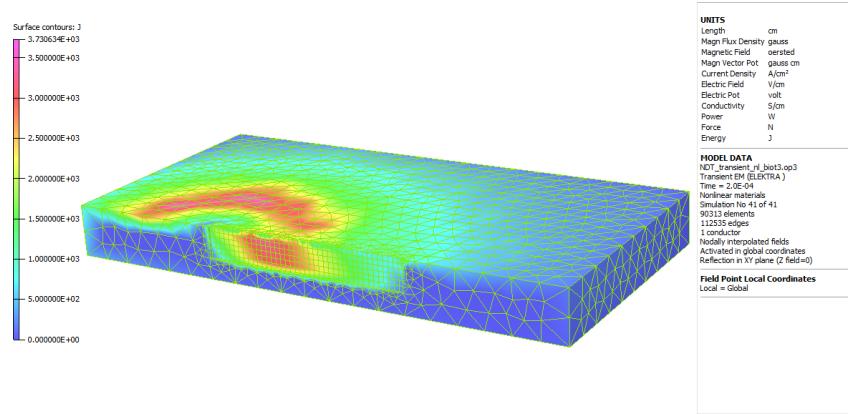


Figure 14.63 Current induced in the plate after 200 $\mu$ s

Opera-3d/Post-Processor also allows the total loss in a material represented by an SIBC in Transient EM simulations to be calculated – in the same way as for Harmonic EM SIBC. But, similarly, the loss intensity distribution displayed on the surface of materials is only valid at the surface and the decay into the material is not displayed correctly.

## Limitation of SIBC with "DC" fields

As mentioned previously, SIBC approximations work because one dimensional skin effect theory is sufficient to represent penetration of the field into the material. When a field tends towards DC, this assumption can no longer be considered valid. However, Opera-3d Transient EM simulations will assume that, once a material has been specified as SIBC, the approximation will be true throughout the simulation.

This can be illustrated simply by changing the drive function for the coil current in the previous example from a sine to a cosine function. A cosine function with phase angle zero is at its peak value at time = 0. Since this is the start of the transient, Opera-3d assumes that this peak current value must have been supplied to the coil as a DC value from time = - $\infty$ . If the plate is meshed with volume elements instead of SIBC being used, it can be seen that the field penetrates the plate at time = 0 (as shown in Figure 14.64). However, when SIBC is applied, the field is prevented from entering the plate and an incorrect answer is obtained (see Figure 14.65).

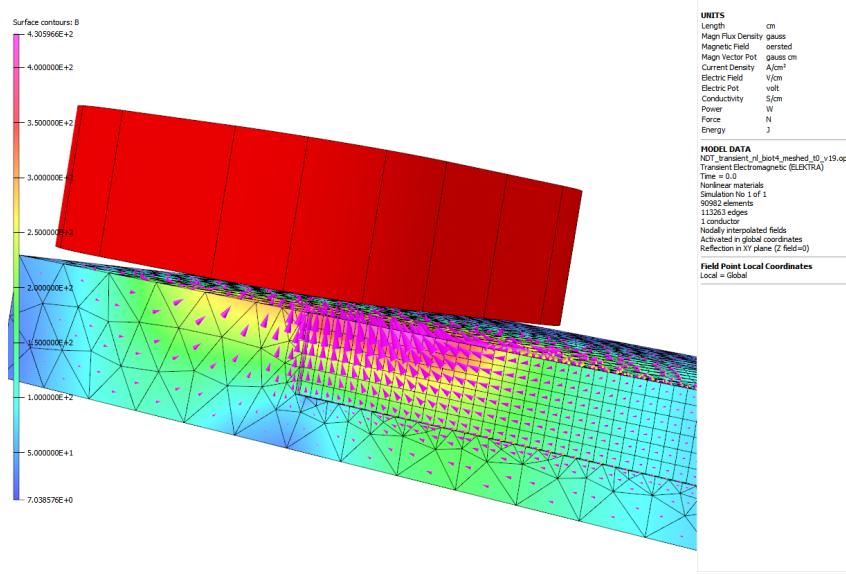


Figure 14.64 Field penetrating into a meshed plate

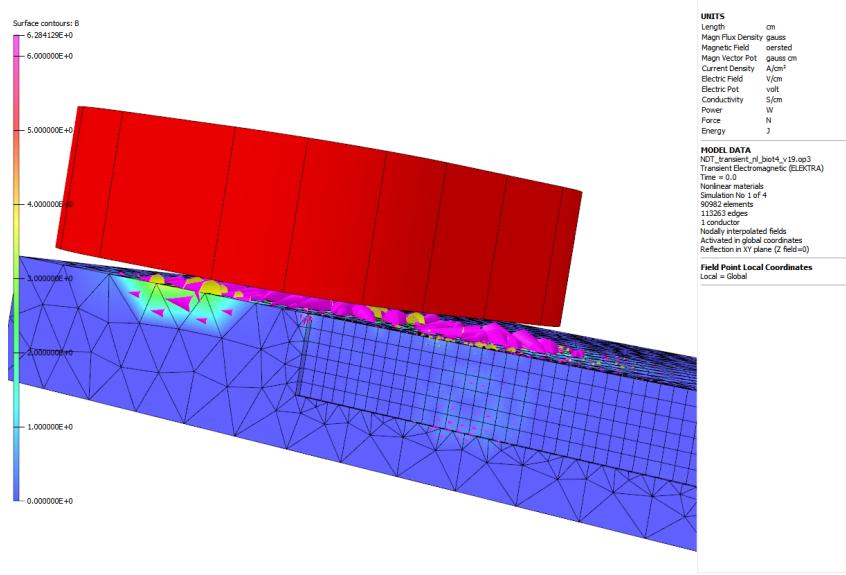


Figure 14.65 Field penetratring into an SIBC plate

Of course, once the periodic function drive starts to operate during the actual transient simulation, the error associated with the non-physical initial time = 0 solution for the cosine drive will gradually decay and results from the two solutions will converge (apart from having a 90 degree phase difference). This can be seen in Figure 14.66, which plots the loss in the plate as a function of time. Note that the loss will oscillate at 20 kHz around a DC value (the time average loss) – so the phase shift in the loss is 180 degrees.

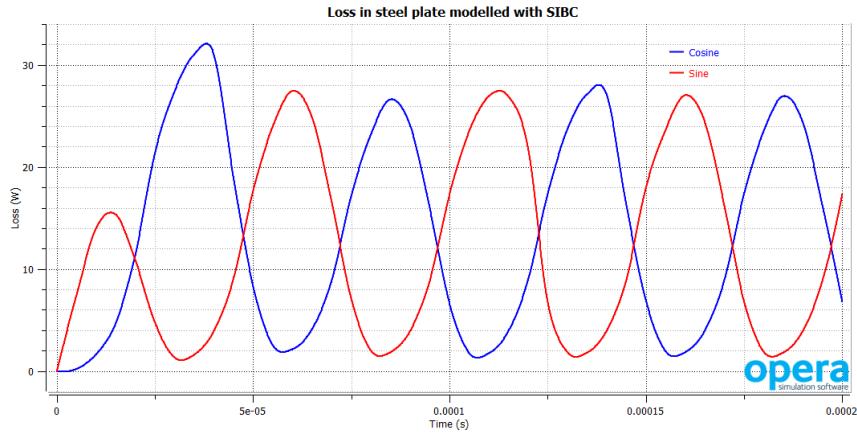


Figure 14.66 Plate loss against time for drives starting at zero and non-zero values

## Limitations with neighbouring electrically conducting regions

### Two or more touching materials both using SIBC

Figure 14.67 shows a very simple model of two electrically conducting blocks made from different materials, with electrical contact on the interface between them.

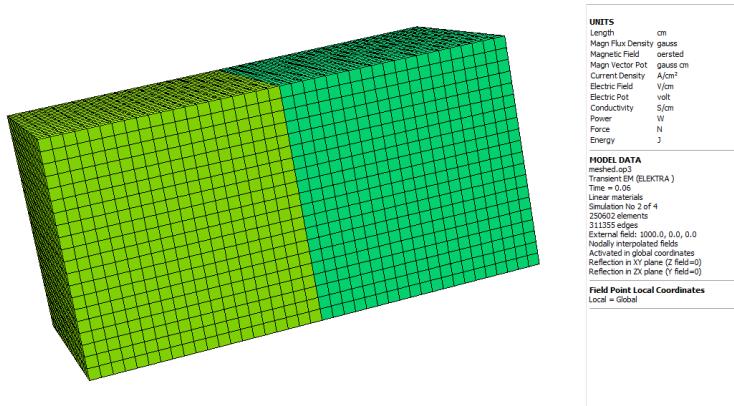


Figure 14.67 Two touching SIBC blocks

If the user attempts to run a simulation with touching SIBC materials, the simulation fails almost immediately with the error message:

```
Error level: 1411201006
Two neighbouring SIBCs have different material properties
```

However, if the different material labels in the model actually hold the same properties (permeability, conductivity and SIBC parameters), the analysis proceeds successfully.

## Two or more touching conducting materials where SIBC materials do not touch

An SIBC material can be in electrical contact with another electrically conducting region (or regions) that uses a volume mesh in the material – that is, the skin depth is sufficiently large that skin effect can be captured by the discretization.

When this occurs, the user receives a warning during database creation:

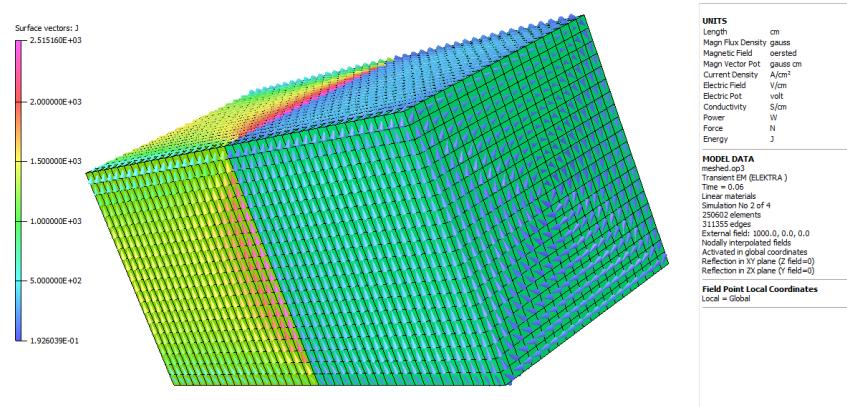
```
WARNING: there are adjacent conducting materials with either or both using SIBC.
SIBC cannot model currents crossing the interface between these materials.
```

SIBC theory assumes that all current on the face is tangential and, consequently, on the same face there cannot be any normal current crossing from the neighbour meshed region. In some modelling situations, this is a valid assumption while in others incorrect answers will be obtained.

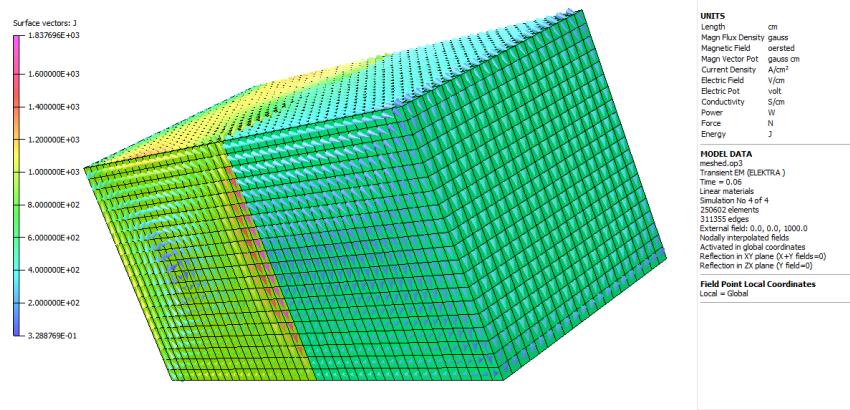
This can be illustrated using the same 2 block example shown in [Figure 14.67](#). The right block now has material properties that allow a meshed volume representation while the left block continues to be modelled using SIBC.

The model is excited by an applied uniform AC external field. In one case, the applied field is in the normal direction to the interface between the blocks (horizontal right-to-left), which makes the eddy currents flow predominantly around each block and not from block to block, as shown in [Figure 14.68](#).

In the second case, the field is applied normal to the front faces of the blocks (tangential to the interface) making the eddy currents circulate such that some of the induced current does want to pass normally from block to block, as shown in [Figure 14.69](#).



*Figure 14.68 Eddy currents from a normal incident field*



*Figure 14.69 Eddy currents from a tangential incident field*

Because of the simple nature of this example, it is also possible to solve the model with both blocks actually meshed – removing the SIBC boundary condition. **Table 142 below** compares the results for the power dissipated in the blocks at a particular time in the transient. As can be seen, in case 1, the agreement between the results is good and the SIBC condition in block 2 only overestimates the loss in that block by less than 5%. This is associated with the one dimensional skin effect theory not being correct at corners. However, in case 2, the SIBC condition in block 2 underestimates the loss in that block by nearly 30% because it has not allowed any normal current to enter block 2 from block 1.

**Table 142: Comparison of losses in 2 block model**

	<b>Both blocks meshed</b>		<b>Block 2 using SIBC</b>	
Case	Power in block 1 (W)	Power in block 2 (W)	Power in block 1 (W)	Power in block 2 (W)
1	10.04	3.30	10.00	3.45
2	4.68	6.34	4.90	4.54

### Two or more touching SIBC regions separated by an electrically insulating boundary condition

The two block example model has been modified to inhibit normal current flow between the blocks by inclusion of an electrically insulating boundary condition (EIBC) on the surface. As shown in **Table 143 below**, there is now good comparison for the loss in block 2 when the field is applied normal to the front faces of the blocks (as in case 2 above).

**Table 143: Comparison of losses in 2 block model with EIBC**

	<b>Both blocks meshed</b>		<b>Block 2 using SIBC</b>	
Case	Power in block 1 (W)	Power in block 2 (W)	Power in block 1 (W)	Power in block 2 (W)
3	4.89	4.39	4.90	4.54

As would be expected in this simple case, the losses in the blocks when the EIBC and SIBC are included (case 3) are identical to the incorrect values in case 2 when the SIBC was applied – precisely because the SIBC was preventing normal current crossing the interface, which is also the function of an EIBC.

## Using Surface Impedance Boundary Conditions in Multiphysics (Electromagnetic and Thermal) Simulations

Before working through this application note, it is recommended that the user reads the related document "**Using Surface Impedance Boundary Conditions in Harmonic Electromagnetic**" to become familiar with the concept of SIBC. This note shows how SIBC could be used in a multiphysics analysis, namely in a **Harmonic Electromagnetic** followed by **Static Thermal** analysis.

Figure 14.70 shows one half of the model that consists of a (meshed) conductor positioned above the nickel plate. The air background is hidden for clarity. Nickel was chosen for this demonstration because, at the frequency of 10 kHz applied to the current in the coil, the skin depth is only about 50  $\mu\text{m}$ , and is, consequently, valid for using SIBC. To analyze this model without using SIBC the user would need to provide a minimum of 2 layers of finite elements in the first skin depth and 1 layer in the next skin depth. Hence a very tiny mesh of elements would be required. However, a thin, 250  $\mu\text{m}$  deep, top layer from the main nickel block, highlighted in Figure 14.70, is created. In the **Harmonic Electromagnetic** model the mesh is not used in the SIBC material, but the model must be suitably discretized for the thermal analysis. The thermal model assumes that all power losses occur in this layer and no losses occur in the other part of the block. There is no need to use a meshed conductor in this analysis, but it has the advantage of offering an opportunity of calculating some parameters of interest on the surface or in the volume of the conductor, if necessary.

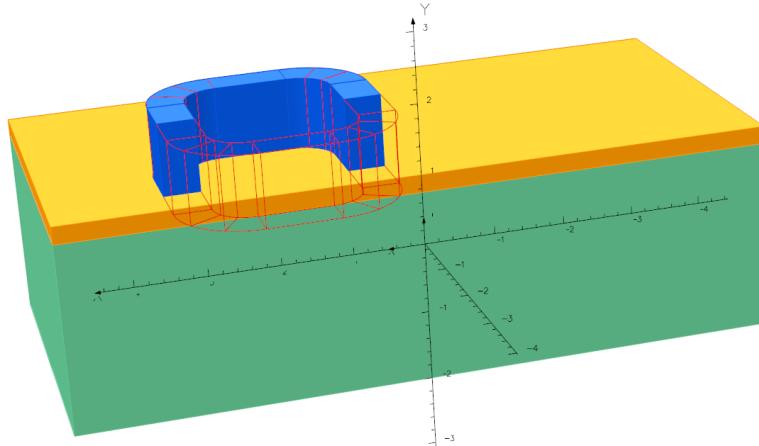


Figure 14.70 Model of an induction heated nickel plate.

The multiphysics analysis is set up by adding one after another **Harmonic Electromagnetic** and **Static Thermal** solvers from the **Analysis Settings** dialog. In the **Harmonic Electromagnetic** analysis, the **Analysis Frequencies** is set at 10000, and the other settings are left unchanged. The **Static Thermal** settings have not been changed from the default values.

The dimensions of the nickel block are 10 mm x 10 mm x 3 mm. For both cells (top and bottom of the nickel block) **Hexahedral or Prismatic** for **Element Shape preference** was selected and a **Maximum element size** of 0.25 mm was chosen. Use of hexahedral or prismatic elements in the top layer gives more accurate results and also improves visualization of results.

In **Material properties** SIBC should be enabled by selecting **nickel** as a material and ticking **use Surface Impedance**. Other properties of nickel are as shown in [Table 144](#).

**Table 144: Model settings for the multiphysics analysis**

<b>Harmonic Electromagnetic Analysis</b>		<b>Thermal Analysis</b>	
Parameter	Value	Parameter	Value
Current frequency	10 , 000 Hz	Material	Nickel
Current density in the conductor	10 , 000 A/mm <sup>2</sup>	Thermal Conductivity	90 . 9 W/(m.K)
Electrical conductivity of Nickel	1 . 43e7 S/m	Heat Transfer Coefficient (applied to side and top faces of the block)	0 . 01 W/(mm <sup>2</sup> .K)
Relative permeability of Nickel	600	Fixed Temperature (applied to the bottom face of the block)	20°C

As can be seen from [Figure 14.71](#), the mesh in the nickel is quite coarse and the thickness of mesh elements is about 5 times bigger than the skin depth of nickel at the chosen frequency of 10 kHz. This mesh would not be adequate to model the skin-effect without the SIBC. However, using the SIBC gives the expected result, as shown in [Figure 14.72](#). Note that in SIBC models fields are only calculated on the surface. Any fields displayed in the interior, [Figure 14.72](#), are the result of interpolation of the field within the first layer of mesh elements from the surface.

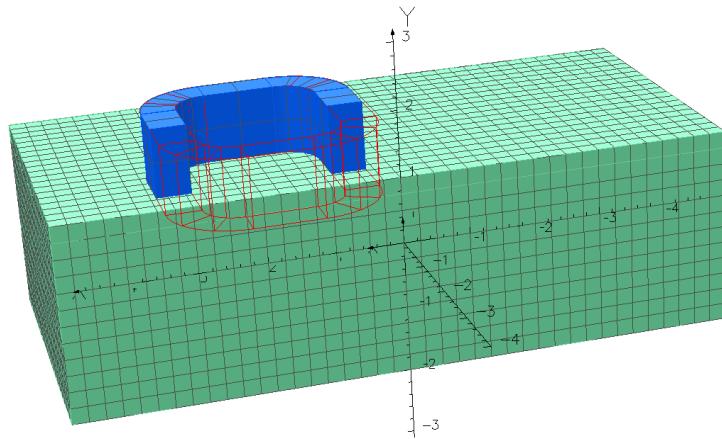


Figure 14.71 Meshed model

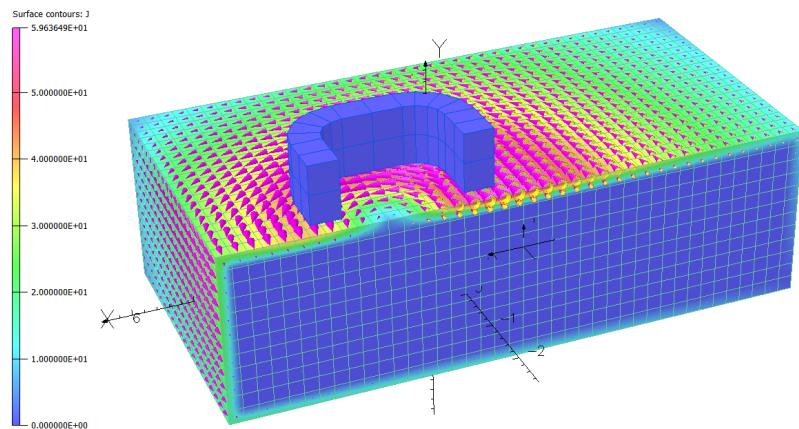


Figure 14.72 Current density in the nickel plate

Losses calculated in the **Harmonic Electromagnetic** analysis are used in the **Static Thermal** analysis to calculate temperature distribution in the nickel plate. Note that the same mesh is used for both analyses and data exchange (of the loss intensity) occurs automatically without the use of table files. The results of the **Static Thermal** analysis are shown in [Figure 14.73](#).

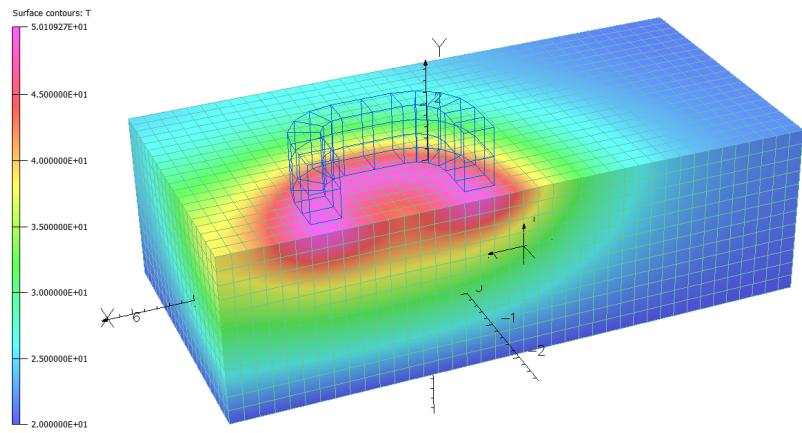


Figure 14.73 Temperature distribution in nickel

# Thin Plate Approximation for Magnetostatic Simulations

## Introduction

Many electromagnetic simulations include ferrous geometry with extreme aspect ratios. A good example of this is magnetic signature calculations of naval vessels where the hull, deck and bulkhead plates are several metres wide but only a few centimetres thick. Opera-3d has several useful features that enable these structures to be meshed with volume elements which also have high aspect ratio, such as layering and hexahedral / prism element meshing. While these are very effective for isolated plates, it can become difficult to build economical meshes when several intersecting plates are present – such as at the junction of a deck and bulkhead with the hull of a ship.

A method to overcome this is to replace the three dimensional volume structure of the plate with a two dimensional surface representation. Opera-3d achieves this in magnetostatic simulations using the **Thin plate** boundary condition.

## Example ship magnetic signature model

Figure 14.74 shows the hull and superstructure of a typical model of a ship used to determine its magnetic signature in the presence of the Earth's field. Figure 14.75 shows the internal structure of decks and bulkheads.

In this model, the thin plates making up these components are represented entirely by **Thin plate** boundary conditions. These are created in two stages (similar to all other boundary conditions). A **Boundary condition label** is applied to the face of a cell and the label is then associated with the **Thin plate** boundary condition. Figure 14.76 and Figure 14.77 show an example of this. In Figure 14.76, a face of one (or faces of many) of the AIR cells inside the hull has been selected and `bulkhead_015` is applied as the **Boundary condition label**. In Figure 14.77, the `bulkhead_015` label is associated with the **Thin plate** condition. As can be seen, there are two parameters associated with the condition: the material that the thin plate is constructed from and the thickness of the plate (in the direction which has been removed by replacing the volume with a surface). The **Material name** is a label for a material, which can also be used for a volume material, and the thickness is given in the units which will be used when creating the `.op3` file. As with volume materials, the **Material name** is also associated with a particular set of material properties, including non-linear magnetic characteristics, using the **Material properties** dialog.

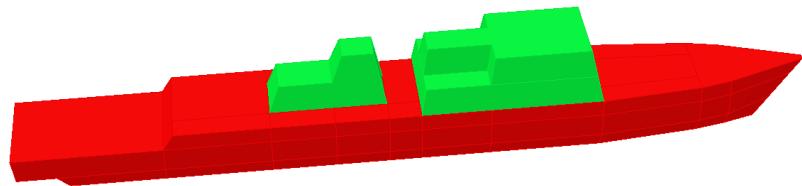


Figure 14.74 Hull and superstructure

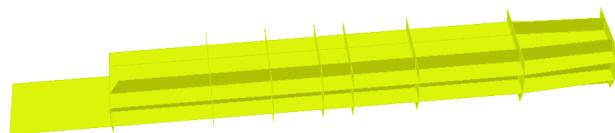


Figure 14.75 Decks and bulkheads

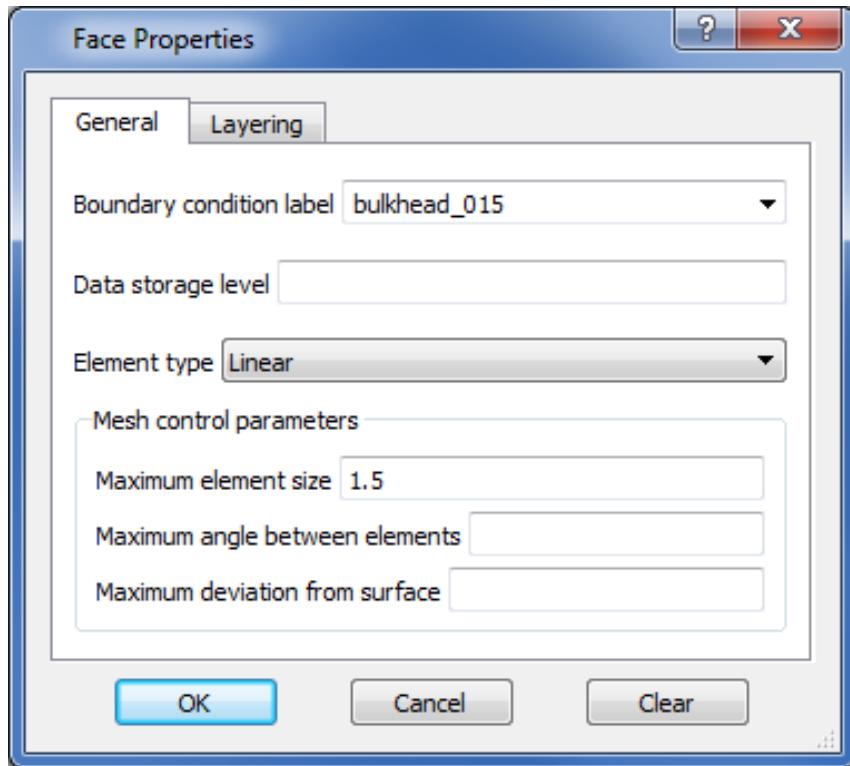


Figure 14.76 Adding the boundary condition label to a face

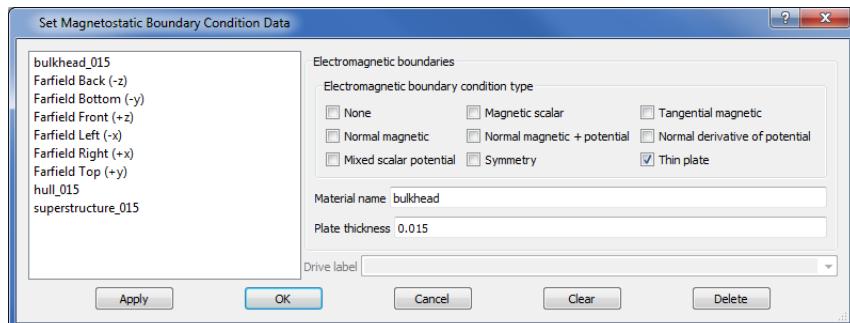


Figure 14.77 Specifying the Thin plate boundary condition

## Requirements for using the Thin plate boundary condition

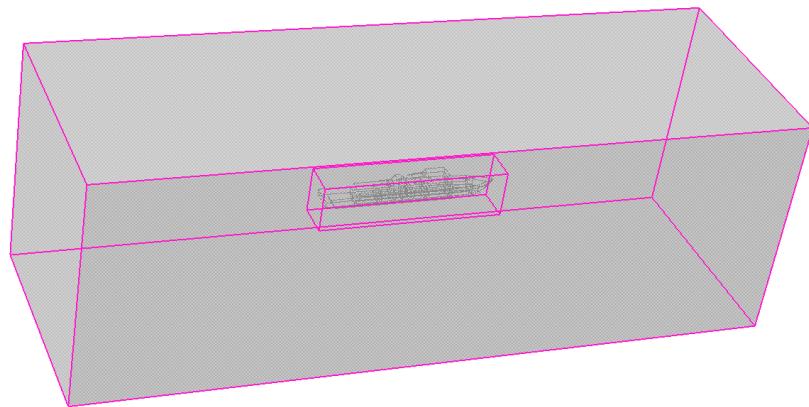
The boundary condition can only be used when the thin plate is completely surrounded by Total magnetic scalar potential. **Opera-3d Modeler** detects if the cells on either side of the boundary condition are set as Reduced potential (the default for material AIR) and will issue an error message while creating the **.op3** file.

Boundary conditions:

## THINPLATE

ERROR: Cannot be used on a face of a reduced potential cell.

In the example ship model, all cells in the model are set as Total potential, using the **Cell properties** dialog, except for the outer background region cell, which is set as Reduced potential to allow the model to be excited by an external magnetic field (see [Figure 14.78](#) for the overall model). However, the Reduced potential does not touch any of the faces labelled with the Thin plate boundary condition.



*Figure 14.78 Overall model space with outer Reduced potential cell highlighted*

## Results from the ship example

An athwartships external field of -70 A/m has been applied to the thin plate ship model. The model has been run as a non-linear simulation but, in fact, converges after the first iteration. [Figure 14.79](#) shows the magnetic flux density in the ship structure using a cut plane view. Even though the **Thin plate** boundary condition is applied on a two dimensional surface, the field values are multivalued. This can be seen more easily in [Figure 14.80](#) and [Figure 14.81](#) which show the athwartships component of flux density on one side of the hull and superstructure. As would be expected, there is a strong (negative) component on the inside where the bulkheads and decks meet the hull but this cannot be seen on the exterior as the flux is then predominantly in the plane of the hull.

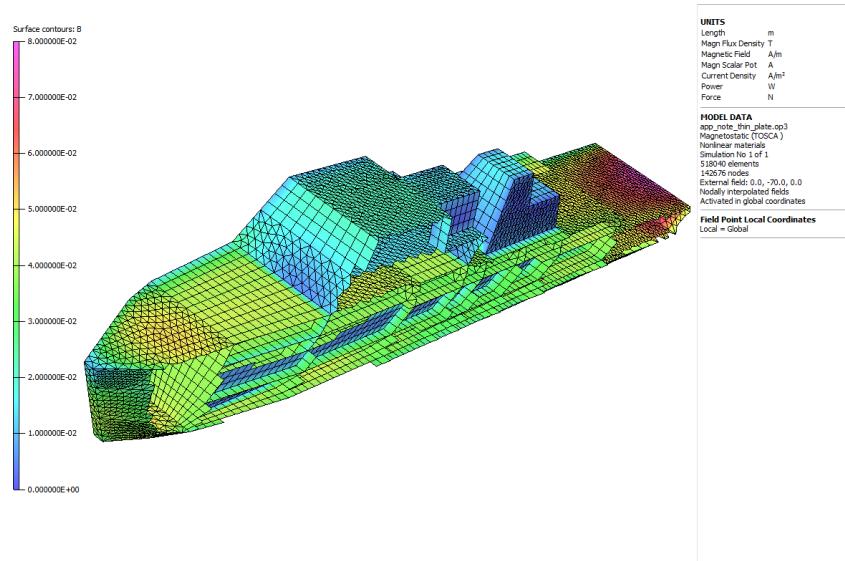


Figure 14.79 Flux density on ship

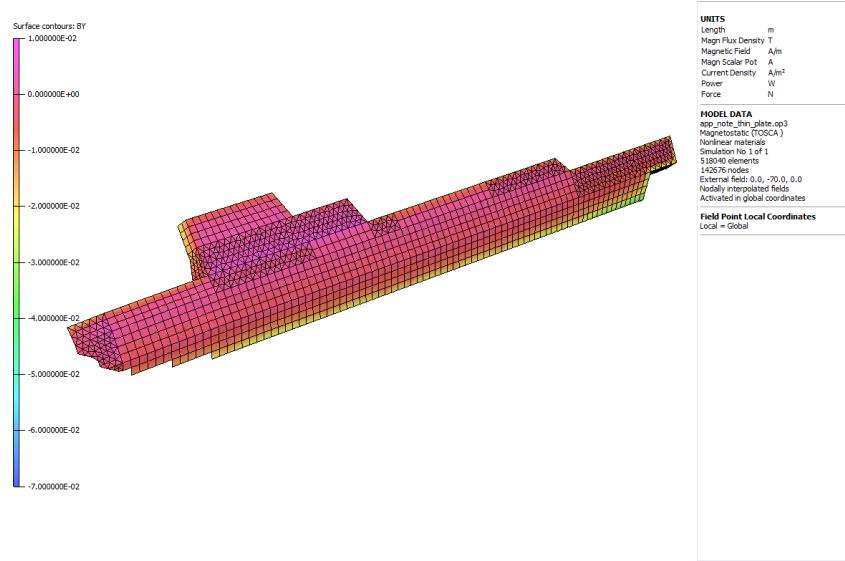
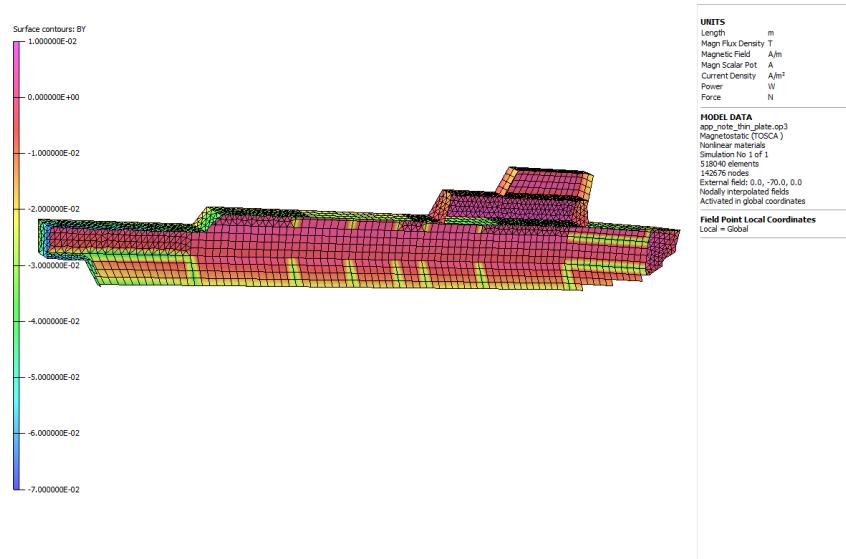
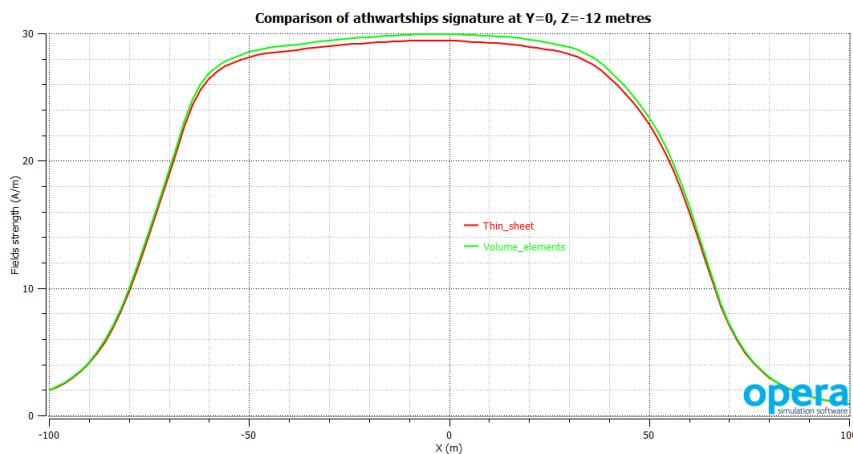


Figure 14.80 Athwartships flux density on the exterior of the hull and superstructure



*Figure 14.81 Athwartships flux density on the interior of the hull and superstructure*

The same ship model has also been simulated with volume elements in the thin plates. The number of equations solved in this model is more than 5 times greater than with the thin sheet model and it takes nearly 10 times longer to solve. However, as can be seen in Figure 14.82, the computed athwartships signature field on a line in the longitudinal direction 6 metres below the bottom of the hull and at the centre of the ship is similar. The signature field is computed from component HY+70 (since the applied field is -70 A/m) and integral field calculations are used.



*Figure 14.82 Athwartships signature field*

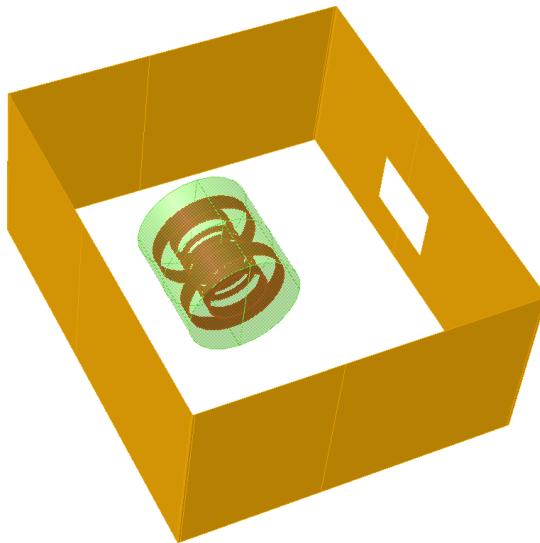
## A second example: Magnetic shielding of an MRI magnet

The **Thin plate** boundary condition has also been used to model the magnetic shielding in a calculation of the stray field from a shielded room surrounding a representative superconducting coil set of an MRI magnet. [Figure 14.83](#) shows the geometry of the shielded room and the coil set, which is enclosed with a volume of Reduced potential. The remainder of the free space in the model uses Total magnetic potential.

In [Figure 14.84](#) and [Figure 14.85](#) respectively, the magnitude of the flux density from the **Thin plate** model 10 cm outside the window is compared to the same result with a volume representation of the same shielded room meshed with hexahedral and prism elements in the shield walls.

[Figure 14.86](#) shows the difference in these two results obtained by using the **Arithmetic on Table Files** option in the Post-Processor to subtract the thin sheet result, shown in [Figure 14.84](#), from the volume result, shown in [Figure 14.85](#). As can be seen the maximum error resulting from using the **Thin plate** approximation is less than around 1.5 Gauss compared to maximum stray fields of around 5-6 Gauss, even though the sampling for the field is being made close to the approximated geometry. At greater distances from the plates, the error would be smaller. (The overall room dimensions are 700 x 600 x 200 cm by comparison.)

17/Sep/2015 16:48:21



*Figure 14.83 Thin plate representation of a shielded room*

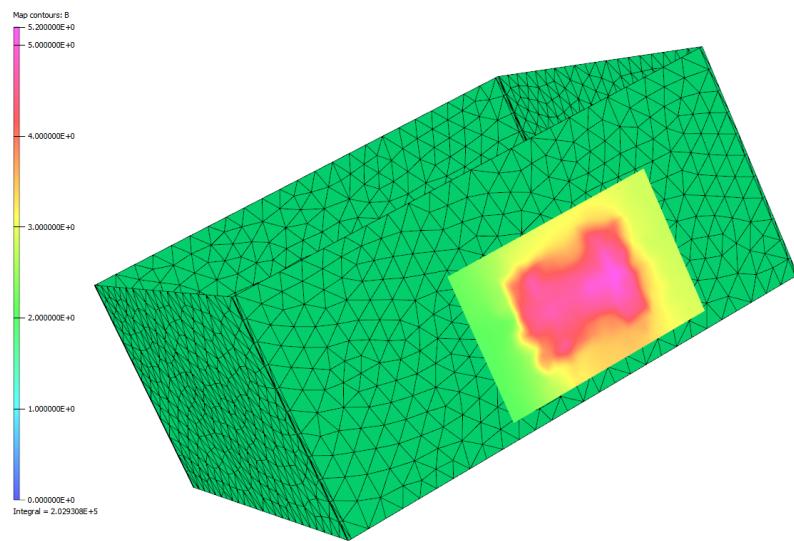


Figure 14.84 Stray field outside window in thin plate model

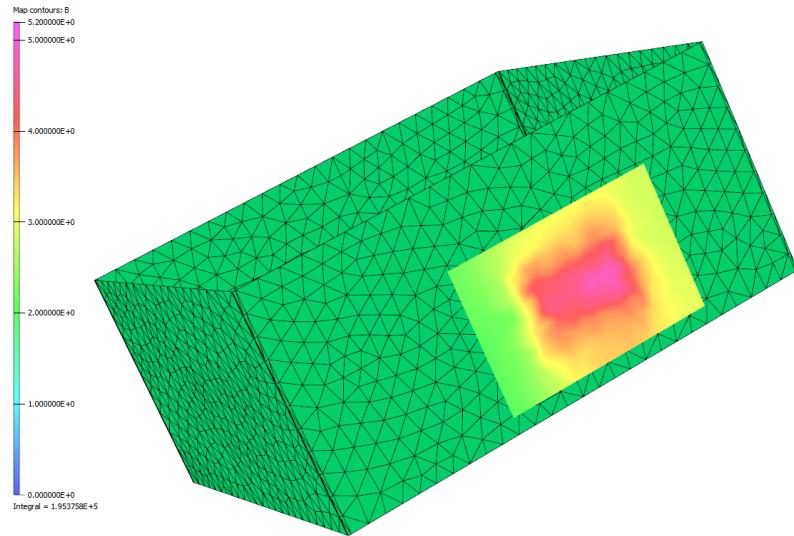


Figure 14.85 Stray field outside window in volume meshed model

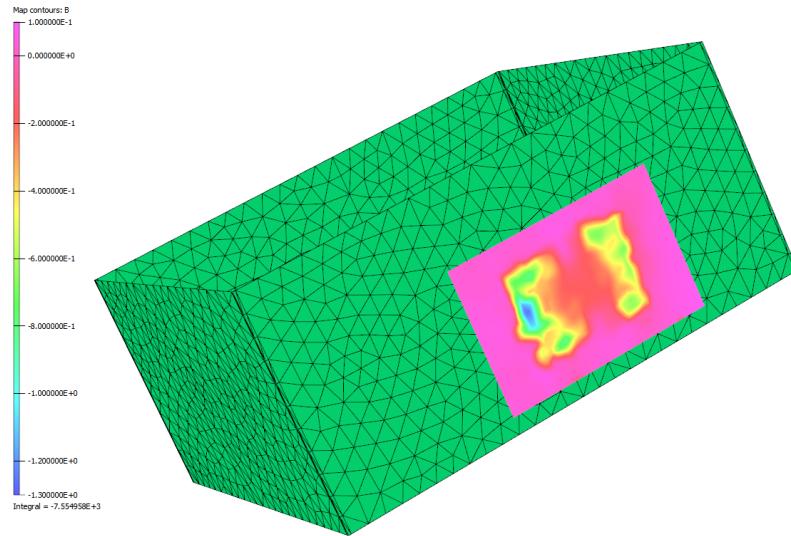


Figure 14.86 Difference in stray field calculations

## Guidelines and limitations

The **Thin plate** approximation is designed to model geometry with extreme aspect ratio and allows the two dimensional surface elements to be much larger than the plate thickness. However, where the dominant field is normal to the plate the accuracy of the approximation relies on the AIR elements surrounding the sheet being large in all dimensions compared to the thickness of the sheet. This is naturally enforced in the directions tangential to the sheet, since modelling extreme aspect ratio geometry is the purpose of the boundary condition, but also needs to be true in the direction normal to the sheet. Experience has shown that the surrounding elements need to be 10 times larger than the plate thickness for reasonable accuracy where the field is normal to the plate. Hence, the approximation could not be used successfully to represent two parallel sheets with a separation in the same order as the plate thickness. Conversely, if the dominant field is tangential to the plate this restriction can be relaxed and experience has shown that elements either side that are of the order of the plate thickness can still give reasonable accuracy.

There are two other limitations in the use of the approximation, apart from the restriction on only using Total potential to surround the sheet.

1. Using a **Thin plate** boundary condition within a model turns off automatic cutting, even if the user has selected this in the **Analysis Settings** dialog (the default behaviour). This means that neither volume meshed Total potential regions nor **Thin plate** surfaces can be multiply connected (if they surround a non-zero net current). A Total potential volume or **Thin plate** surface is multiply connected if it is possible to define a continuous path from a starting point and return to the same point enclosing a volume of Reduced potential. The user becomes responsible for making the cuts in volume meshed regions, while a **Thin plate** can never be multiply connected.

For example, the **Thin plate** approximation could not be used to model a plate with a hole through it where the GO current of a coil passes through the hole but the RETURN current is outside the plate. Consequently, users of magnetostatic simulation for signature calculations in naval vessels should not use the boundary condition if degaussing coils pass through a hole in a bulkhead or deck.

Opera-3d issues a warning that automatic cutting has been turned off because the boundary condition has been used.

2. The **Thin plate** boundary condition cannot exist on the exterior surface of a model. The model must include Total potential on both sides of the sheet, even if it is implied by the symmetry of the model and a **Tangential magnetic** boundary condition is used for the remainder of the symmetry face.

## Using Current Source Boundary Conditions

Opera-3d solvers have the ability to drive the software through applied voltage boundaries. In many cases, this implementation is sufficient. However, for operations such as lightning strikes, the current waveform of the strike is known. Replacing the imposed current by a voltage drive is difficult and can make the matrix poorly conditioned, leading to poor solutions.

Implementing a general boundary condition will make these conditions significantly easier to implement for the user.

The Static Current Flow solver has a DERIVATIVE boundary condition.

$$\frac{\partial V}{\partial n} = -(\bar{E} \cdot \bar{n}) \quad (14.23)$$

Hence, the input current density is

$$(\bar{J} \cdot \bar{n}) = -\frac{\sigma \partial V}{\partial n} \quad (14.24)$$

This new boundary condition is similar, but is extended to the Transient EM eddy current solvers as well.

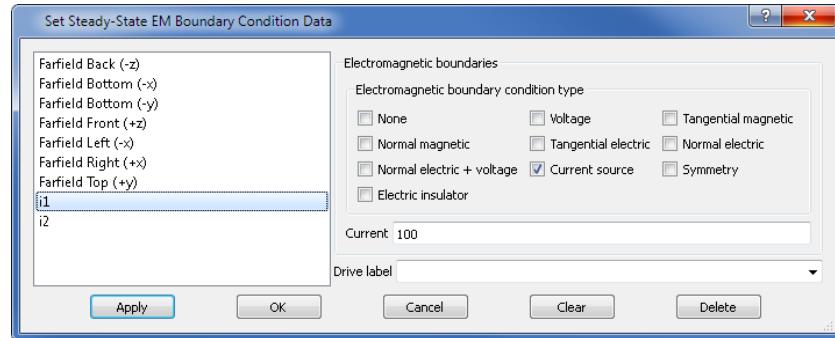
The current source boundary condition is available for the following Opera-3d solvers:

- Static Current Flow;
- Harmonic / Transient Electromagnetic and Fixed Velocity;
- Harmonic High Frequency;
- Transient Motion; and
- Magnetization.

The current source boundary condition can also be used with materials having surface impedance boundary condition (SIBC). For more on SIBC, please refer to [Using Surface Impedance Boundary Conditions in Harmonic Electromagnetic \[page 610\]](#).

### Building a model with a Current Source boundary condition

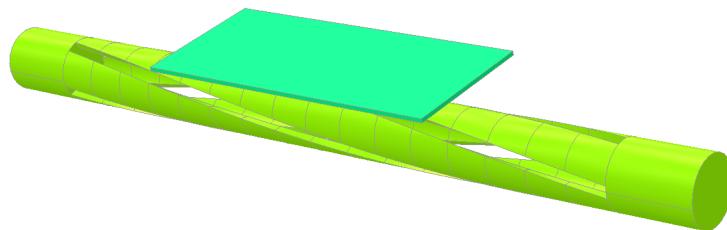
A model using the Current Source BC is built in exactly the same way as a model with voltage boundary conditions. Labels are applied to material faces on the outer boundary of the finite element mesh. On these labels, the current source boundary condition is then defined as shown in [Figure 14.87](#).



*Figure 14.87 Boundary condition dialog with current source boundary selected*

Please note that the condition defines the total current entering a face. So for an eddy current simulation the value of  $J$  can be non-uniform due to skin-effects.

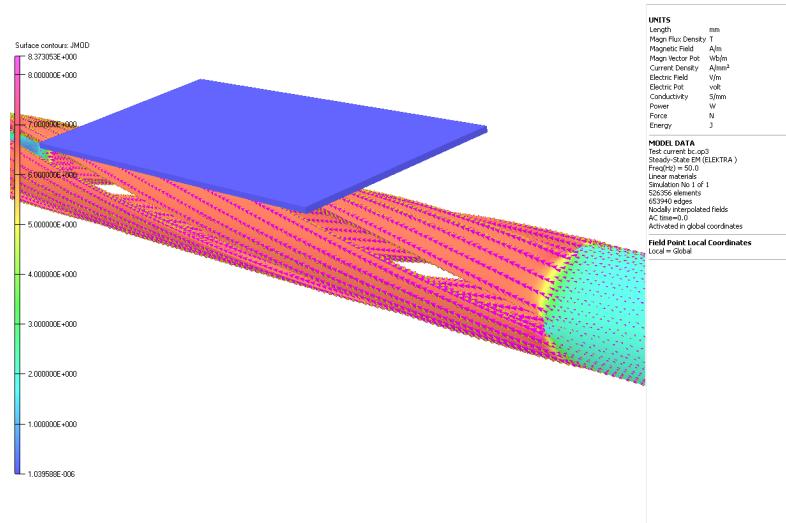
*Figure 14.88* shows a simple model where current source boundary condition has been used.



*Figure 14.88 A current carrying conductor and an aluminium plate*

A copper conductor with some gaps in it is carrying a current that will induce eddy currents in an aluminium plate above. The current on the end face on the right-hand side, labelled  $i_1$ , is set to 100 Ampere.

The end face on the left-hand side, labelled  $i_2$ , is not set to -100 Ampere, but clamped to 0 Volt, so that the value of voltage is gauged in the finite element model. This is recommended if a material would have matching input and output current source boundary conditions and no normal electric or assigned voltage conditions.



*Figure 14.89 Current density in copper*

Figure 14.89 shows a plot of current density in the copper bar. The current density vectors point from the right to the left. The induced eddy currents in the aluminium plate are about 4 orders of magnitude smaller; therefore the plate looks blue at this level of scaling.

## Limitations

The user is responsible for ensuring that the boundary conditions are consistent. Examples of inconsistency would be:

- Defining 2 boundary conditions, both of which are current sources into the model, with no other conducting boundaries where current can exit the model.
- Defining a surface with current flowing in to the surface, but where there is no conducting path for the current to flow out of the model.
- Current source boundary conditions applied to different labels may not share a common edge or node on the surface, where these nodes are a part of a region with conductivity defined.

Current source boundaries may only be applied to external faces of the finite element mesh, and at least one of the elements touching that face must have a conductivity defined.

Figure 14.90 shows a plot of flux density.

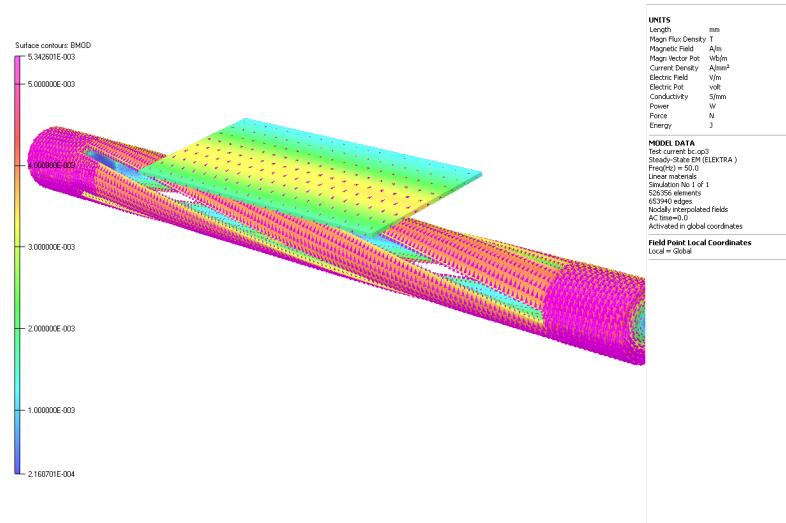


Figure 14.90 Flux density in the copper bar and aluminium plate

## Power and Energy Calculation in Harmonic solutions

---

### Power Calculations

If current and voltage waveforms vary as:

$$\begin{aligned} i &= I \cos((\omega t)) \\ v &= V \cos((\omega t - \phi)) \end{aligned} \quad (14.25)$$

then the instantaneous power in the system is given by:

$$\begin{aligned} P(\omega t) &= vi = VI \cos((\omega t)) \cos((\omega t - \phi)) \\ &= \frac{VI}{2} [\cos((\phi)) + \cos((2\omega t - \phi))] \\ &= A + \frac{VI}{2} \cos((2\omega t - \phi)) \\ &= A + \frac{VI}{2} \cos((\phi) \cos((2\omega t)) + \frac{VI}{2} \sin((\phi)) \sin((2\omega t)) \\ &= A + B \cos((2\omega t)) + C \sin((2\omega t)) \end{aligned} \quad (14.26)$$

where

$$\begin{aligned} A &= \frac{VI}{2} \cos((\phi)) \\ B &= \frac{VI}{2} \cos((\phi)) \\ C &= \frac{VI}{2} \sin((\phi)) \end{aligned} \quad (14.27)$$

Hence when  $\omega t=0$ ,  $\omega t=\pi/4$ , and  $\omega t=\pi/2$ :

$$\begin{aligned} P(0) &= A + B \\ P\left(\frac{\pi}{4}\right) &= A + C \\ P\left(\frac{\pi}{2}\right) &= A - B \end{aligned} \quad (14.28)$$

The time-average power is given by:

$$P_{tav} = \frac{P(0) + P\left(\frac{\pi}{2}\right)}{2} = A = \frac{VI}{2} \cos((\phi)) \quad (14.29)$$

### Energy Calculations

Since nonlinear materials are not modelled, in a magnetic circuit, **B** and **H** are sinusoidal. Replacing the *i* and *v* with **B** and **H**, the same arguments follow, such that the time-average energy density is given by:

$$E_{tav} = \frac{E(0) + E\left(\frac{\pi}{2}\right)}{2} = A = \frac{B \cdot H}{2} \cos((\phi)) \quad (14.30)$$

## General

*P* is the system variable **POWER**, and *E* is the system variable **ENERGY**, set after the **ENERGY** command has been issued, and *ωt* is set by the **TIME** parameter (in degrees). Further notes are available under the relevant commands in the Reference Manual.

## Inductance Calculations in Opera-3d

---

The inductance of a coil is defined by either of the following equations:

$$E = \frac{1}{2} LI^2 \quad (14.31)$$

$$L = \frac{N\Phi}{I} \quad (14.32)$$

where  $E$  is the energy,  $I$  is the current applied at the terminals of the coil,  $\Phi$  is the flux linking the coil,  $L$  is the inductance, and  $N$  is the number of turns in the coil.

### Single Coil Model

In models where there is only one coil, no permanent magnets and no external fields, the inductance of the coil may be calculated in two ways.

#### No magnetic material

If no magnetic material is present (**Magnetostatic** and **Charged Particle**) or no magnetic or conducting material (**Harmonic / Transient Electromagnetic** and **Transient Motion**) is present, then the coil may be represented in the Post-Processor with no mesh. The flux linking the coil may be calculated using a suitable patch and the expression for  $B_{normal}$  is integrated by the **MAP** command where

$$B_{normal} = B_x n_x + B_y n_y + B_z n_z \quad (14.33)$$

Then  $\Phi = \int \int B_{normal} ds$  and equation 14.32 may be used.

#### With magnetic material

If a magnetic material is present (**Magnetostatic**) or if a magnetic and/or conducting material is present (**Harmonic / Transient Electromagnetic** and **Transient Motion**), then a mesh must be prepared, and an analysis carried out. In the Post-Processor, the total energy of the whole model may be calculated using the **ENERGY** command and the inductance is given by re-arranging equation 14.31.

## Multiple Coil Models – Linear Materials

In models where linear magnetic or conductive material is present, and multiple coils are being modelled, the inductance can be determined in two ways.

#### Multiple analyses and superposition

The total inductance of the system is given by

$$\begin{aligned} L_{total} &= L_{11} + M_{12} + M_{13} + \dots + M_{1n} + L_{22} + M_{21} + M_{23} + \dots \\ &+ M_{2n} + L_{nn} + M_{1n} + M_{2n} + \dots + M_{nn} \end{aligned} \quad (14.34)$$

$$L_{total} = L_{11} + L_{22} + \dots + 2M_{12} + 2M_{13} + \dots \quad (14.35)$$

To obtain  $L_{nn}$  an analysis is carried out with the  $n^{\text{th}}$  coil excited, and all other coils carrying no current. Each self-inductance is determined using a separate analysis and equation 14.31.

To obtain the mutual-inductance,  $M_{nm}$ , an analysis is carried out with the  $m^{\text{th}}$  and  $n^{\text{th}}$  coils excited with all other coils carrying zero current. A second analysis with the  $n^{\text{th}}$  current reversed is then carried out. This gives two equations:-

$$E_1 = \frac{1}{2}L_{mm}I_m^2 + \frac{1}{2}L_{nn}I_n^2 + M_{nm}I_mI_n \quad (14.36)$$

$$E_2 = \frac{1}{2}L_{mm}I_m^2 + \frac{1}{2}L_{nn}I_n^2 - M_{nm}I_mI_n \quad (14.37)$$

Subtracting equations 14.36 and 14.37 gives

$$E_1 - E_2 = 2M_{nm}I_nI_m \quad (14.38)$$

and  $M_{nm}$  may be determined.

## Flux linkage

Using the flux,  $\Phi$ , and the current in the coil,  $I$ , the self-inductance can be calculated as:

$$L = \frac{N\Phi}{I} \quad (14.39)$$

A similar equation can be used for the mutual-inductance, using the flux in one coil due to the flux generated by the current in another. This is calculated using:

$$M_{ij} = \frac{N_i\Phi_j}{I_j} \quad (14.40)$$

Using the solution to the model with one coil switched on will allow you to find the self-inductance of the coil, and the mutual-inductance of other coils with respect to that coil.

## Multiple Coils – Nonlinear Materials

If you wish to use nonlinear properties it is necessary to have a solution with all coils at normal operating conditions. This gives the equation for the flux linking coil 1 as:

$$N_1\Phi_1 = L_1I_1 + M_{12}I_2 + M_{13}I_3 + \dots + M_{1n}I_n \quad (14.41)$$

for a model with  $n$  coils. Similar equations can be generated for  $\Phi_2$  etc.

The model must then be altered so that the current in any one of the coils is changed by a small amount ( $\Delta I$ ), saving the model, and generating a new solution file (using restart run to speed matters up).

The small change should not greatly affect the total field and so a material permeabilities should not change significantly, but a small change of  $\Delta\Phi_1$  will occur for each coil. For example, for a change of  $\Delta I_1$ :

$$\begin{aligned} N_1(\Phi_1 + \Delta\Phi_1) &= L_1(I_1 + \Delta I_1) + M_{12}I_2 + M_{13}I_3 + \dots + M_{1n}I_n \\ N_2(\Phi_2 + \Delta\Phi_2) &= L_2I_2 + M_{21}(I_1 + \Delta I_1) + M_{23}I_3 + \dots + M_{2n}I_n \\ &\dots \end{aligned} \quad (14.42)$$

Hence taking the difference between the first line of equation 14.42 and equation 14.41, we can calculate  $L_1$ :

$$L_1 = \frac{N_1 \Delta\Phi_1}{\Delta I_1} \quad (14.43)$$

The second line of equation 14.42 and equation 14.41 will give  $M_{21}$ , and so forth. Modifying  $I_2$  would allow similar calculations for  $L_2$  etc. Therefore, finding the self and mutual-inductance for all  $n$  coils requires  $n$  models to be solved.

This method can also be used if other energy sources are present, e.g. permanent magnets or external driving fields, as it works on the basis of a *change* in the field.

A change in energy calculation is possible but would require more solutions to be calculated, as only one piece of information is found from each model.

## Flux Linkage in Meshed Conductors

For meshed conductors, the flux linked can be calculated using

$$\Phi = tn \int \int \int \left( \frac{\mathbf{A} \cdot \mathbf{J}_c}{|\mathbf{J}_c|} \right) dV, \quad (14.44)$$

where

- $t$  is the turns density;
- $n$  is the number of symmetry replications to form the complete coil;
- $\mathbf{A}$  is the vector potential;
- $\mathbf{J}_c$  is the current density.

The value of the flux linked can be calculated using the **VOLUME** command with  
`component=#t*#n* (A_X*JCX+A_Y*JCY+A_Z*JCZ)/JC.`

Note that it is necessary to enable calculation of  $\mathbf{J}_c$  using the **SET** command. Also note that the vector potential  $\mathbf{A}_-$  is only available in edge variable solvers like **Transient Electromagnetic**. For flux linkage calculations in **Magnetostatic**, which is a nodal solver, please use the methods described above.

The following command script creates a dialog to obtain the coil information from the user and then calculates the flux linked, *SI* units assumed.

```
$DIALOG start 'Coil Data'  
$PROMPT coilname 'Circuit Winding label or Group label'  
$ASK #t 'Turns density (turns/m**2)'  
$ASK #n 'Number of reflections'  
$DIALOG stop  
  
/ Enable current density in coils  
SET JCOIL=YES  
  
/ Integrate over the selected conductor volume  
VOLUME ACTION=RESET  
VOLUME ACTION=ADD LABEL=&coilname&  
VOLUME ACTION=INTEGRATE TAVERAGE=NO,  
COMPONENT=(A_X*JCX+A_Y*JCY+A_Z*JCZ)/JC*#t*#n ADAPT=NO  
$CONSTANT #Flux INTEGRAL  
  
/ Report the flux linked  
$DIALOG start 'Flux'  
$GROUPBOX start 'Flux linked in &coilname&' vertical  
$DISPLAYLINE 'Flux linked %real(#Flux) Wb-turns'  
$GROUPBOX stop  
$DIALOG stop
```

## Legendre Polynomials

### Introduction

In the MRI worked example ([Getting Started \[page 35\]](#)) a number of field quantities were used to assess the model. However the quality of the field in the centre of the magnet was not investigated as it was outside of the scope of the example. This application note shows how the Modeller can be used to give accurate field quality results by precise control of the mesh.

In the original version of the MRI example, the mesh size in the centre of the magnet was undefined, with the mesh size being controlled only by the global element size used when meshing. This is a reasonable starting point for general field recovery, but when a very accurate field solution is required, as in MRI systems, this is far from ideal.

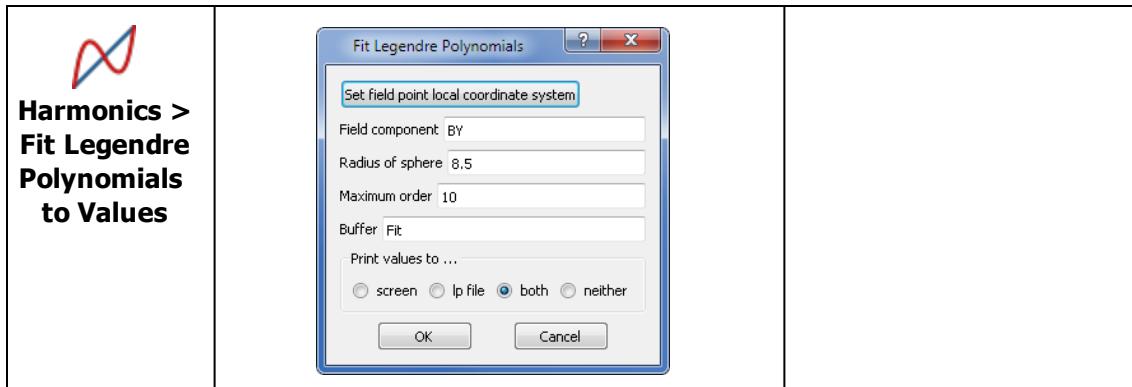
The method used to calculate the field quality in the Post-Processor is to compute the Legendre polynomial coefficients on a sphere with a user defined radius (using the **FIT** command). By ensuring that the model contains an accurate field solution where these field points are going to be calculated, the accuracy of this method can be greatly improved.

### Legendre Polynomial Coefficients

The fitting to obtain the Legendre polynomial coefficients is performed on a sphere with a defined radius. The program samples the field on the surface of the sphere and fits the results to the polynomial expansion. These are reported in a separate window. The form of the polynomial of order  $N$  is

$$B_y = \sum_{n=0}^N \sum_{m=0}^n P_n^m \cos(\Theta) [\alpha_n^m \cos(m\phi) + \beta_n^m \sin(m\phi)]. \quad (14.45)$$

The table of coefficients shown are the  $\alpha$  and  $\beta$  values in this equation. The order columns are  $n$  and  $m$  respectively. Consequently, the value of  $\alpha$  for  $n = m = 0$  is the mean value of the dipole field at this radius. If the sphere contains no sources (magnetic materials or currents), the associated Legendre polynomial is the solution to Laplace's equation and this value will be the field at the centre of the sphere.



Providing the field satisfies Laplace's equation, homogeneity on any other radii spheres can also be evaluated by inclusion of an additional  $r^n$  multiplier in the equation, where  $r$  is the radius normalized from the actual evaluation radius, i.e.

$$B_y = \sum_{n=0}^N r^n \sum_{m=0}^n P_n^m \cos(\Theta) [\alpha_n^m \cos(m\phi) + \beta_n^m \sin(m\phi)] \quad (14.46)$$

As may be expected from the symmetry of the magnet, the largest harmonic contaminants are at  $n = 2, m = 0$  and  $n = 4, m = 0$ . These probably result from the return flux through the legs of the magnet.

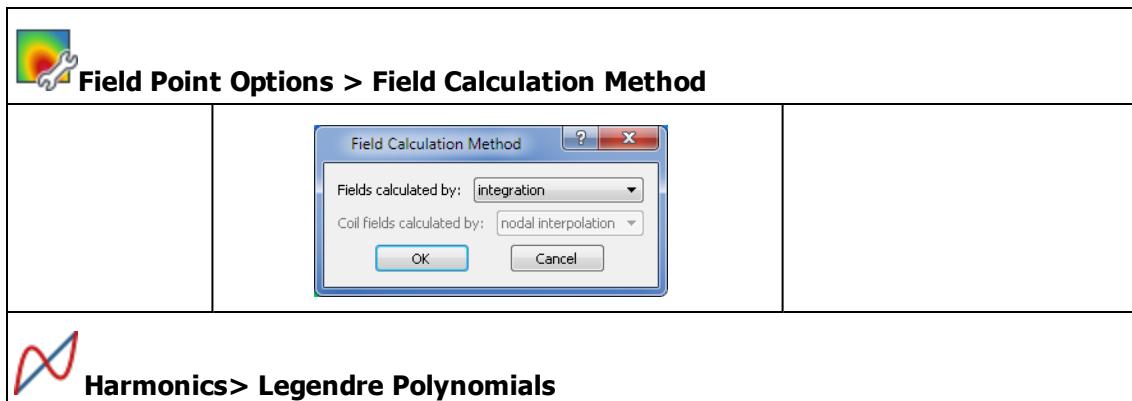
Results from this calculation method are shown in [Results \[page 649\]](#).

## Integral fields

In the above calculation, the values of field sampled on the spherical surface were computed using the element basis functions. The program determines in which element the sampling point lies and interpolates from the nodally averaged values. An alternative method of computing the field is to integrate the magnetization of the magnet. This has the advantage that the value computed at the sample point is not dependent on the local mesh size.

Calculating integral fields is more computationally expensive than nodal averaging, but for many

models it gives improved accuracy. Note that the process can be aborted using the  **Cancel** button .



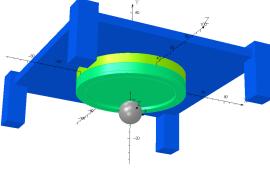
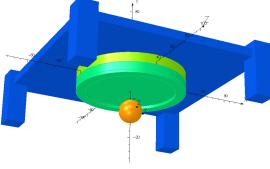
## Setting up the Model for Accurate Field Results

Calculating the harmonics of the field directly from the finite element mesh requires that the mesh is fine enough to capture the variation of the harmonic.

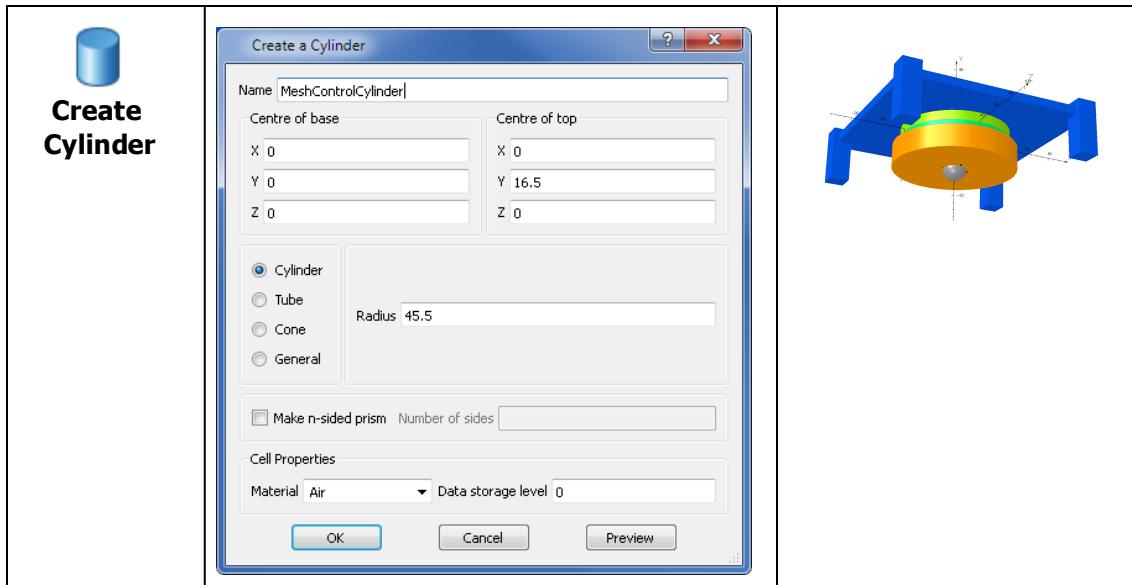
Azimuthally, the **FIT** command uses  $N+2$  points, where  $N$  is the order selected. So if the 10th order is required ( $N=10$ ) then it is essential to ensure that there are at least 12 points around the sphere's circumference.

In the polar direction the spacing of the points is defined to fit in with the polynomial zeros. This is much harder to obtain using a simple mesh control parameter. However, if quadratic elements are used on curved surfaces an accurate solution can be achieved when more than  $2N+4$  points are defined in the polar direction. So again if order 10 is chosen then 24 points around the polar circumference will give a good starting point for an accurate solution.

In the Modeller this is easy to achieve as a simple sphere of **AIR** can be defined at the centre of the magnet, where the field accuracy is most important. The cell properties for the sphere can then be adjusted to ensure that a fine mesh (mesh size 1.5) and quadratic elements are applied.

 <b>Create Sphere</b>	<p><b>Create a Sphere</b></p> <p>Name: MeshControlSphere</p> <p>Centre:</p> <ul style="list-style-type: none"> <li>X: 0</li> <li>Y: 0</li> <li>Z: 0</li> </ul> <p>Radius: 8.5</p> <p>Cell Properties:</p> <ul style="list-style-type: none"> <li>Material: Air</li> <li>Data storage level: 25</li> </ul> <p>OK Cancel Preview</p>	
 <b>Pick Cell</b>	<p>Select the sphere as a <b>CELL</b> and right-click to change it's Cell Properties.</p> <p><b>Cell Properties</b></p> <p>Material label: Air</p> <p>Potential type: Total Element type: Quadratic</p> <p>Data storage level: 25</p> <p>Volume data label</p> <p>Group label</p> <p>Mesh control parameters:</p> <ul style="list-style-type: none"> <li>Maximum element size: 1.5</li> <li>Maximum angle between elements</li> <li>Maximum deviation from surface</li> </ul> <p>Element shape preference: None</p> <p>OK Cancel Default</p>	

To improve the mesh under the pole face more widely, a cylinder of radius 45.5, from (0,0,0) to (0,16.5,0) is created. The cell properties of this cylinder are then adjusted to have a mesh size of 6, to allow the fields to be calculated more accurately.



The new mesh is generated, as seen in Figure 14.91 and the new analysis database created and solved. The Legendre calculations can be repeated on the new solution.

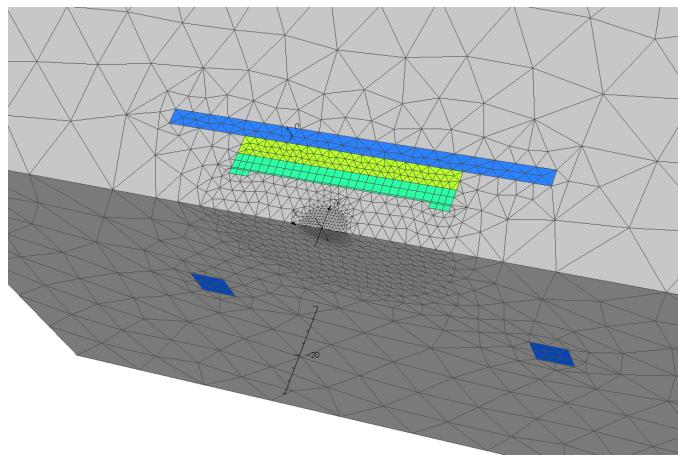


Figure 14.91 Surface mesh with sphere and mesh control changes

## Results

The following table compares the 0th, 2nd and 4th order Legendre polynomial coefficients for the original MRI magnet and for the refined model. The refined model features a sphere with a radius of 8.5 and a maximum element size of 1.5, and a refined mesh below the magnet pole.

<b>Order (m,n=0)</b>	<b>Original model</b>		<b>Refined model</b>	
	<b>Nodal</b>	<b>Integral</b>	<b>Nodal</b>	<b>Integral</b>
0	1909.7	1919.2	1919.1	1923.1
2	1.11	1.70	1.13	1.36
4	0.41	0.22	0.24	0.20

To achieve greater accuracy, it may be necessary to refine the mesh further throughout the model. Note that the **FIT** command and the integral field calculation method require that the complete model  is loaded (the default behaviour of **Open**).

In this example the symmetry has automatically been recovered from the database. If the model symmetry has not been defined in the Modeller, then it is necessary to activate the model with the correct reflections and rotational symmetry.

The values of the coefficients shown in the table will be dependent on the mesh generated. They may vary because of different representation of floating point numbers in different processor types. However, as can be seen, the nodal and integral results for the refined mesh are closer to each other than in the original model. This gives greater confidence on the accuracy of the model and the harmonic representation.

## Circuits in Harmonic and Transient EM simulations

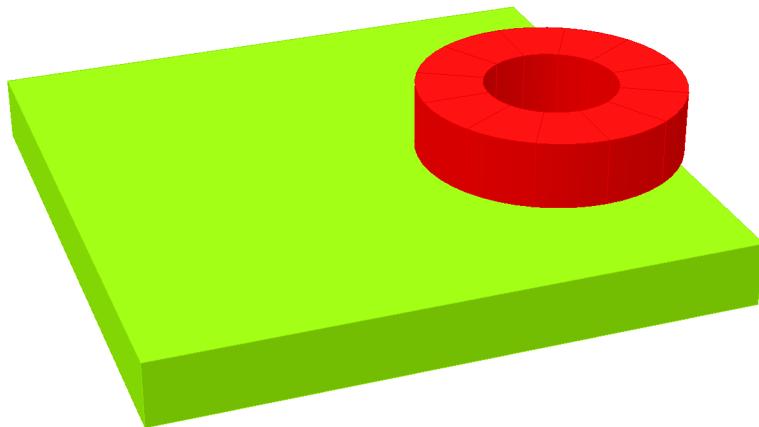
The coils in an eddy current analysis are frequently connected to a circuit. Rather than specify the current density in a coil, it is also possible to connect each coil or several coils in series to one or more circuit loops. The circuit may include active and passive components - voltage or current sources, switches, diodes, resistors, inductors and capacitors. External circuits can be used with any of the following solvers: **Harmonic / Transient Electromagnetic, Transient Electromagnetic with Motion, Magnetization and Quench.**

This application note shows the steps that must be followed to use these facilities using a very simple example of a solenoid coil connected to a voltage source<sup>1</sup>.

The data files used in this application note can be found in the **3D/External\_circuits** sub-folder of the Opera examples folder.

### Example geometry

The model consists of a simple solenoidal coil positioned above a conducting plate - a common geometry used in calibrating eddy current non-destructive test equipment, as shown in [Figure 14.92](#).



*Figure 14.92 Geometry of NDT model*

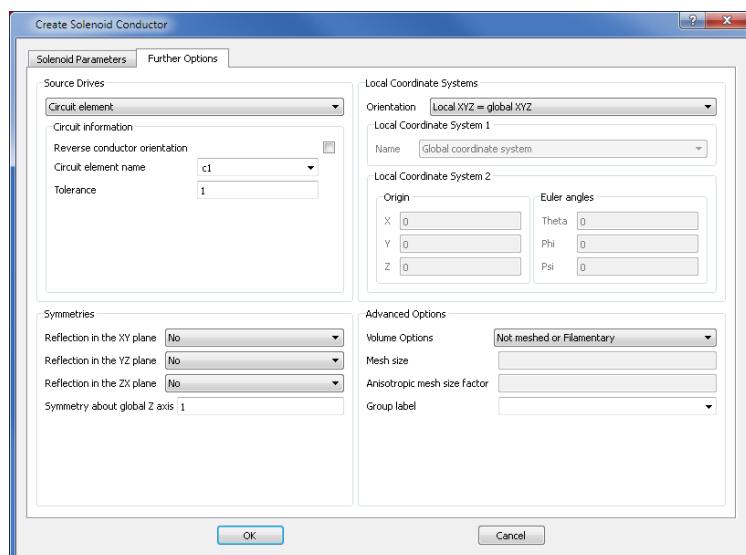
The coil consists of 50 turns of wire whose resistance is  $10 \mu\Omega/\text{cm}$  and the terminals of the coil are connected to a 50 Hz AC supply with 1 V peak voltage.

---

<sup>1</sup>Note that circuits are not available in Fixed Velocity Electromagnetic.

## Coil definition

The geometry of the coil is defined in exactly the same way as for a Biot-Savart coil that has a defined current density. The user identifies that the coil is connected to a circuit on the **Further Options** tab of the **Create solenoid Conductor** dialog. Two options are available on this tab - **Biot-Savart current source** or **Circuit element**, as shown in [Figure 14.93](#). When **Circuit element** is selected, the user is also prompted for the name of the circuit element. In this case, **c1** has been chosen. A check-box can also be selected to orient the current in the winding in the opposite direction.



*Figure 14.93 Dialog for selecting coil as circuit element*

## Winding properties

The second part of the circuit definition is to relate the **Circuit element name** chosen for the coil to the circuit in which it is connected. This is achieved using the Circuit Editor. The Circuit Editor is only available when the user has specified the Analysis type as being harmonic or time-transient (see previous page).

Clicking on the

toolbutton opens the Circuit Editor in a separate window, as shown in [Figure 14.94](#).

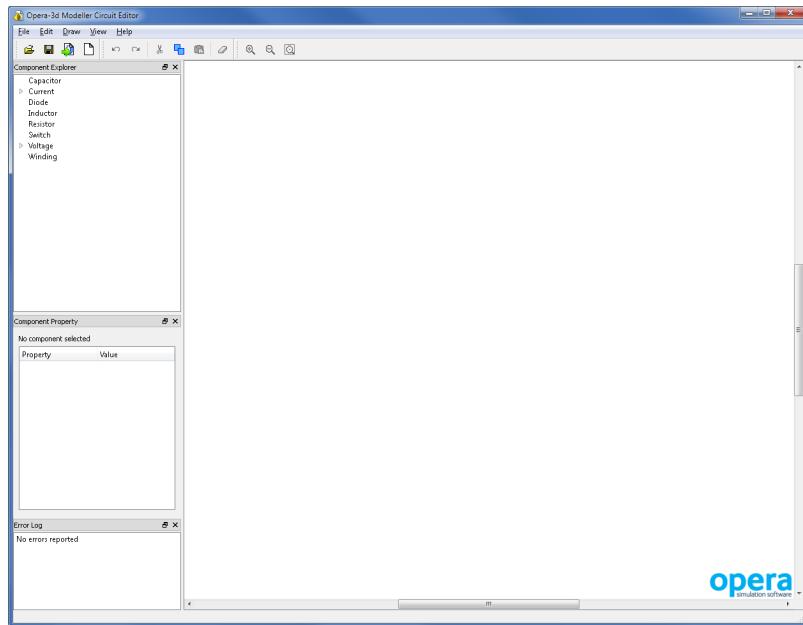


Figure 14.94 Initial window in Circuit Editor

Opening the **Winding** tree in the **Component Explorer** shows the winding label **C1**. Clicking on **C1** opens the **Component Property** window with the default set of parameter values for the coil.

These should be modified as follows:

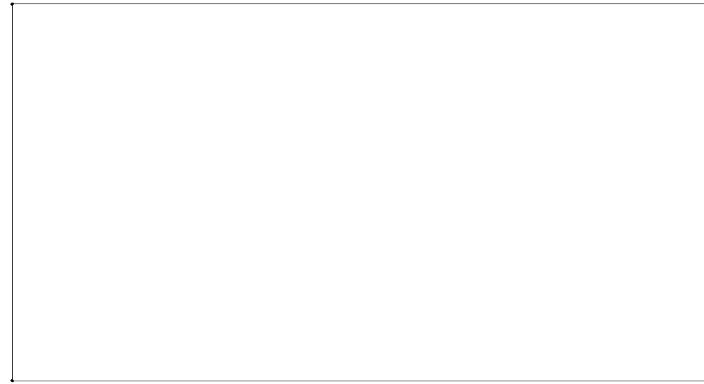
Wire resistance	$1e-5$ ohm/cm
Turns	50

The number of **Representative filaments** specifies from how many internal filaments the flux linking the coil (and hence the voltage of the coil) will be evaluated. In this case only **1** is chosen.

The other numerical parameter, **Symmetry adjustment**, defines the ratio of the total length of wire in the winding (including any symmetry copies) to the length of wire that actually exists in the meshed space. The Circuit Editor can determine the model symmetry (from **Model -> Model Symmetry**) and, if this also matches the winding symmetry, there is no need to specify the symmetry value separately. If this is not the case, the symmetry must be set. For example, if the coil in this example was moved to be centrally above the plate, a one quarter model could be constructed with appropriate boundary conditions. Three quarters of the coil would then be outside the model. So the **Symmetry adjustment** would be **4**. However, in this model the entire coil is in the meshed space so the value is set to **1**.

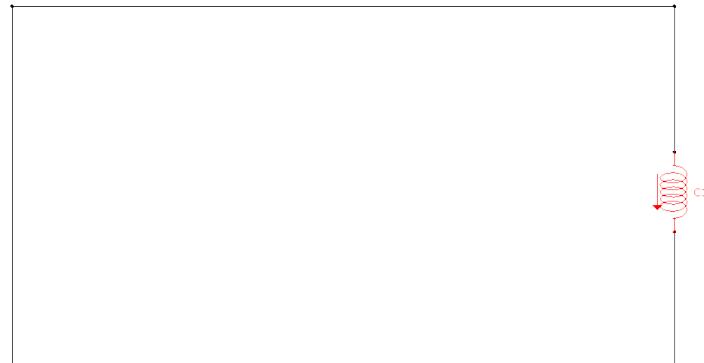
The next stage is to lay out the topology of the circuit. In this model, a single closed loop is all that is needed. Double-clicking with the mouse in the drawing space starts the beginning of a wire. Double-clicking again terminates the wire. Note that the Circuit Editor will always draw vertical or horizontal

wires. If the beginning and end points of the wire are not vertically or horizontally aligned, two wires will be drawn to connect them. [Figure 14.95](#) shows a simple example closed wire loop.



*Figure 14.95 Wire loop without any components*

The **C1** winding can now be dragged from the **Component Explorer** and dropped onto the right vertical arm of the loop, as shown in [Figure 14.96](#).



*Figure 14.96 Winding added to loop*

Note that an error message identifies that the winding is currently short circuited.

The voltage source can be added to the circuit by expanding **Voltage** in the **Component Explorer** tree and dragging an **AC voltage** onto the left vertical arm of the loop, as shown in [Figure 14.97](#).

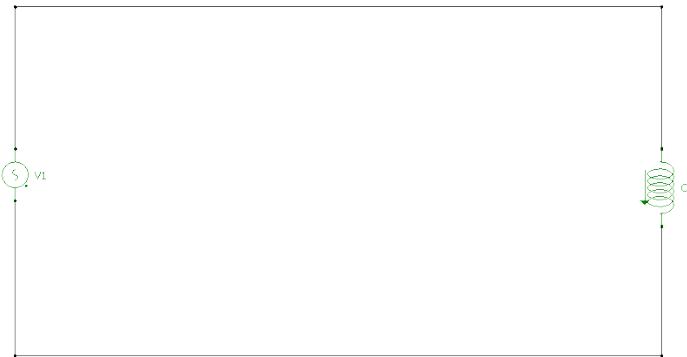


Figure 14.97 Completed circuit loop

A different error message now appears identifying that the peak voltage for the new supply has not been set. Either by double-clicking on this message or on the component, the **Component Property** panel for **V1** is displayed and the **Peak Voltage** can be set at 1 V. The default value of **Phase** is used.

## Analysis

An **op3** file can then be created in the normal way. During mesh generation, the mesh generator will ensure that edges of the elements lie along the representative filaments in the coil. During the analysis, the vector potential along the edges is integrated to compute the flux linking the winding.

After the equations have been solved, the current and voltage in the components of the circuit can be displayed in the **Circuit Viewer** of the Post-Processor (**Conductors -> View Circuit**). Moving the mouse above the component produces a tool tip information box that shows the complex voltage and current.

## Post-Processing

As with all Opera-3d models, the current density in coils can be evaluated using the system variables **JCX**, **JCY**, **JCZ** and **JCMOD**. The value will be the computed current in the coil (-2.468+j20.593 A) multiplied by the number of turns (50) and divided by the cross-sectional area of the coil (in this case 1 cm<sup>2</sup>). Figure 14.98 shows contours and vectors of the current density in the coil and plate.

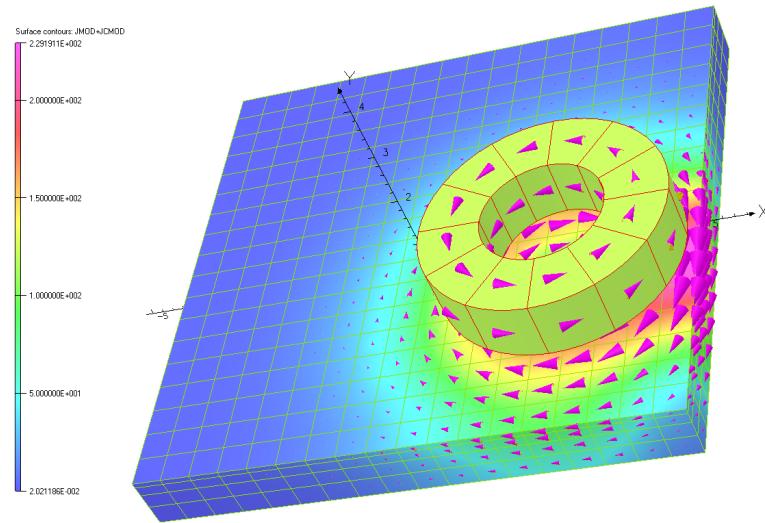


Figure 14.98 Current density in the coil and plate

## Time Dependent Drive Functions

In the previous example, the model was run using the **Harmonic Electromagnetic** analysis at a fixed frequency. Circuits can also be used in any of the transient analysis programs: **Transient Electromagnetic, Fixed Velocity, Transient Motion, Magnetization** and **Quench**. The Circuit Editor supports DC, AC, 3-phase AC and functional voltage sources and DC, AC and functional current sources. It also supports switches and diodes allowing more complex waveforms to be applied. Figure 14.99 shows a simple modification of the circuit from the previous model to apply a step function DC voltage at the beginning of the transient.

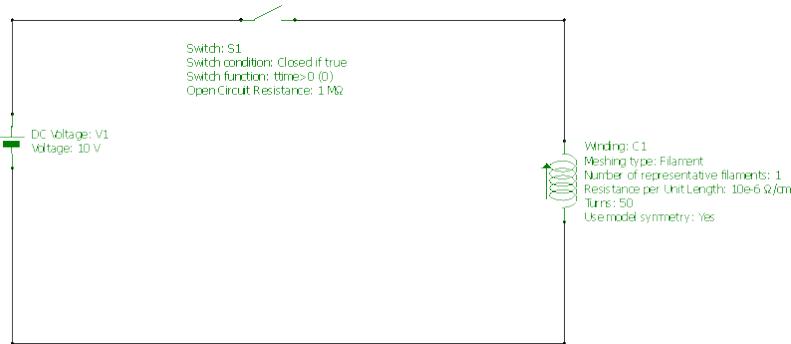


Figure 14.99 Simple switched circuit

## Functional drives

In modern power electronics, the switching functions required are often much more complex. A PWM drive is a good example of this.

To enable complex drives to be specified, it is possible to define the voltage or current source as a user defined function. Alternatively, the **Switch function** for a switch can make use of a user defined variable. Before the analysis is run, the user should create a **comi** file in the same folder as the **op3** database. The user should also create the user defined variable in the Modeller assigning it an arbitrary value (as the **comi** file will overwrite it). The **comi** file defines the value of the user variable at any time during the transient analysis. The **comi** file must be called **op3filename\_PROGRAMNAME.comi**, where **op3filename** is the name of the **op3** database and **PROGRAMNAME** is the name of the analysis program e.g. **ELEKTRATR** for the **Transient Electromagnetic**. A simple example follows.

## Chopped sine wave drive

The example shown earlier in this application note is now run as a **Transient Electromagnetic** simulation with a chopped sinusoidal drive. The chopping is achieved by opening or closing a switch. The function used for the switch is a square function of the same frequency taking the value 1 or 0 and a user defined mark-space ratio from the associated **comi** file. Because of the inductance of the system, the current in the winding will not become zero when the voltage supply is switched off, but will dissipate through the 0.02 Ohm resistor, which is switched in when the supply is disconnected. To see the effect of the chopping, the model is also run with a sinusoidal drive. Figure 14.100 shows the circuit.

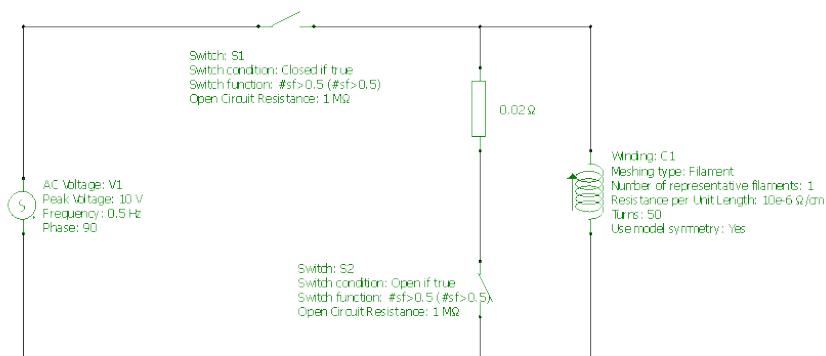


Figure 14.100 Chopped circuit

The name of the database is **external2\_chopped.op3**; the command file which defines the value of **#sf** is in the same folder and is called **external2\_chopped\_ELEKTRATR.comi**. Figure 14.101 shows the command file to achieve the chopped drive.

```

/ Specify period of sine wave in seconds
$constant #periodT 2

/
/ Specify the mark-space ratio
$constant #mark 0.6

/
/ Calculate the time in the current period
$constant #t Mod(ttime;#periodT)

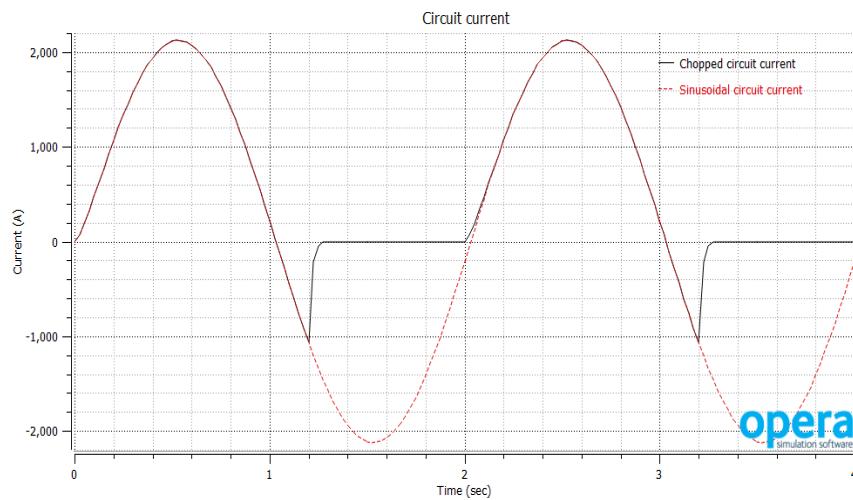
/
/ Set the drive on if the time in the period is less than the switch off
/
$constant #sf (#t/#periodT)<=#mark

```

*Figure 14.101 Command file for chopped drive*

After the analysis has been run, the current in the circuit can be plotted from the file **external2\_chopped.log** using the Post-Processor.

The menu option  **Plot Graph** results in the graphs shown in **Figure 14.102**. The current for the sinusoidal drive at 0.5 Hz is also shown.



*Figure 14.102 Current in winding*

As expected, when the voltage drive is disconnected by the switch after 1.2 seconds in every period (equivalent to a mark-space of 0.6), the current in the circuit reduces exponentially, with a time constant defined by the ratio of the inductance to the resistance of the circuit.

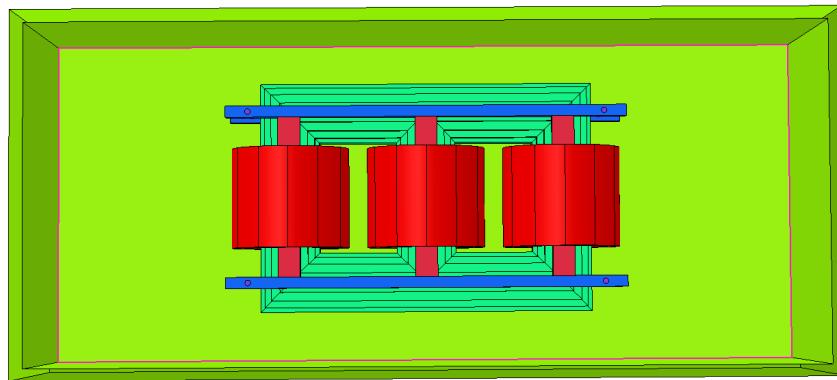
## Example Circuits

The coils in electromagnetic devices are usually connected to an electronic circuit. In order to model this, Opera-3d supports circuits in **Harmonic / Transient Electromagnetic, Fixed Velocity, Transient Motion, Magnetization** and **Quench**. This application note shows examples of circuits used for various applications constructed using the Circuit Editor. The circuit (**vfc**) files detailed are available in the examples folder (under **3D\External\_circuits**) provided with Opera. This can be accessed from the Opera Manager using **File -> Open Examples Folder**.

In all cases the units for a particular value specified within the Circuit Editor may be adjusted as required (e.g. mm, cm, m, inch). Explicit values may need to be replaced by constants/parameters or functions (e.g. #R1, Range(TTime;#t1;#t2)\*#R1) which may be adjusted either from within the Modeller or a command control file used during the simulation process (any constants or parameters used must be defined before the analysis is started).

### Example 1: 3-Phase Transformer Circuit

The model consists of a simple 3-phase transformer modelled using a solenoid conductor, as shown in [Figure 14.103](#).



*Figure 14.103 A model of a 3-phase transformer*

The primary coil consists of 400 turns of wire whose conductivity is  $5.76E7 \text{ S cm}^{-1}$  and the terminals of the coil are connected to a 50 Hz 3-phase AC supply with 230 volt line-to-line voltage. The secondary coil consists of 100 turns and is connected to a load resistance of 10 ohms.

## Circuit setup

The 3-phase voltage is modelled using two functional voltage drives, as shown in Figure 14.104.

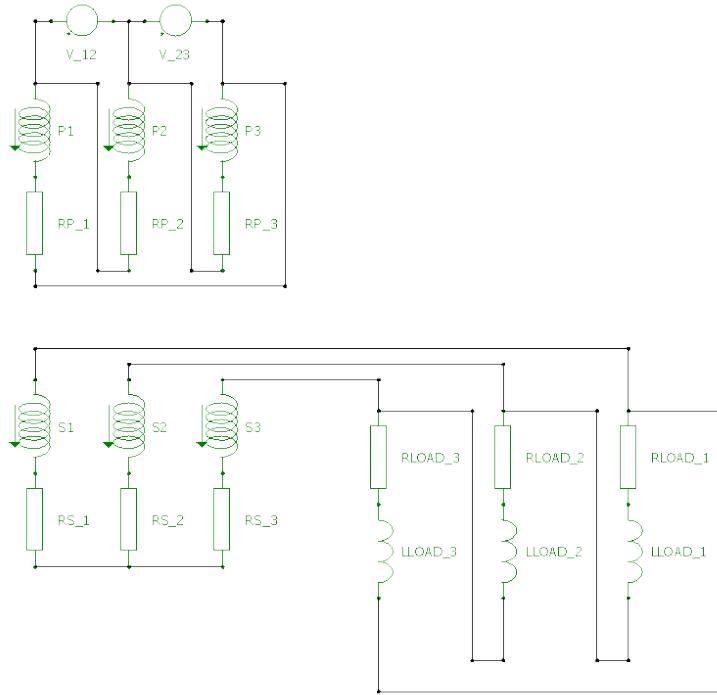


Figure 14.104 3-phase circuit of a transformer

The winding configuration used here is "Dy1" as per the international standard IEC 60076-1, where the number represents the phase displacement<sup>1</sup> clock number (i.e. the primary is delta connected and secondary is star connected). The voltage applied to the primary is ramped to the rated voltage. The model is analysed using the **Transient Electromagnetic** analysis, which solves the transient low frequency electromagnetic equations using a time-stepping method.

## Results

The current and voltage from the primary and secondary windings are recorded and are as shown in Figure 14.105, Figure 14.106, Figure 14.107 and Figure 14.108.

---

<sup>1</sup>Phase displacement of a 3-phase winding - the angular difference between the phasors representing the voltages between the neutral point (real or imaginary) and the corresponding terminals of two windings.

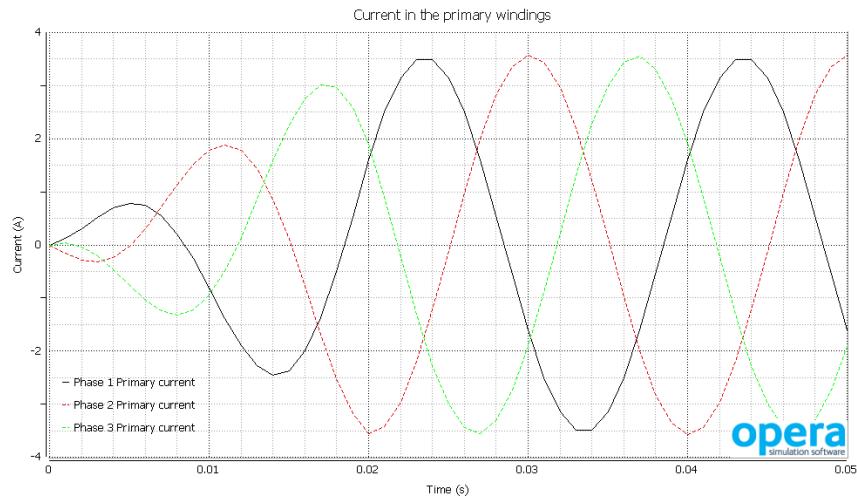


Figure 14.105 Current in the primary windings

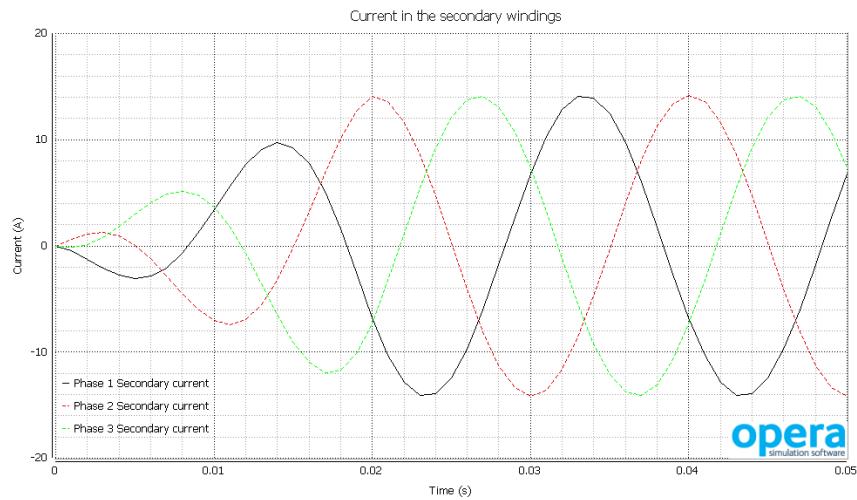


Figure 14.106 Current in the secondary windings

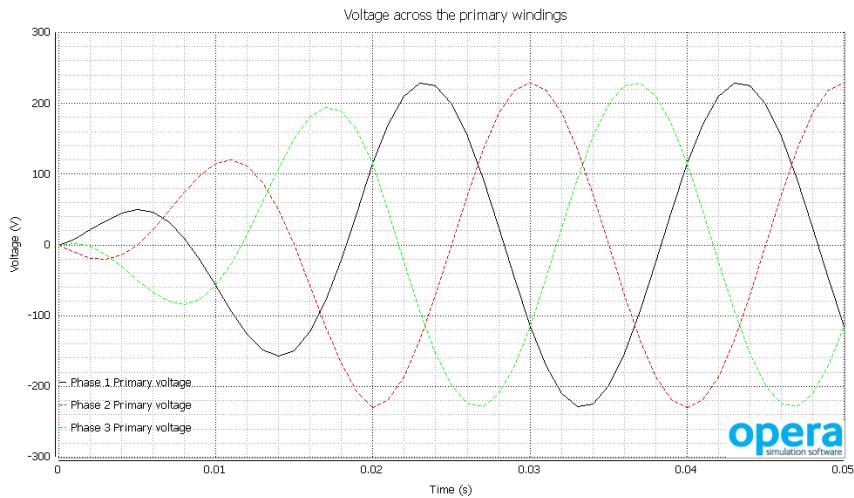


Figure 14.107 Voltage in the primary windings



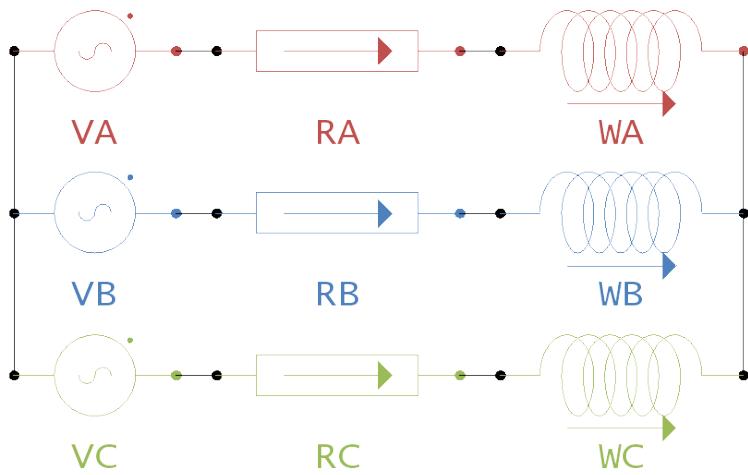
Figure 14.108 Voltage in the secondary windings

## Example 2: 3-Phase Drive Circuit for a Rotating Machine

All of the circuits outlined here relate to 3-phase electrical machines and may be used to control various modes of operation but cover only a small subsection of the possible 3-phase circuits achievable using the Circuit Editor.

## Sine-wave drive

There are many possible ways to define 3-phase sinusoidal drive circuit. The example shown in [Figure 14.109](#) makes use of three separate sinusoidal drive voltage supplies with phase offsets of -60, 60 and 180 electrical degrees. A similar situation could be achieved however with the 3-phase voltage circuit component or 3 functional voltage supplies where the function for each is defined as a sinusoid at a 120 degree phase shift from the others.



*Figure 14.109 Sine-wave - 3-phase AC drive circuit*

## Square-wave drive

The circuit shown in [Figure 14.110](#) is a standard representation of a 3-phase DC driven star-star circuit. In this case the circuit has a DC voltage supply and six switches to maintain two windings on and one off at all times.

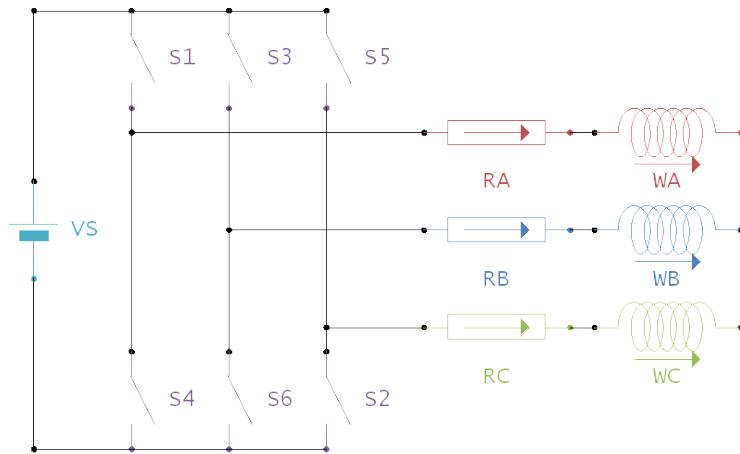


Figure 14.110 3-phase DC drive circuit

The switching states for the 6 switches with electrical excitation angle are shown in Figure 14.111. As a default the **square3d.vfc** is supplied with the switch functions defined as a range of times for which the switch is closed (making use of the system variable **TTIME**, which allows control of the switching with respect to a particular frequency). Figure 14.112 shows the drive profile for the Square-wave drive.

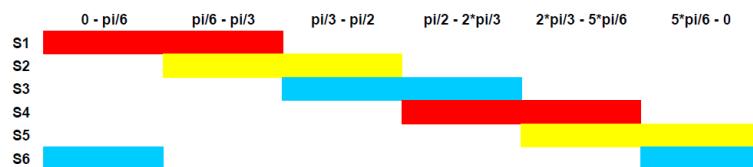


Figure 14.111 Switching states for the square-wave drive

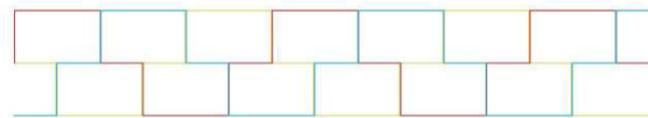


Figure 14.112 Drive profile for the Square-wave drive.

## Stepped Square-wave drive

Stepped square-wave drive makes use of exactly the same star-star circuit layout as the previous square-wave example. The only difference between the two is the switching regime. In this case the switches are maintained so that all three windings are on with two of them splitting the current that flows through the other one. The switching states for the 6 switches are shown in Figure 14.113.

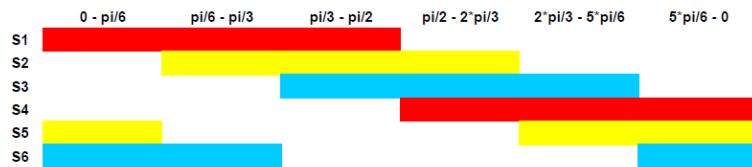


Figure 14.113 Switching states for Stepped Square-wave drive.

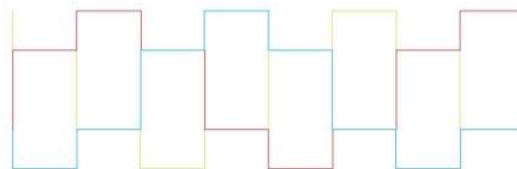


Figure 14.114 Drive profile for the Stepped Square-wave drive

### Note

Each of the conductors specified within the Opera Modeller has its own representation in the list of windings available to the circuit. For this reason it is necessary to modify the 3d circuits to include each of the conductors in a particular series. Figure 14.115 shows an example of a 3-phase, 12-slot double layered permanent magnet machine where each of the 12 conductors within the model are included explicitly.

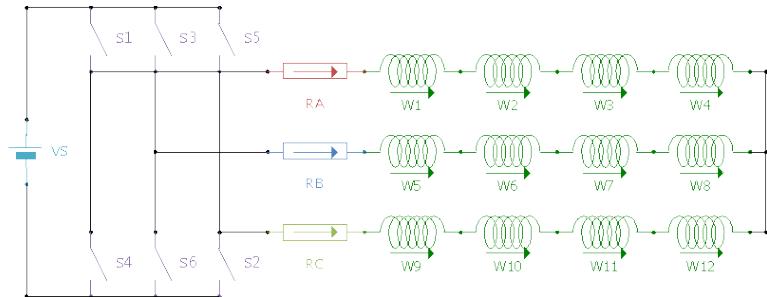


Figure 14.115 Example 3-phase, 12-slot, double layered drive circuit.

### Example 3: Hysteresis chopping

In a chopping-mode control region, the turn-on and turn-off angles are controlled together with the current level. The turn-on angle and turn-off angle are controlled so that the current flows during the rising-inductance or positive-torque producing region. This involves switching the voltage across the phase on and off in such a manner that the current is maintained between some chosen upper and lower hysteresis current levels.

Figure 14.116 shows a voltage driven circuit for a 3-phase 8-pole DC motor which uses a command control file to handle the chopping-mode control. This is done in the Circuit Editor by storing the status of the switch in a user variable and then using this variable to activate the switch.

The example comi file (**Hysteresis\_Chopping.comi**) and circuit file (**Hysteresis\_chopping.vfc**) are provided in the examples folder. Figure 14.117 shows the results for a 3-phase 8-pole DC motor. It can be clearly seen that the currents in the windings are maintained between the upper and the lower current levels as specified in the comi file.

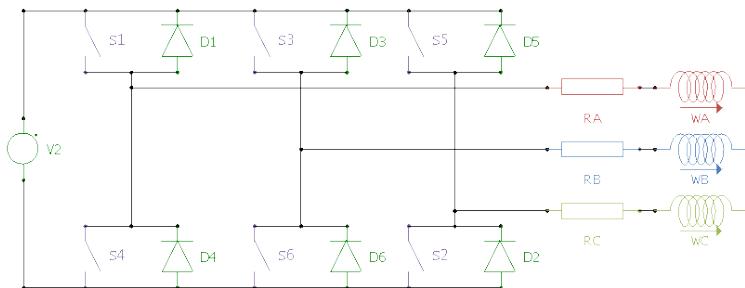


Figure 14.116 3-phase voltage driven circuit

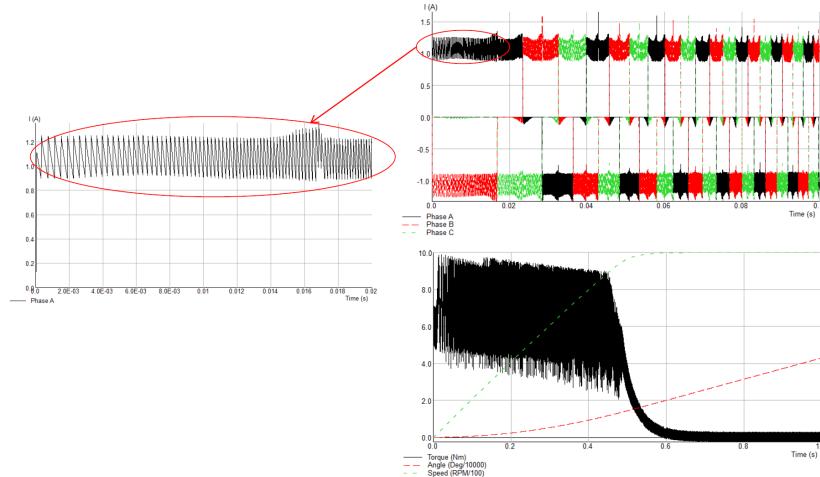


Figure 14.117 Results from the hysteresis chopping circuit

## Example 4: A 12-Pulse Transformer Circuit using a 3-Phase Voltage Supply

A simplified model of a 12-pulse transformer is constructed and analysed using the Opera-3d **Transient Electromagnetic** simulation. The model for this is shown in Figure 14.118.

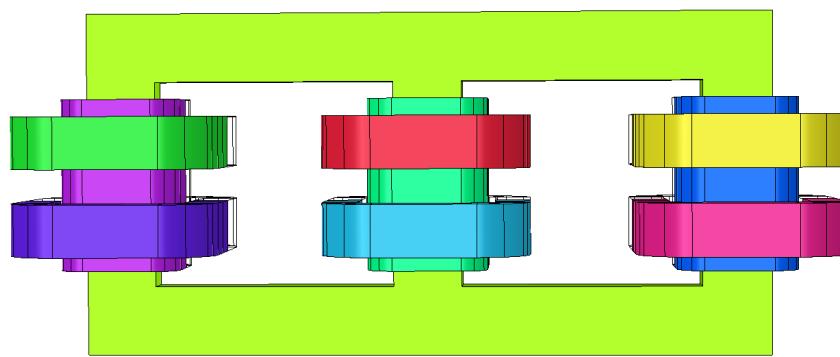


Figure 14.118 A simplified 12-pulse transformer

## Circuit setup

Figure 14.119 shows the circuit layout for the 12-pulse transformer circuit. The transformer is on open circuit with resistor  $R_{load}$  having a very high resistance. The primary windings, connected in delta, have 100 turns each and the secondary have 56 (star) and 97 (delta) turns each. The inductors L1 and L2 are both the same value ( $100 \mu\text{H}$ ). The forward bias of the diodes is 0.3 V.

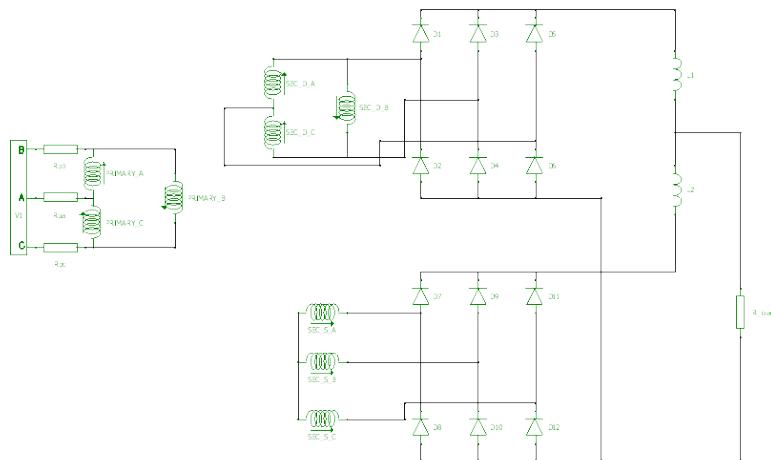


Figure 14.119 12-pulse transformer circuit

## Results

Figure 14.120 shows the voltage across the  $R_{load}$  resistor up to a time when steady-state has effectively been achieved. Note the high transients associated with the sudden switch-on of the circuits. At time = 0 seconds, phases A, B and C will have applied voltages of  $115 \sqrt{2}$ ,  $115 \sqrt{2} \cos(120^\circ)$  and  $115 \sqrt{2} \cos(240^\circ)$  V respectively.

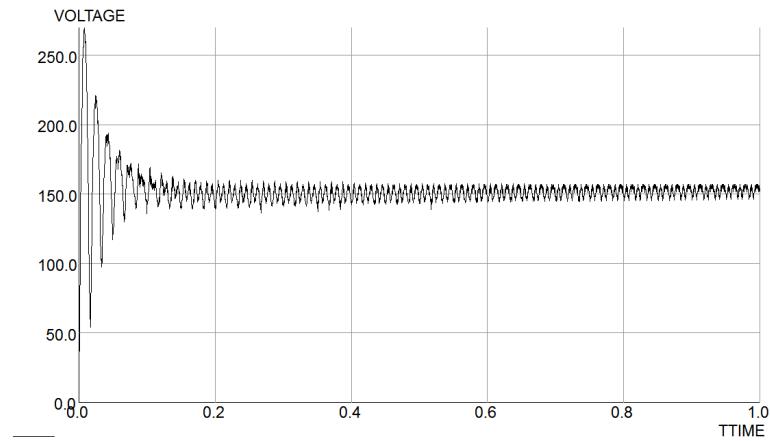


Figure 14.120 Voltage on resistor  $R_{load}$

Figure 14.121 shows the final 100 ms of the  $R_{load}$  voltage characteristic in more detail. The 12-pulse nature can be clearly seen.

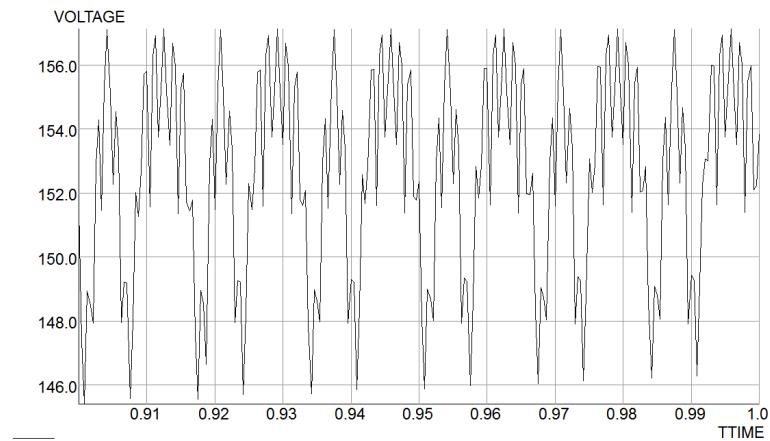


Figure 14.121 Detail of voltage across resistor  $R_{load}$  from 0.9 s to 1.0 s

## Bulk Conductors

---

### Introduction

A common feature of pulsed power systems is the existence of high and rapidly changing magnetic fields, which lead to current redistribution and the generation of eddy currents in local conducting structures. These structures include any coil windings, which, Opera, are often represented by Biot-Savart conductors. In these, the current density is constant across the conductor cross-section, and is not affected by the local field environment. Hence they cannot support eddy currents. Instead, bulk conductors must be used if it is necessary to account for the effects of current redistribution in the windings themselves - such effects including localized increases in Joule heating and varying winding impedance.

In Opera's Transient EM solver, bulk conductors may be driven by voltage or current boundary conditions, or they may be connected to circuits. In the last of these, the field and circuit equations are solved in a coupled simulation and, optionally, optimization, allowing, for example, the drive circuit to be designed to be tolerant to the effects of varying impedance.

This note outlines the use of circuit connected bulk conductors in the simulation of a pulsed magnet, known as a septum magnet, of the type often used to control particles trajectories as they enter or leave an accelerator storage ring.

### Operation

A schematic of a septum magnet is shown in [Figure 14.122](#).

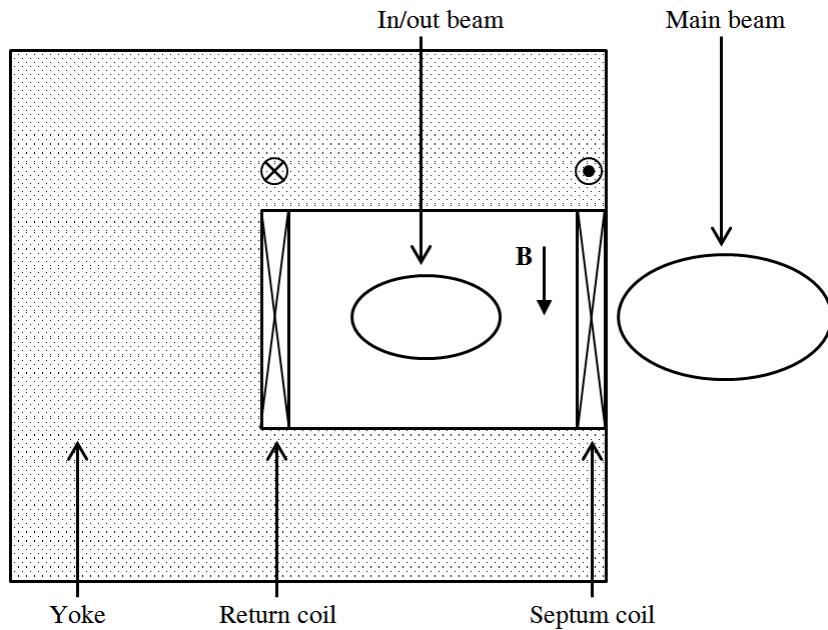


Figure 14.122 Septum magnet schematic

The in/out beam propagates in a channel within the magnet; and is separated from the main beam-line by the outer part of the winding, known as the septum, from which the magnet gets its name. The coil is wound within a high permeability yoke, which completes the magnetic path. This arrangement provides a uniform field within the coils, with a low field in the region outside the septum, ensuring minimal perturbation of the main beam. To allow passage of the in/out beam, the coil is split horizontally in two, with the ends of each bent vertically out of the beam path, as shown in Figure 14.123.

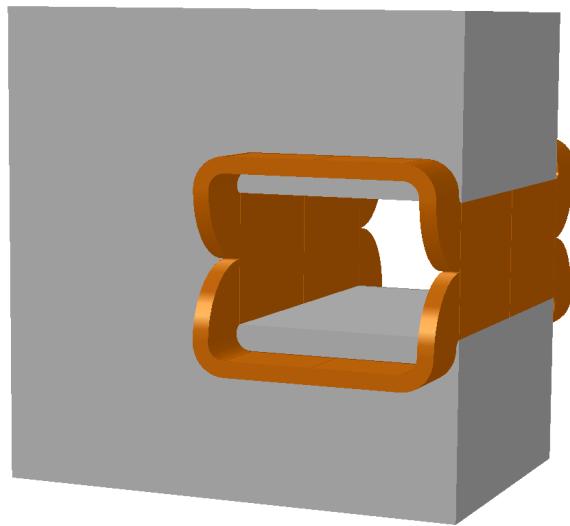


Figure 14.123 Septum magnet geometry

When feeding particles into the accelerator ring, the magnet is pulsed as the input beam travels through, causing a deflection to bring it closer to the path of the main beam. Final alignment is then performed by a kicker magnet. On extraction, the kicker magnet deflects part of the main beam through the septum magnet, which then aligns the trajectories to the extraction beam line.

In principle, the septum magnet could be operated in dc mode. However, the need for high field strengths, of the order of 1T, and the relatively thin dimension of the septum coil mean that the coils dissipate considerable power. The problem of dissipation is exacerbated by poor thermal transfer from the septum location in the beam line. Since beams are injected or extracted in bunches, the magnets may be operated in pulse mode, limiting the total energy requirement and thermal load.

## Model

The full geometry is shown in [Figure 14.123](#). Advantage has been taken of two-fold symmetry to reduce this to a 1/4 model, as shown in [Figure 14.124](#). The winding consists of a single shaped copper bar, with one turn in each half magnet to give the magnet a low inductance and, hence, a fast rise time for the current and field. Appropriate boundary conditions are specified to imply the symmetry. For the purpose of this example, it has been assumed that the permeability of the steel is isotropic and linear with  $\mu_r = 6400$ , and that it is constructed from laminations such that it is non-conducting.

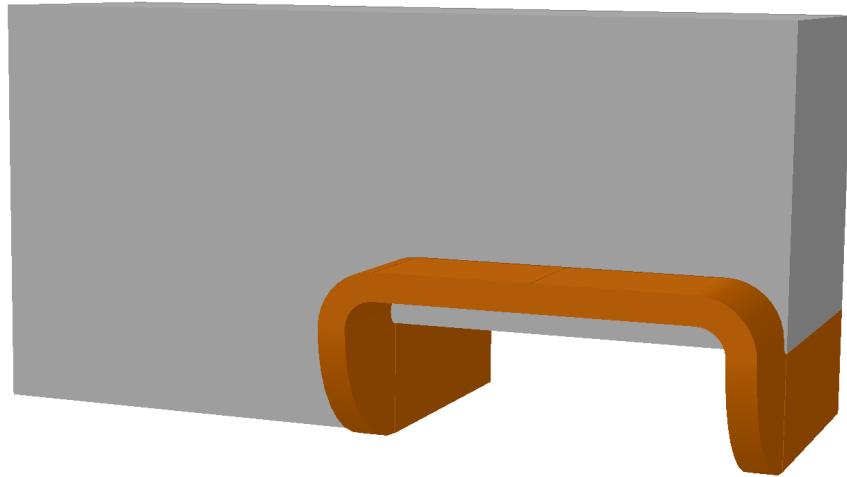


Figure 14.124 Model with two-fold symmetry applied.

The circuit connection to the coil geometry is made at the coil faces that lie in the symmetry plane on  $z=0$ . In the model, these have been labelled as boundary condition faces,  $v0$  and  $v1$ . These have been assigned as the positive and negative **Terminal of bulk conductor**, and the **Circuit element name** set to **b1**, as indicated in Figure 14.125.

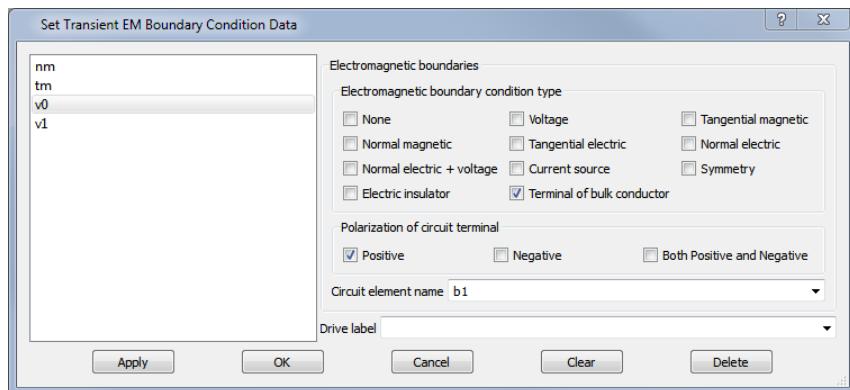
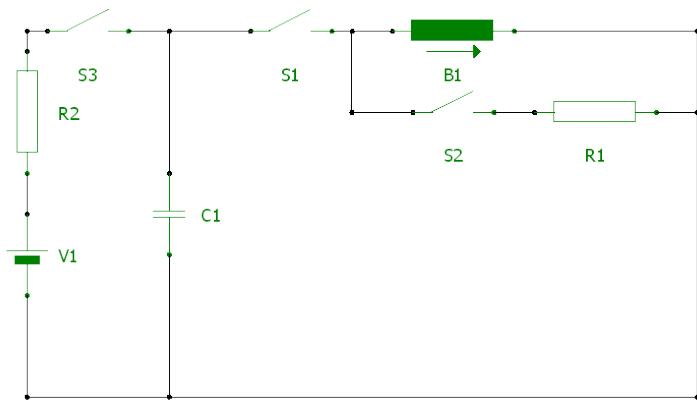


Figure 14.125 Connecting the magnet terminals to the circuit.

The circuit created to give an appropriate pulse excitation to the magnet is shown in Figure 14.126, where the magnet winding is depicted by its circuit element name, B1.



**opera**  
simulation software

Figure 14.126 Pulse drive circuit.

## Circuit Operation

The pulse excitation is generated by changing the state of the circuit switches at defined times during the simulation. Specifically:

C1, is initially charged to V1 voltage of 200 V. At time zero, all the switches are open

When time > 0, switch S1 closes capacitor, C1, discharges through B1. The switch, S1, remains closed for 0.5 ms

At t = 0.5 ms, switch S1 opens and switch S2 closes. Coil current flows through the dump resistor, R1

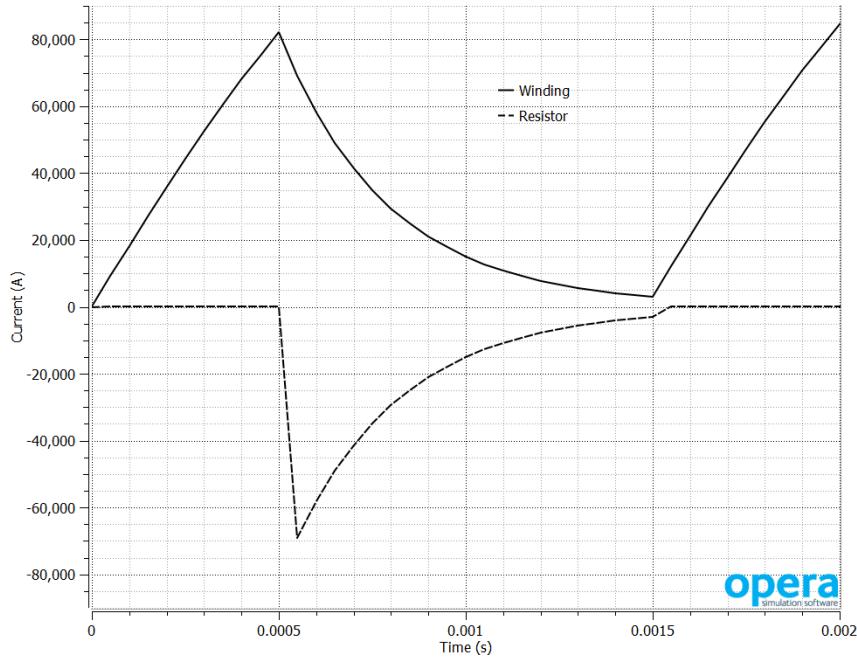
At t = 1 ms, S3 closes allowing C1 to re-charge for 0.5 ms

The cycle then repeats from t = 1.5 ms

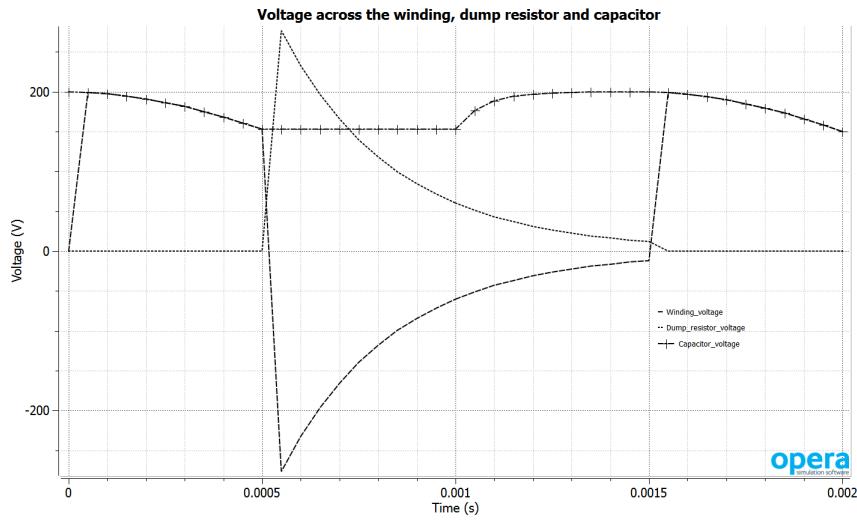
## Simulation Results

The simulation on this example model is run over a time of 0 to 2ms, with a time-step of 0.1 ms. The current and voltage on each circuit element is logged.

**Figure 14.127** and **Figure 14.128** show the current and voltage waveforms in the circuit at later times, as generated during the transient simulation.



*Figure 14.127 Winding and resistor current.*



*Figure 14.128 Winding, resistor and capacitor voltage.*

When time  $> 0$ , switch S1 closes allowing the capacitor, C1, to discharge through the bulk conductor, B1. The switch, S1, remains closed until 0.5ms, by which time the current in the bulk conductor, B1,

has reached 80 kA. By this time, the voltage on the capacitor, C1, has reduced to almost 150 V. After this, switch S1 opens and switch S2 closes. The current in the coil now flows through the dump resistor, R1, and, consequently, reduces.

At time = 1 ms, switch S3 closes allowing the capacitor, C1, to re-charge. The capacitor, C1, approaches 200 V again after another 0.5 ms (time = 1.5 ms) and the cycle then repeats.

Switch S1 closes and the other switches (S2 and S3) open. After a further 0.5 ms (time = 2.0 ms), the current in the winding has reached about 85 kA and the capacitor voltage has once again reduced to about 150 V.

The inductance, resistance and power dissipation in the winding can be extracted from the simulation results; a simple command file may be used to achieve this, for example:

```
$open 1 'bulk_cond_data.txt' overwrite -redi
$format 1 exp 25
$assign 1 1 1 1
$do #case 1 21
simulation case=%int(#case)
/
ENERGY ACTION=RESET
ENERGY ACTION=ADD, | ENERGY LABEL=COPPER
ENERGY ACTION=INTEGRATE ADAPT=NO
/
$write 1 ttime power power/b1_i^2 (b1_v-power/b1_i)/b1_didt
$end do
/
$close 1
```

The **ENERGY** command calculates the power dissipated in materials with material label of **COPPER**, ie in the winding. The `power/b1_i^2` term calculates the resistance of the winding; `(b1_v-power/b1_i)/b1_didt` gives the inductance.

The results obtained from the example model are shown in [Figure 14.129](#) to [Figure 14.131](#).

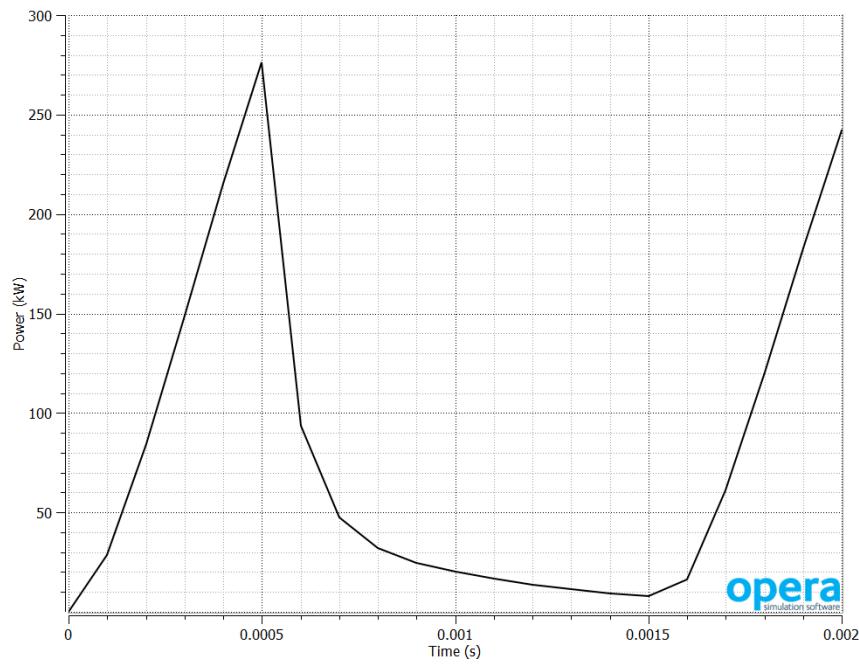


Figure 14.129 Power dissipation in the winding.

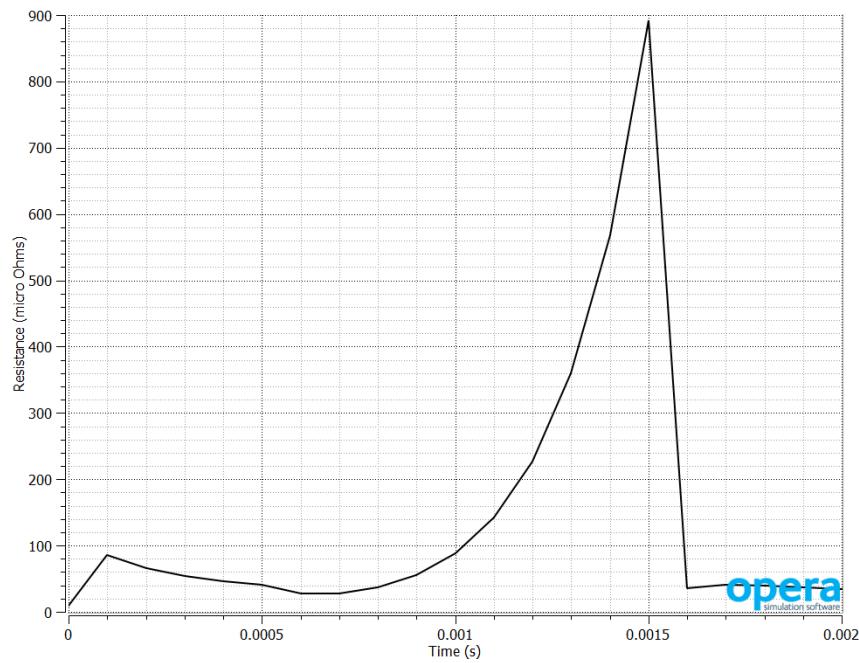


Figure 14.130 Winding resistance.

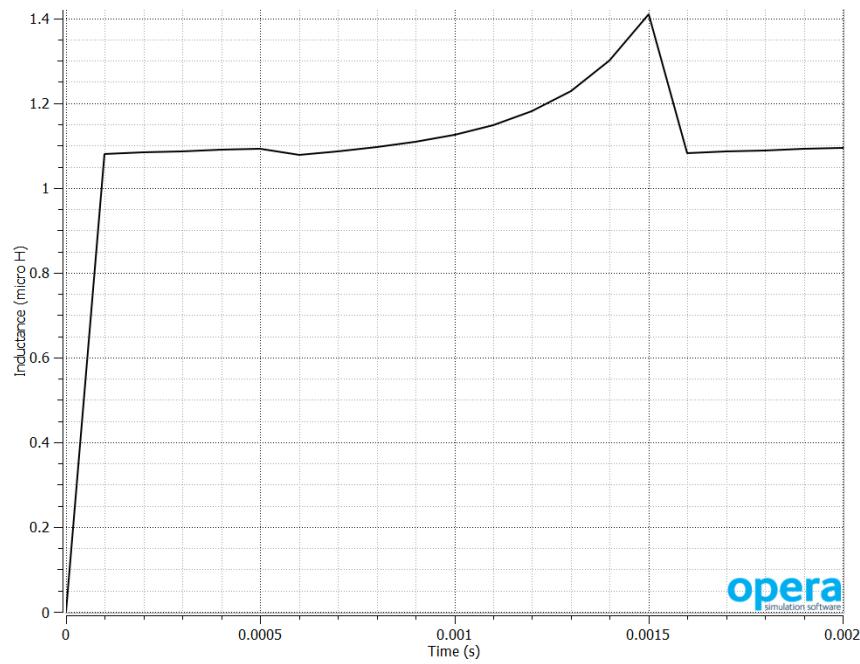
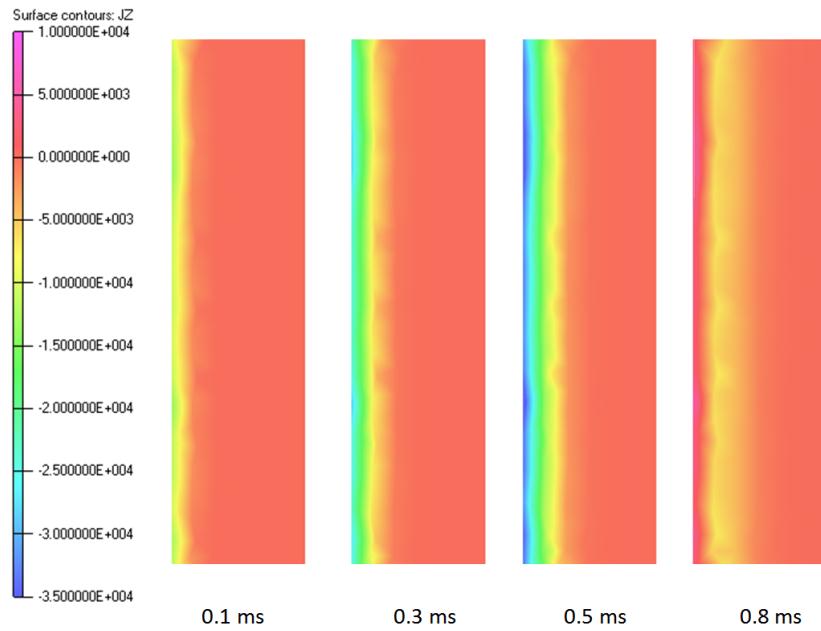


Figure 14.131 Winding inductance.

## Discussion

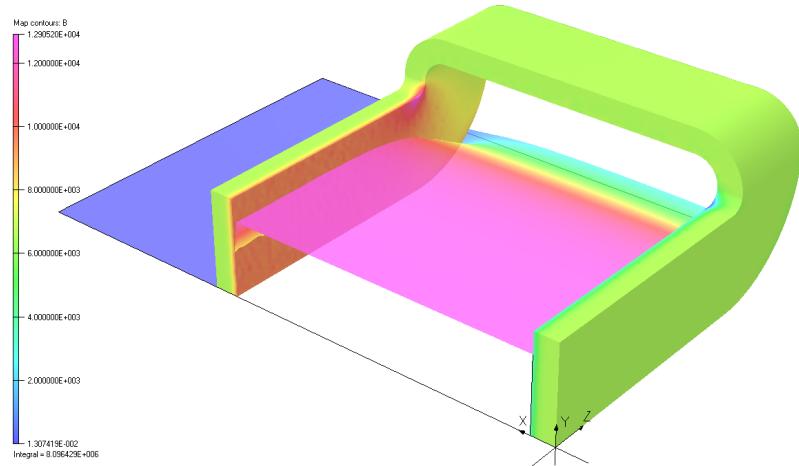
From the figures above, it can be seen that the redistribution of the current in the winding significantly affects the resistance of the winding - with greater than an order of magnitude variation seen during the excitation cycle - while the change in inductance is limited to a few 10s of percent.

The current redistribution is illustrated in Figure 14.132, which shows the z-component of the current density over the cross-section of one of the septum windings at various times during the simulation. In this figure, the left edge is on the inner surface of the winding. As can be seen, the current concentrates progressively towards the left edge of the conductor up to a time of 0.5 ms.



*Figure 14.132 Current density at various times in the drive cycle.*

Another consequence of the eddy current is the effective shielding of the field beyond the septum. Figure 14.133 shows a field map of the flux density,  $B$ , within the winding and into the region of the main beamline.



*Figure 14.133 Magnet current and magnetic flux density.*

Both the uniformity of the internal field and the screening can be seen more clearly on a plot of  $\log B$  along the x axis.

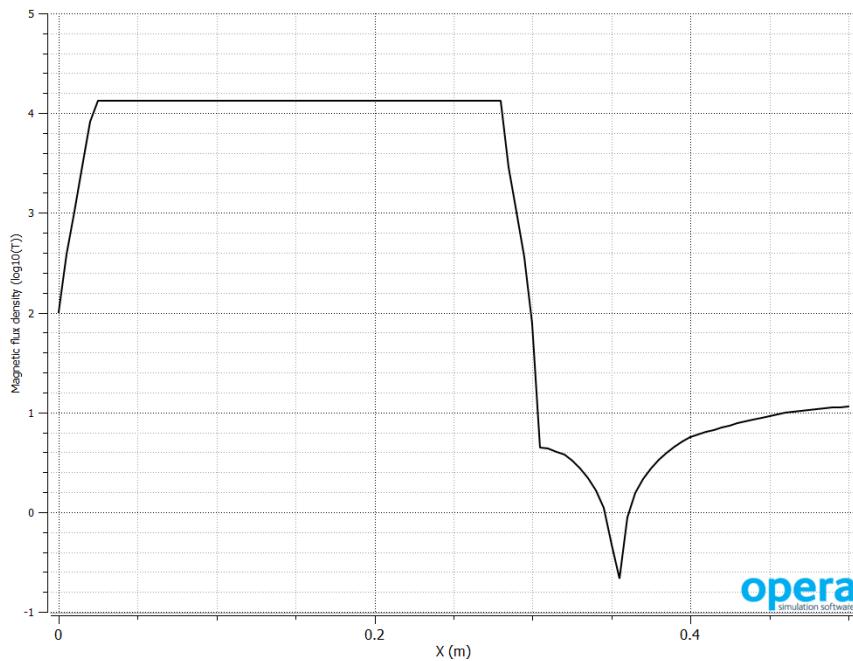


Figure 14.134 Magnetic flux density along  $x$ .

## Restarting Simulations

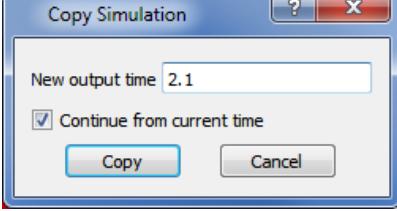
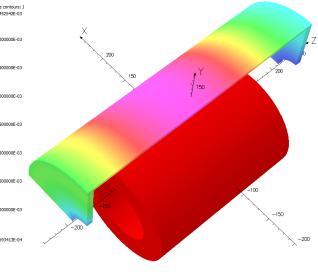
### Introduction

There are a number of occasions where it would be advantageous to perform a simulation starting from the result of an existing simulation. An example might be when a steady-state result is required from a transient simulation. If the simulation had not been run for long enough to allow the start-up transients to decay, then additional time-steps might need to be performed. Continuing for this additional time from the known solution at a previous time would clearly be more efficient than repeating the simulation from time zero.

A similar example would be to allow further investigation of the late-time behaviour of a system excited by a pulse. For instance, the radiation screen discussed in "[Radiation Screen Transient EM Example](#)" on page 196 was excited by a pulse that fell to zero at 1.8 seconds and the last simulation output time was set to 2 seconds. At this point the fields and currents in the screen had not decayed completely – continuing the simulation would allow the late-time behaviour to be explored.

Opera allows the continuation of such simulations - either from a previously solved case or from the initial conditions; this function is termed a 'restart'.

The restart feature is enabled by the **COPYCASE** command, which may be issued either on the command line or by selecting the restart toolbutton . As an example, run the radiation screen model mentioned above (the model, **radiation.opc**, is available in the Opera-3d examples folder). When complete, open the database and load the last simulation, which is at 2.0 seconds.

 <b>Copy Simulation</b>	<p>Enter a new output time later than the current time. With the <b>Continue from current time</b> box checked, the new simulation will start from the result of the previous simulation.</p>  <p>Select <b>Copy</b> to create the new simulation and to close the dialog. The new simulation may be run after the database has been unloaded from the Post-Processor.</p>	
---	---	--

The options displayed in the restart dialog will depend on the type of simulation being performed. For a transient simulation, for instance, the dialog allows an additional time-step to be entered and gives the option to either continue from the currently loaded time or to start from time zero.

Having entered the data and selected **Copy**, the additional case will be written to the database. The database must be unloaded from the Post-Processor before the simulation can be restarted. Using this method, only one additional simulation – with one time-step in this case - may be added at a time. As will be mentioned later, a more advanced approach that allows enhanced flexibility for transient simulations would be to use a control command file.

It is important to note that the solutions to additional simulation cases will be added to the database – they will not overwrite existing solved cases. Newly added cases will be written to the end of the database; this can result, for example, in the output time of simulations not increasing monotonically from case to case.

## Applicable simulation types

The restart feature is not only available for transient simulations. However, the restart function has different capabilities depending on the type of simulation being performed. This is detailed below.

### Transient simulations

Output time-steps may be added. The user can choose whether to continue the additional time step simulation from the loaded simulation or from time zero. If the former, the new output time must be at or later than the loaded simulation.

### Static simulations

New values for the drive scaling factors may be added.

### Harmonic simulations

New values for the simulation frequency may be added. The user can choose whether to use the loaded simulation as a starting point for the solution of the new frequency case.

### Charged Particle

Further iterations may be added and the non-linear tolerance can be modified. The extra iterations will continue the simulation from the last iteration of the original simulation.

For the remaining simulation types (Electrostatic, Static Thermal, Static Stress, Current Flow, Modal EM, Modal Stress) the loaded simulation case can be duplicated as an unsolved case by the **COPYCASE** command. However the model properties for the new simulation may be redefined as described below.

When a simulation is restarted, either from the loaded simulation or from the initial conditions, only those parameters that are definable in the **COPYCASE** command as listed above, are affected. Modifications may also be made to properties or data that are either read by the particular solver during the solution process or have been defined as functional properties.

The first of these is particularly relevant when a control command file is used in transient simulations. Key variables, such as time-stepping behaviour, can be set within control files and since these files are read by the solver several times at each time-step they may be used to modify simulation behaviour during restarts. This provides additional flexibility to the restart function. Examples of using control command files for time-step control are given in "[Example Control File](#)" on page 705 and the section entitled 'Transient Analysis Logging and Control' in the ***Opera-3d Reference Manual***.

Functional material properties may be used in any simulation type, not just in transient. As described in the **MATERIALS** command section of the *Opera-3d Reference Manual*, properties can be defined as functions of position (X, Y, Z) or can be determined from element or nodal vectors and added into the analysis database using the **TABLE** command. The original properties assigned to a particular material label will have been defined by using the **\$FUNCTION** command operating on the table file that contains those properties. The same material label may be redefined by issuing the **\$FUNCTION** command again in the Post-Processor before restarting a simulation. If the table file includes data that is dependent on the results from the current simulation the file should first be updated. Alternatively the function may be associated with a differently named file. In either case the new simulation created by using the restart feature will use the modified properties.

# Machine Analysis using Motional EM solver

## Introduction

The first part of the application note shows how to set up a **Motional Electromagnetic** model of an electrical machine.

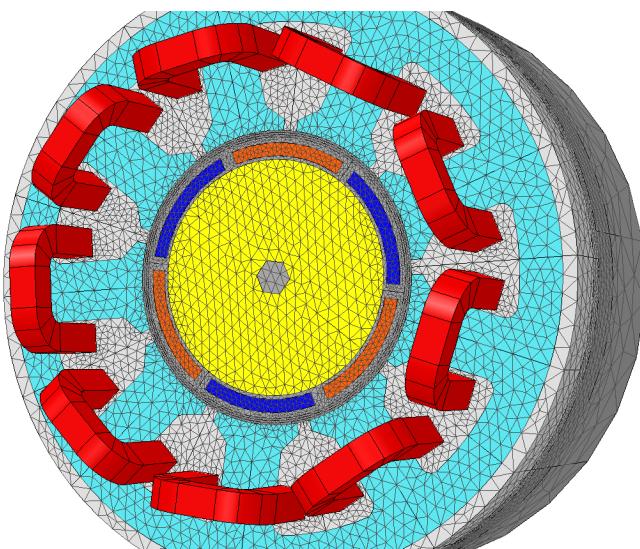


Figure 14.135 Model of a generator

A **Motional Electromagnetic** model can include permanent magnets, and it can be current driven (using Biot-Savart conductors) or voltage/current driven (using filamentary or meshed conductors connected to circuits). Both rotating and linear machines can be modelled and analysed.

The **Motional EM** analysis is based on the **Transient Electromagnetic** analysis, and a **Motional EM** database is created in much the same way as a **Transient Electromagnetic** one. There are, however, a few extra considerations which are detailed in this application note.

To adapt the concepts of this example to a specific machine, please refer to the documentation in the *Opera-3d Reference Manual* (the description of the **Motional Electromagnetic** solver and the section "Transient Analysis Control").

The particularities of modelling and analysis of linear motion devices as **Motional EM** models are presented in the [Linear Actuator Example \[page 698\]](#).

## Symmetry

Models built with the Modeller can include periodicity conditions which reduce the geometry for analysis to the minimum symmetric section, for example to 180 degrees for a 2-pole machine or 90

degrees for a 4-pole machine. A simple example exploiting this is described in [Periodic Models \[page 694\]](#).

Symmetry in the axial direction, where the magnetic field is tangential or normal on the centre plane of the machine, is also possible.

## Group Labels

In a **Motional Electromagnetic** model all cells and conductors must have a group label. Therefore the **CELldata** and all the **CONDUCTOR** commands have a parameter called **GROUPLABEL**.

The aim of a group label is twofold. It allows the independent motion of different parts of a model and it also identifies gap regions in the model.

Any names can be used for group labels except for **GAP**, which is reserved as the label for the gap region. The first two characters of a group name are used in the variable names holding values of angle, acceleration, etc. Names, therefore, should be unique in the first two characters.

A check is performed at the "Create Database" step and a warning is issued to the user if a cell or a conductor exists without a group label.

## Gap Regions

Gap regions provide a buffer between different sections of the model. For example in a motor model we may have two non-gap regions, a rotor region and a stator region. To allow the free rotation of the rotor section, and to maintain the continuity of the finite element mesh, the gap region is remeshed after each update of the rotor position.

- The gap region must have the group label **GAP**.
- The gap region must be assigned **TOTAL** potential.
- The gap region can consist of several cells.
- Cells with a particular group label can only touch cells with the same group label or cells with the **GAP** label.
- Cells touching the **GAP** should always have material label **AIR**, meaning that a layer of air should be added between the **GAP** and any other non-air cells. Otherwise, the software will issue a warning message informing the user that fields may not be displayed correctly on cells touching the gap.

[Figure 14.136](#) shows cells with the label **GAP** in the model from [Figure 14.135](#). In this example the gap has been cut into several rings. Some of the rings have been picked, but only in order to highlight them. The topology of this **GAP** matches configuration B which is discussed below in [Figure 14.137](#).

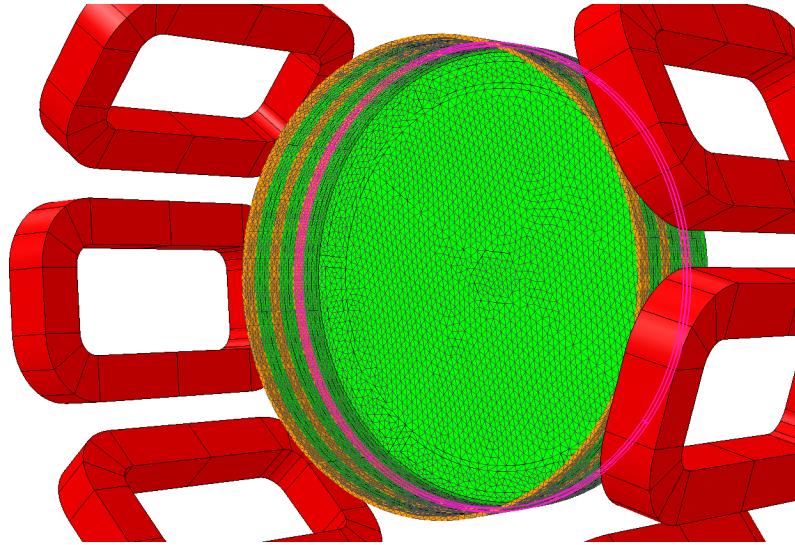


Figure 14.136 Cells with the label GAP

Some common shapes of gap regions for radial flux and for axial flux machines are discussed in the following examples. The examples are using symmetry in the axial direction, therefore only 1/2 of each machine is modelled in the Z-direction.

## Radial flux machine

Two configurations of radial flux machines are shown in Figure 14.137. In Configuration A the gap region is a tube and extends on both sides to the outer boundary of the Finite Element mesh.

In Configuration B, the gap region is a short cylinder with closed ends. The rotor region only consists of the rotating parts. The model of a generator presented in Figure 14.135 uses this shape of gap. For efficient meshing the gap region should consist of several cells - a disk and several rings, see Figure 14.136.

Note that the **GAP** in Configuration B is touching the Z-axis. For models with a rotational symmetry  $\geq 3$  this should be avoided to run the **Motional EM** analysis as efficiently as possible. This issue is discussed in more detail later.

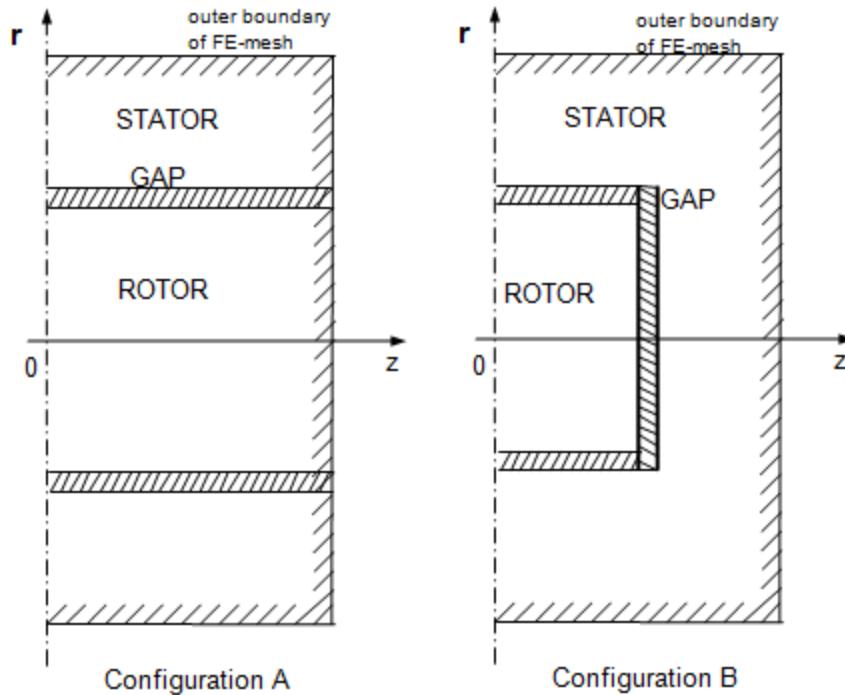


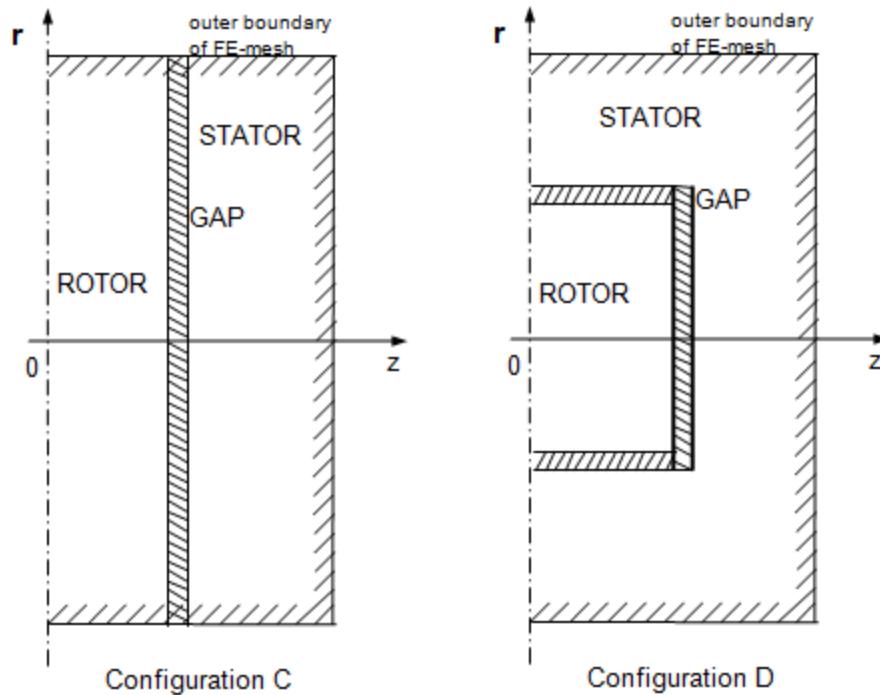
Figure 14.137 Two common gap configurations for a radial flux machine

## Axial flux machine

Two configurations are shown for an axial flux machine in Figure 14.138. In Configuration C the gap region extends to the outer boundary of the mesh on both sides.

An axial flux machine can be modelled using the same configuration as the B configuration of a radial flux machine, with the difference that the flux lines that produce the torque are parallel to the Z-axis (Configuration D in Figure 14.138).

Note again that in both cases the **GAP** is touching the Z-axis.



*Figure 14.138 Two common gap configurations for an axial flux machine*

### Gap not touching the axis

If the gap region is not touching the Z-axis, the **Motional Electromagnetic** solver creates the gap mesh in only one period of a rotationally symmetric rotating machine. For a machine with 6-fold rotational symmetry, for example, the gap mesh is reduced by a factor of 6. This reduces the execution time, memory usage and the size of the database.

The full gap mesh is used in models where the gap touches the Z axis.

A common practice in the modelling of large wind turbine generators is to omit the inner radius from the Finite Element mesh and model only a sector of an annular ring, like in [Figure 14.145](#). This is a simple way of avoiding the gap touching the axis. It is shown here as Configuration E.

Configuration F shows another option to avoid the gap region touching the axis.

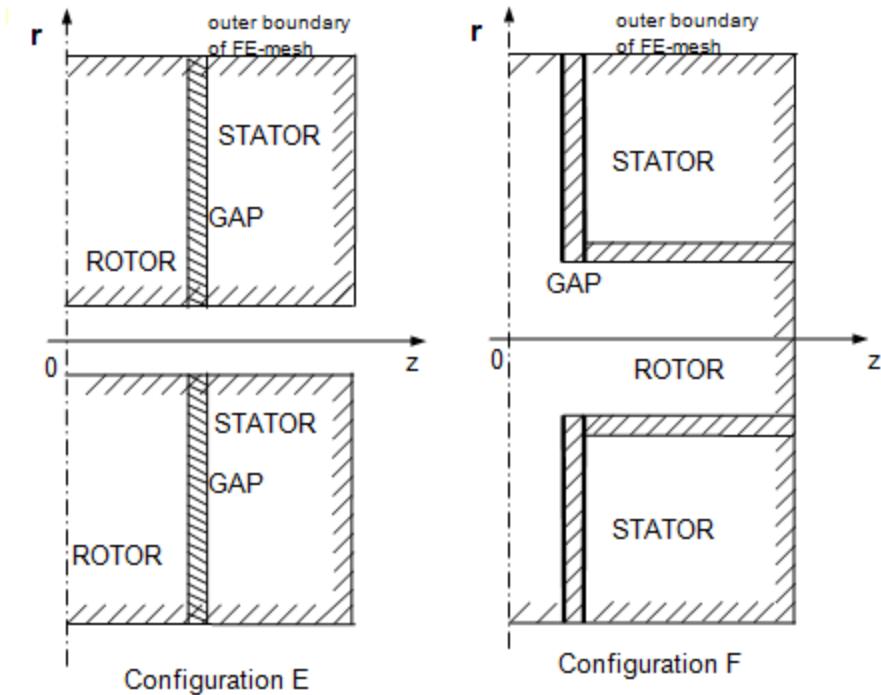


Figure 14.139 Configurations where the gap regions do not touch the Z-axis

### Other gap configurations

Configuration G in Figure 14.140 gives further ideas on the structure of a gap region. It shows a more complicated case with two rotor regions rotating at a different speed (such as in a magnetic gear), and a more complex shape of some cells of the gap region. Note that for the cells of the gap region

- a face between two cells must be a planar or a cylindrical face;
- a face on a boundary of the gap must be a planar or a cylindrical face around the axis of rotation.

Group labels have to differ in the first 2 characters. Therefore the names `R1_ROTOR` and `R2_ROTOR` have been chosen.

The last configuration H doesn't model the bearing of the rotor; it is only shown as an example of what can be modelled.

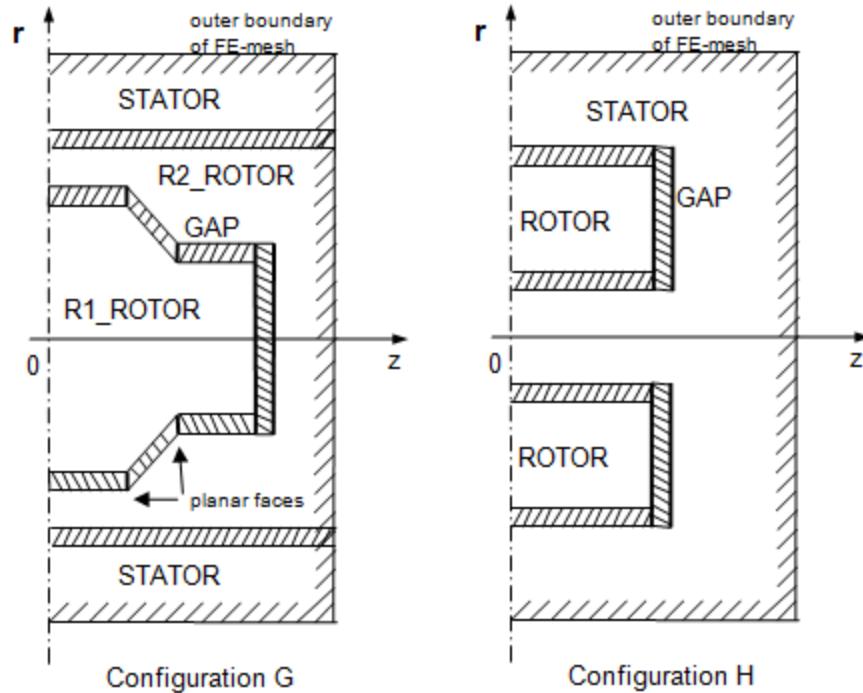


Figure 14.140 Further gap configurations

### Tips on gap meshing

Gap meshing can be made more reliable using the following suggestions.

- As explained above, the gap region, which is remeshed at every step, can consist of several cells (for example, for a radial flux machine a cylindrical annulus cut into a number of axial rings plus an end cap disk - see [Figure 14.136](#)). Even multiple gap regions are allowed as shown in [Figure 14.140](#) - they only have to have the group label **GAP** indicating that these cells have to be remeshed.
- Keep the shape of the gap region simple.
- Minimize the volume of the gap region.
- Remeshing problems may occur if the ring is very thin (radially) or rather long (axially). Such problems can be avoided by cutting the gap region into several short rings (axially) thereby avoiding internal faces.
- Wherever possible, a prism mesh is preferable to a tetrahedral mesh in the gap, since only a 2d cross section has to be remeshed (which is then copied into the third dimension). However, a prism based mesh is only possible in cylinders as in Configuration A and not in disks as in Configuration C.
- It is best if the gap region does not have many internal nodes. This can be achieved by defining element sizes for the cells in the **GAP** which are a larger than the thickness of the **GAP**.

- When the **GAP** region contains a disk (like Configuration C in [Figure 14.138](#)), the default cylinder options will create edges on the curved faces between the two ends of the cylinder. These edges would connect stator cells and rotor cells, which is not allowed.

To avoid this, it is recommended to build the cylinder with the command

```
CYLINDER ... SIDES=1
```

and not

```
CYLINDER ... SIDES=2
```

It is also possible to delete connecting edges manually.

For further advice on meshing see [Meshing in the Modeller \[page 541\]](#).

## Mosaic meshing

The algorithm for remeshing the **GAP** region requires a triangular mesh on all the faces that span between moving and stationary cells. Hence, the volume of the **GAP** cells can be meshed with either tetrahedral or prism elements (but not hexahedral elements).

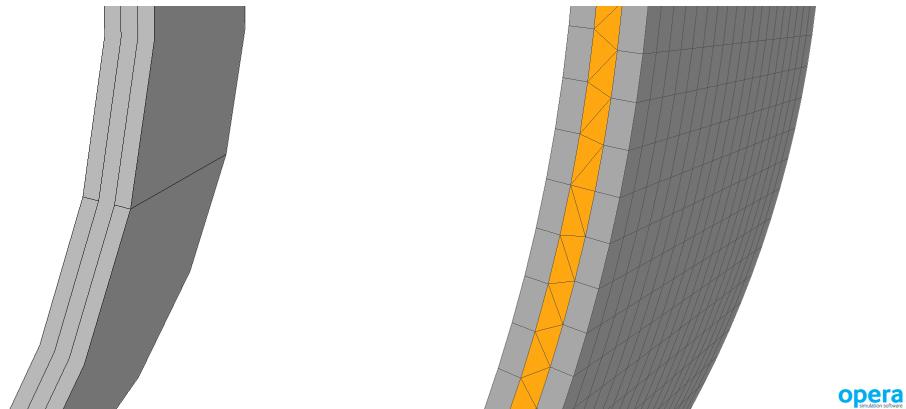
Cylindrical **GAP** regions where the end faces lie in the xy-plane and which are by default hexahedral meshable are automatically downgraded in order to create prism elements. The correct orientation of the prism elements can be obtained in other topology **GAP** regions by adding two extra edges on the azimuthal faces of the **GAP** region, parallel to the rotation axis. This can be achieved, for example, by cutting the air layers adjacent to the **GAP**, but not the **GAP** itself (see [Figure 14.141](#)).

The control of the mesh type should be done by explicitly applying the Mosaic mesh preference on the gap cell(s).

For topologies which contain both cylindrical and disk **GAP** regions (e.g. Configuration B), prism elements can still be used in the cylindrical cells of the mesh, while the end disk region has to be meshed with tetrahedral elements. In this case, in order to avoid a regular mesh on the azimuthal faces of the disk, this cell should have the Tetrahedral mesh preference explicitly applied to it.

Cells adjacent to a prismatic meshable **GAP** can be meshed with any element type. However, it is recommended that these are meshed with prism or hexahedral elements as well, in order to avoid the creation of pyramid elements close to the **GAP**.

In order to avoid noisy output characteristics (especially torque) due to the 'in-step' reconnection of the mesh nodes, the adjacent cells should have a slightly different mesh size (see [Figure 14.141](#)).



Gap region and adjacent air layers

Prism mesh elements in gap (selected) and hexahedral elements in adjacent air layers

Figure 14.141 Setting up a prism gap cell

## Motion Control

Motion is controlled through the use of a command file that is executed at various times through the course of a time-step. A more detailed description of the workings of the command file is given in the Opera-3d Reference Manual. An outline of the **Motional EM** specific features is presented here.

The name of the command file is important. There are two choices:

- **a specific name:** If the **Motional EM** analysis database is called ***test.op3*** then the command file associated with this database must be named ***test\_carmen.comi***. The first part is derived from the name of the analysis database and the second part identifies it as a **Motional EM** command file.
- **a generic name:** A control file called ***carmen\_control.comi*** can be used with any **Motional EM** database in the same folder which does not have a specific control file.

Control files must exist in the same folder as the database files they are used with. Command file control is available in **Motional EM**, **Magnetization**, **Transient Electromagnetic**, **Quench** and **Transient Thermal** and the names of the control files for each analysis type must include the appropriate program name: ***carmen***, ***demag***, ***elektratr***, ***quench*** or ***tempotr***.

Each section of the model (the sections identified by the groups labels) can be controlled independently. For each section, except for the gap, parameters must be set to determine how the motion should be controlled.

Consider a model with three sections having group labels **STATOR**, **ROTOR** and **GAP**. In order to specify the rotational speed of the **ROTOR** section, the command file should consist of only two lines:

```
$STRING RO_MOTIONCONTROL SPEED
$CONSTANT #RO_OMEGAZ PI/10
```

The first line sets the method of control by defining the string variable `RO_MOTIONCONTROL` and setting it equal to `SPEED`. The `RO_` at the start of the variable indicates that it refers to the `ROTOR` section. Only the first two characters of the group name are used. Setting this variable to be the string `SPEED` indicates that the rotational speed of the section is specified.

The second line defines the speed of rotation of the section. Again, `RO_` indicates that it relates to the `ROTOR` section. This value is defined in radian s<sup>-1</sup>.

Other control methods are:

- **STATIONARY** - the section is not moving. No additional information required.
- **POSITION** - allows the position of the section to be set directly. Here we must define a variable, e.g. `#RO_THETAZ`.
- **ACCELERATION** - controls the acceleration. We must define the corresponding control variable, e.g. `#RO_ALPHAZ`.

Motion of the gap region cannot be defined. The gap region is there to provide a buffer between the sections of the model that are moving.

## Control File and Command File Editor

When started by the Opera-3d/Modeller and when the current analysis type is **Motional EM**, the Command File Editor has an option in the **File** menu to create a **New Carmen control file**. The dialog presents options for each group in the model (except GAP) to define the motion type and associated parameters. The command file is then displayed in the editor window and can be changed as required.

## Logging of Variables

Variables can be logged in a separate output file as the analysis progresses. The names of system variables which are available depend on the names of the groups. Each group generates a set of variables, with the stem of the names starting with the first 2 characters of the group label i.e. a group called `rotor` generates variables:

`RO_THETAZ`: Angular position

`RO_OMEGAZ`: Angular speed

Additionally, if the model contains circuits, currents and voltages can also be logged. Up to 80 variables can be logged. For more information, see the *Opera-3d Reference Manual*.

## Generator Example

The example given is a generator with a 3-phase stator winding with 9 coils. On the rotor there are 6 magnets (3 poles). The dimensions are typical but this is not a real machine. The model is provided as an **opc** file in the **3D** sub-folder of the Opera examples folder. A modified version is also provided, in which prismatic elements are used in the **GAP**.

## Periodic Models

Most rotating electrical machines exhibit geometric symmetry - the same physical structure is repeated a number of times around the 360 degrees. (The most common exception to this is induction motors, where the number of rotor slots is deliberately chosen as a prime number to reduce cogging torque.) Consequently, it is desirable to use the **Motional EM** solver to only solve the minimum geometry necessary.

However, either because of armature reaction or as the rotor rotates to different angular positions or both, it is generally not possible to impose boundary conditions that define the magnetic field as tangential or normal to any boundary surface parallel to the machine axis. But it is possible to say that the behaviour of the field on a particular boundary surface is the same as it is on an equivalent boundary surface one period of the geometry away.

Opera-3d allows this type of behaviour to be modelled using periodicity boundary conditions. Two types of periodicity can be used: **positive symmetry** is where the field is exactly the same at each boundary surface and **negative symmetry** is where the magnitude of the field is the same but the sign is opposite. For example, in a 4-pole synchronous machine, positive symmetry will usually exist over 2 pole pitches and negative symmetry over one pole pitch (unless there is a very unusual armature winding).

### Periodicity

Figure 14.142 shows a simple 4-pole machine with only the rotor windings defined (as would be required for an open circuit analysis).

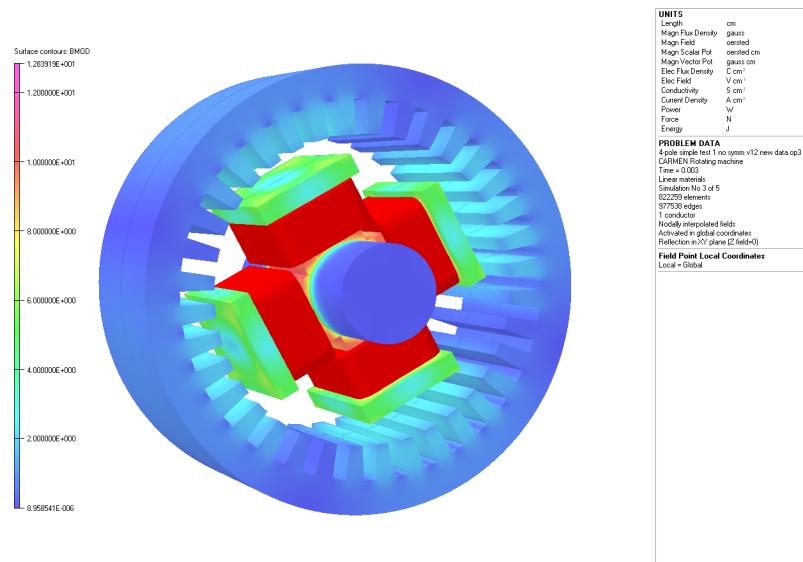
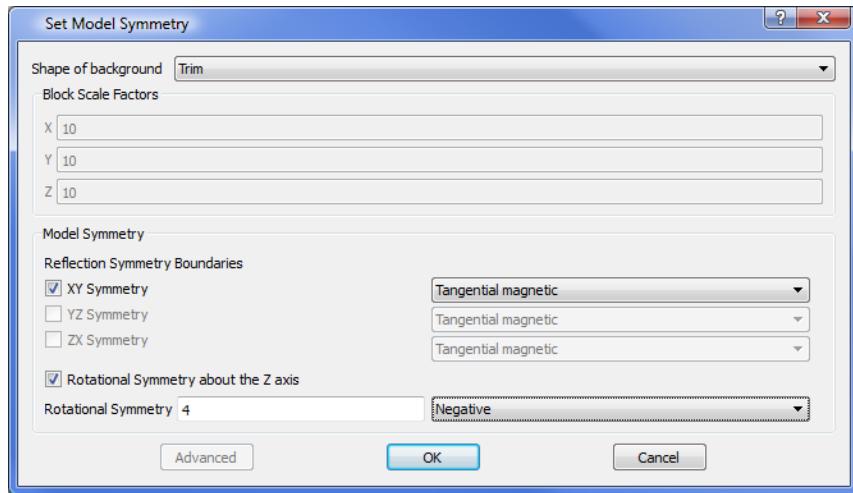


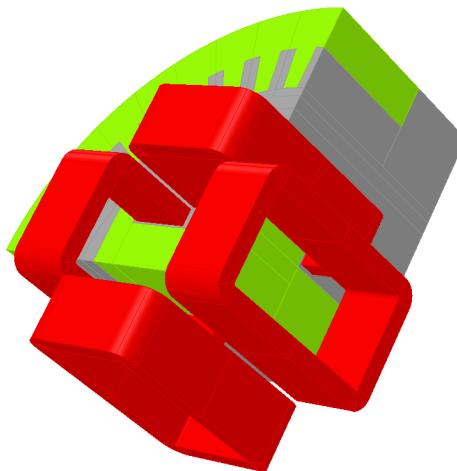
Figure 14.142 4-pole machine geometry

To exploit the symmetry in this machine, use  **Model symmetry** in the Modeller. This produces the dialog shown in [Figure 14.143](#), which has been completed to show that 90 degree symmetry exists and that the boundaries 90 degrees apart exhibit negative symmetry. Note that the only other type of symmetry that a transient EM with motion model can exploit is at the Z=0 plane.



*Figure 14.143 Dialog for setting periodicity*

When the model body is created, the section of the model in the positive X, Y and Z octant only is retained, as shown in [Figure 14.144](#).



*Figure 14.144 The Model Body*

Negative periodicity boundary conditions are automatically imposed on the boundary surfaces of the model that are parallel to the axis of the machine (Z). At its starting position (shown in [Figure 14.144](#)), these are the YZ and ZX planes, but as the machine starts to rotate the rotor boundary surfaces will not lie along one of the coordinate system planes and boundary surfaces in the azimuthal-axial will also be created. At each time-step the solver re-imposes the periodic boundary condition for the new position.

If circuits are being used, it is important that the correct symmetry is also set for the circuit to allow for parts of the winding outside the model space.

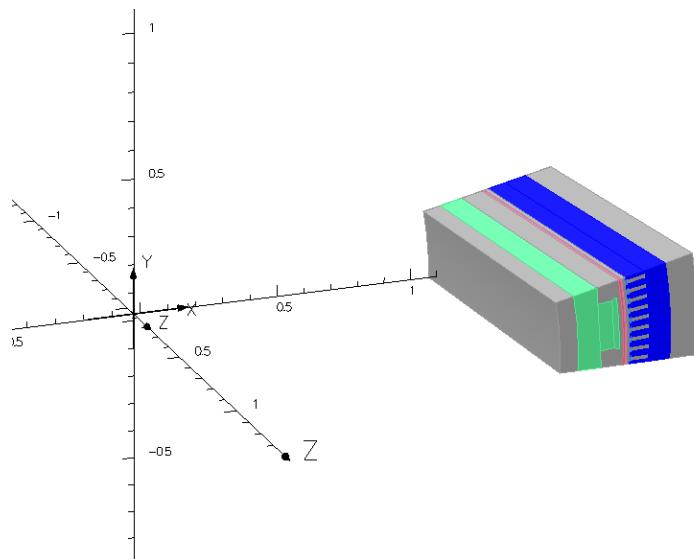
## Opera-2d data

The model of the 4-pole machine covered the full 360 degrees before periodic boundary conditions were applied and all the air space around the model also existed. The **Trim** option was used in the dialog to say that no additional background was required, but that all geometry outside the period should be removed.

Opera-2d models created for the rotating machines (RM) analysis often only contain the minimum period anyway. When these are imported as a starting plane for creation of the 3d model using



**Insert Opera-2d file**, they must be re-oriented so that one of the periodicity boundary surfaces lies on the ZX plane in positive X and the model extends from the ZX plane in a positive azimuthal sense i.e. counter clockwise around the Z-axis. The air regions should be added (possibly from the Opera-2d model) and the group labels set before creating the model body. In this case, the **Trim reflections** option should be chosen (rather than **Trim**). [Figure 14.145](#) shows a correctly oriented model for 1 pole of a superconducting hydrogenerator.



**Figure 14.145 Transient EM with Motion** model imported from Opera-2d

## Post-processing

When the model has been solved, results from any of the output cases can be viewed in the Post-Processor. The default selection for **Model symmetry** is **Database defined**. This means that the Post-Processor will know that the complete 360 degree symmetry (with appropriate reflections of the field) exists. However, the default setting for geometry display selection is **none**. Hence it is possible to only have the solved part of the model displayed but examine the results throughout space. Figure 14.146 shows a map of the radial component of flux density at the mid tooth radius.

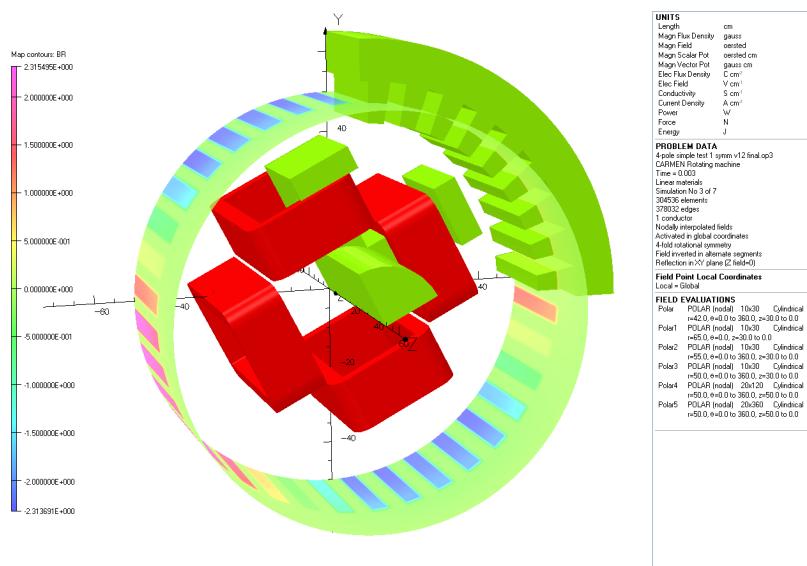


Figure 14.146 Fields at mid-tooth radius

Some or all of the repeated geometry can be displayed using the **Show model symmetry** option

in the **Select** dialog. For this machine, the options to show are

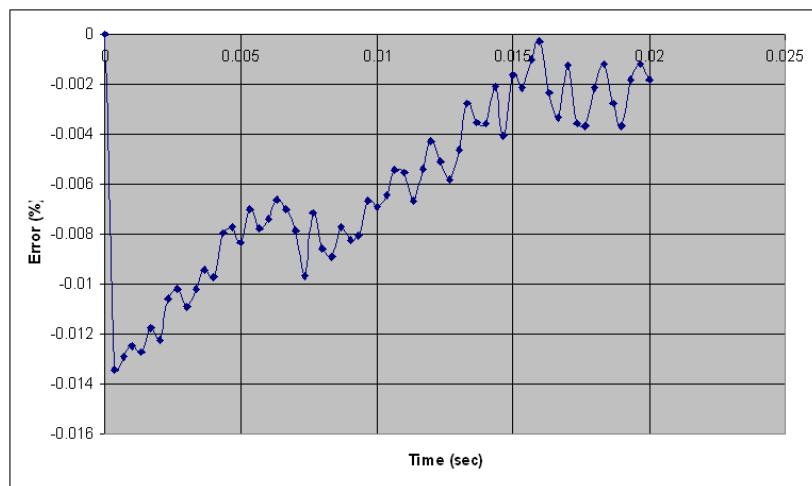
- **Quarter** (solved model reflected in XY plane),
- **Half** (Quarter model + a copy through 90 degrees),
- **Three-quarters** (Quarter model + 2 copies through 90 and 180 degrees),
- **Full** (360 degrees).

## Performance

The full 360 degree model has also been run for this machine. The symmetric model runs approximately 2 times faster than the full model. In fact, the equation solution time is about 3.5 times faster, but meshing the air gap takes slightly longer in the symmetric model as the periodic mesh structure has to be imposed. The periodic model only has 336000 equations - a small model for a **Motional EM** solution- so the advantages of the periodicity are not fully seen. Larger models with low numbers of poles dominated by the solution time rather than the remeshing will see a considerably greater

benefit. For models such as the hydrogenerator illustrated previously, which has 28 poles, only exploiting periodicity allows the user to obtain results efficiently.

The rotor coils in this model were connected to a circuit fed by a voltage step function. For the 360 degree model, only symmetry in the XY plane was exploited so the circuit symmetry was set to 2. In the symmetric model, the circuit symmetry is adjusted to 8, i.e. the part of the circuit coil inside the model space is only 1/8 of the total circuit (assuming all 4 rotor coils are in series). The predicted values of current in the circuit from the 360 degree and the symmetric models agree to within better than 0.015%, as shown in [Figure 14.147](#).



*Figure 14.147 Error in circuit current*

## Linear Actuator Example

The linear model presented in the following example simulates the dynamic behaviour of a tubular electromagnetic actuator. The specific steps needed for implementing a **Motional EM** model are illustrated, including the setting of motion control parameters and definition of a circuit.

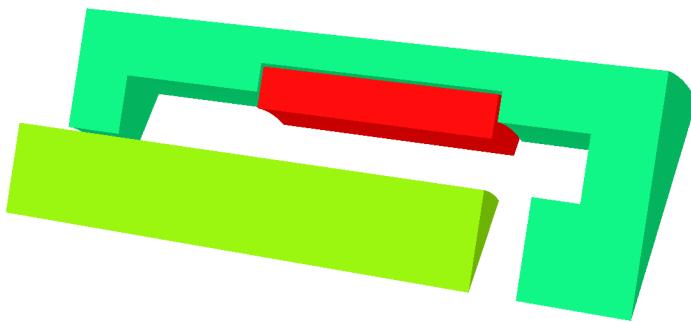


Figure 14.148 Linear actuator model for analysis with Motional EM

Due to the periodicity conditions included in the **Motional EM** models, only one eighth of the actuator needs to be modelled and analysed. The same group labelling techniques outlined in [Group Labels \[page 685\]](#) apply to the linear models as well. For this particular model, the cells are part of one of the groups: **STATIC**, **GAP** or **MOVING**.

The air region surrounding the actuator also needs to have a label assigned to it (**STATIC** in this case) and thus it will be created as a separate body rather than using the **BACKGROUND** option.

The air region inside the actuator is divided into three layers along the radial direction. The layer closest to the stator will be part of the group named **STATIC**, while the one closest to the plunger will be a part of the **MOVING** group. The remaining layer will make up the **GAP** group which will be reshaped and remeshed at each time-step according to the motion defined by the dynamic equations. [Figure 14.149](#) shows the three different layers of air between the stator and the plunger with the layer labelled **GAP** selected.

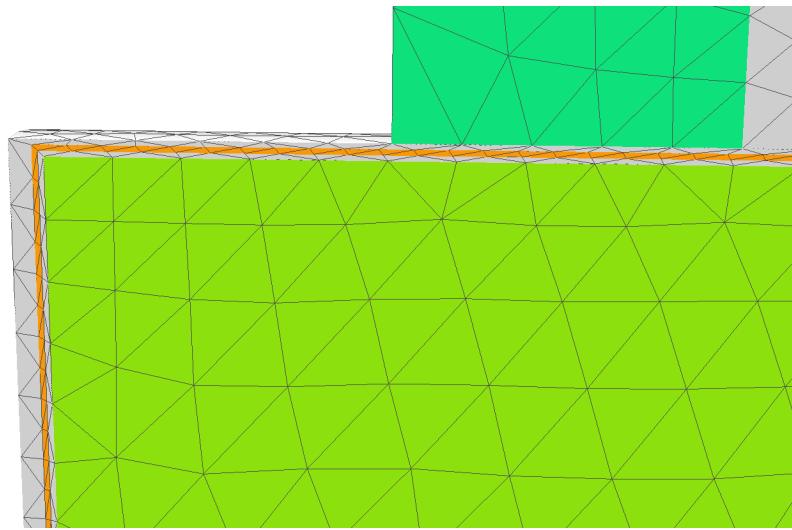


Figure 14.149 Air gap region in the linear model

## Coil setting

A model prepared for use with the **Motional EM** solver can include a Biot-Savart conductor or a conductor that is part of a circuit. In this example a circuit will be used; this allows a complete control of the current flow in the conductor based on the position of the plunger. The conductor will be included in the mesh (meshed conductor); a suitable mesh size and the group label **STATIC** need to be specified. The symmetry condition will be left to the default value of 1, but in the final model the symmetry condition imposed by the analysis options will take precedence and thus only one eighth of the coil will be meshed and analysed.

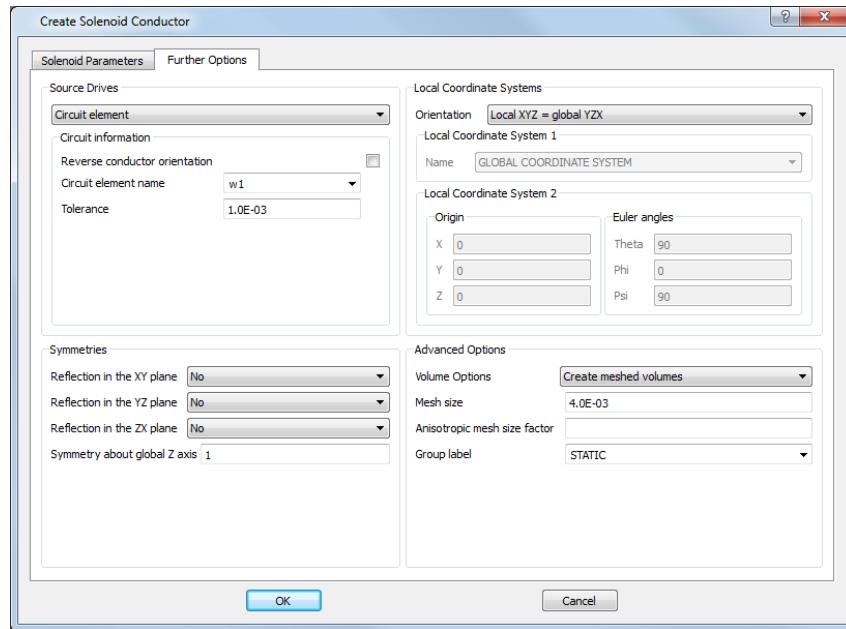


Figure 14.150 Meshed conductor setup

## Setting up the circuit

The circuit is set up using the Circuit Editor (see [Circuits in Harmonic and Transient EM simulations \[page 651\]](#)). The circuit is switched on at the first step of the simulation and it powers the coil until the plunger has reached its final position. The function that needs to be true for the switch to be closed is: `TTime>0 && #Mo_Shiftz<0.014`, where `TTime` is the simulation time and `#MO_Shiftz` is the variable that defines the shift of the moving group at a particular step.

The piece-wise linear diode `D1` is included in the circuit in order to provide a return path for the decaying current once the switch has been opened.

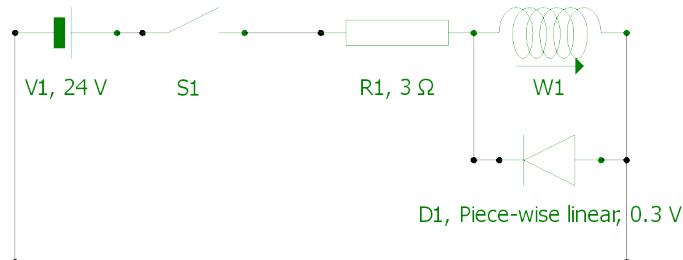


Figure 14.151 Electrical circuit for the linear actuator model

The coil `w1` is **Volume Meshed** which is needed for the integration of the coil in the final model body. The **Model symmetry** is used in order to keep the representation of the coil properties con-

sistent with the symmetry of the model. Since at **TTime=0** the switch **s1** is opened, the initial current in the coil is considered to be 0.

Winding: W1	
Property	Value
Name	W1
Meshing type	Volume Meshed
Number of representative filaments	2
Resistance per Unit Length	0
Turns	225
Use model symmetry	Yes
Symmetry Factor	16
Initial current	0
Unit	A
Notes	
Show label	Name and Value
Font	A [MS Shell Dlg 2, 8]
Colour	[0, 128, 0] (255)
Show Direction	Yes

Figure 14.152 Setting up the winding

## Movement

The movement of the plunger is defined in the generic motion control file **carmen\_control.comi**. The motion equation is expressed in terms of the acceleration in the Z direction, given by the proportion of the applied force and of the plunger's mass.

```
$STRING MO_MOTIONCONTROL ACCELERATION
$CONSTANT #MO_MASS 0.05
$CONSTANT #MO_ACCELX 0
$CONSTANT #MO_ACCELY 0
$CONSTANT #MO_ACCELZ MO_FORCEZ/#MO_MASS
```

The movement needs to be limited in order to account for the physical interaction between the plunger and the stator end. As explained in [Gap Regions \[page 685\]](#) there must be a region of air gap between two cells with different **GROUPLABELS**. Because of this, the plunger's movement limit needs to be set slightly before the stator end in order to allow for a meshable **GAP** between the two. The element size set for the **GAP** region needs to be considered when defining the movement limits.

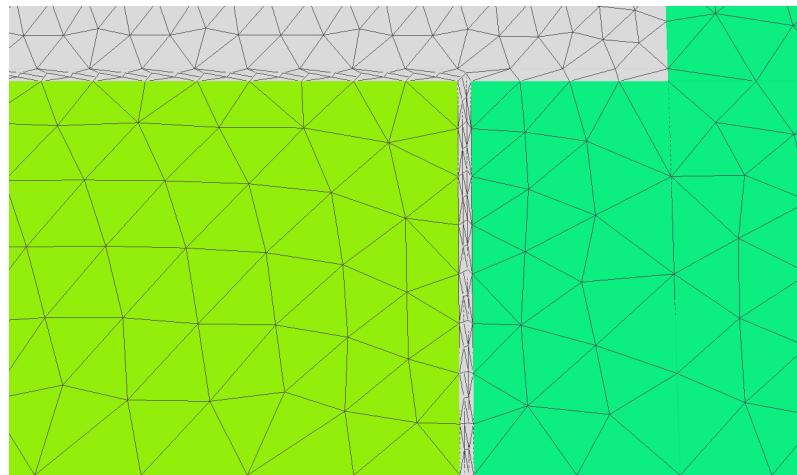


Figure 14.153 Air gap region between the plunger and the stator end for the position at the movement limit

## Analysis data

The specific analysis data options can be accessed through the Analysis Settings window. The solver will output the selected system and/or user variables at the times specified by the user. The type of movement is set to and linear motion.

## Results

After the model has been solved, the results of the analysis can be viewed in the Post-Processor at any of the output times (see [Post-processing \[page 697\]](#)). The logged variables will be output to a separate log file. In this case, the movement of the plunger and the effect that it has on the winding current have been plotted.

As expected, the variation of the current through the conductor is linked to the position of the plunger. The free-wheeling diode will allow the current through the winding to decrease when the switch is opened. A more accurate representation of these effects can be obtained by changing the time-stepping value from simple to adaptive and by reducing the time-step.

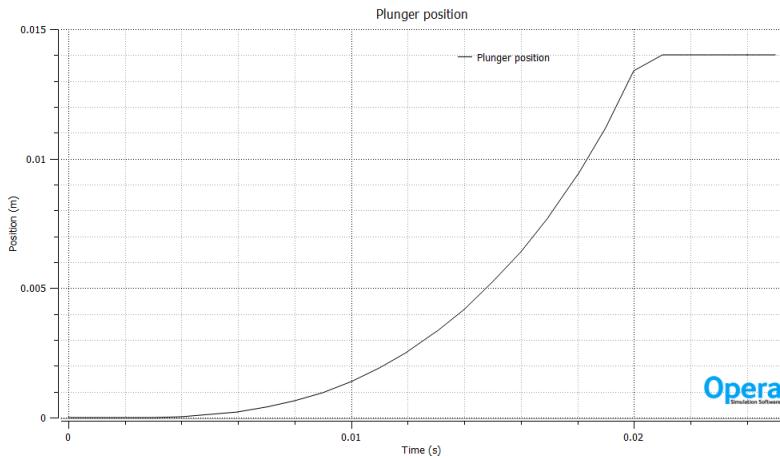


Figure 14.154 Plunger position vs. time

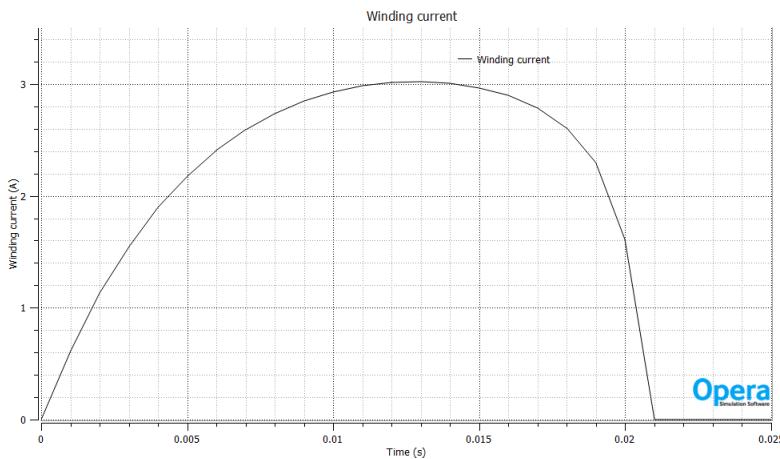


Figure 14.155 Winding current vs. time

## Example Control File

Transient Motion simulations are controlled using a **comi** file that describes the movement of the different sections defined in the model. Other simulation parameters can also be managed by the control file, such as time-stepping and external circuit properties (current/voltage functions, variable resistances, switch states, etc.). Both the Linear and the Rotational motion types make use of the control comi file in a similar way. However, control files can be used with any transient analysis.

The main use of the control comi file for a **Transient Motion** simulation is to set the movement type for the different groups.

Four different types of movement can be defined: stationary, position, speed and acceleration allowing fixed and variable speed components and mechanical coupling. The types of movement can be specified for each labelled group within the model. In a rotating machine, group labels are used to specify the rotor, stator and air gap. Similar groups must be defined for the moving and stationary parts of a linear machine. For further information about the use of Group labels in Opera-3d please see the **CELldata** command in the 3d Reference Manual.

The operation of the external circuit can also be controlled from the control file, by means of the state of functional components. Any variable defined in the model can be altered during the solving process. The calculated circuit parameters (currents, voltages and rates of change of current for each component) as well as mechanical parameters (angle, speed, acceleration) are available at every time step and can be used to define the drive functions in the control file.

With the help of the control file, the time-step method can also be changed during a transient analysis (between fixed and adaptive) as well as the size of the time-step. For further information on the time-stepping methods in Transient simulations please see Time-Stepping in the **Opera-3d Reference Manual**.

### Example 1

Example 1 presents a switching strategy implemented for a Switched Reluctance Motor (SRM) which relies on the rotor position for selecting the phase that needs to be switched. This model is driven using the mechanically coupled motion equations; the rotor position is not known in advance as a function of time (as it is known in the case of constant and variable speed analyses) so the calculated rotor position (**xx\_THETAZ** - where **xx** is the first two letters of the group label) must be used to determine the drive functions.

First, the rotor position is transformed into mechanical degrees and its value is reduced in the interval [0,...,#switch\_period] which represents one period of the switching function. The angular position of the rotor, in degrees (#positionD), is calculated with a three decimals precision.

```
$constant #temp mod(RO_THETAZ*180/PI,#switch_period)
$constant #positionD abs(NINT(#temp*1000)/1000)
```

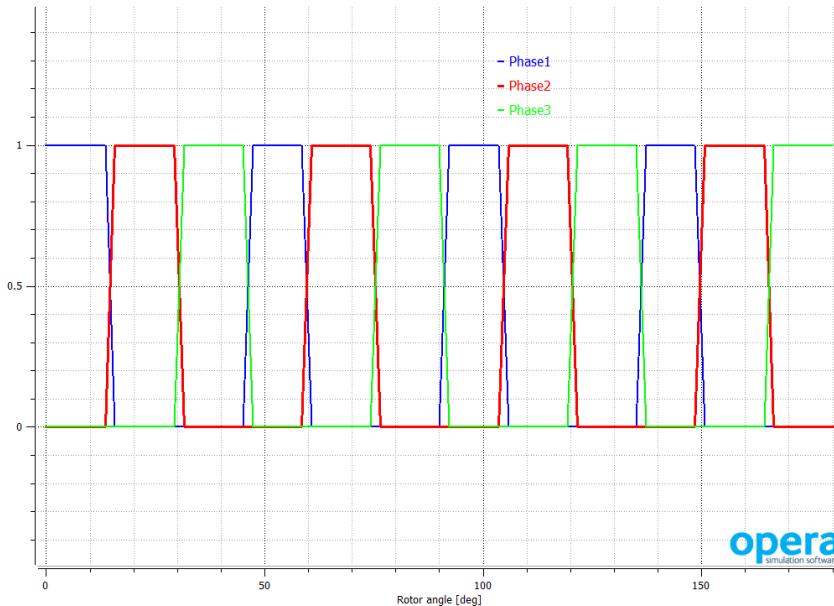
For the case of a three phase SRM, the switching pattern, as shown in [Figure 14.156](#), requires phase 1 to be ON between 0 and 15 deg, phase 2 between 15 and 30 deg and phase 3 between 30 and 45 deg. This is equivalent to a 45 degrees switch period.

```
$constant #Phase1_ON 0
$constant #Phase1_OFF 15
$constant #Phase2_ON 15
$constant #Phase2_OFF 30
$constant #Phase3_ON 30
$constant #Phase3_OFF 45
```

The external electrical circuit used to drive the SRM is presented in [Figure 14.157](#). The state of the switches that connect each of the phases to the voltage supply is calculated using the `RANGE()` function. Based on the rotor position, one of the three phases will be switched ON while the other two will be OFF. The state of each switch is defined by a variable (`#sn`) which is set in the control file. The value of these variables can be either 0, meaning that the switch is open, or 1 when the switch is closed.

```
$do #index 1 3 1
    $constant #switchState RANGE(#positionD;#PHASE%NINT(#index)_ON;#PHASE%NINT(#index)_OFF)
    $constant #S%NINT((#index-1)*2+1) #switchState
    $constant #S%NINT((#index-1)*2+2) #switchState
$end do
```

N.B. There must be no line break in control commands such as the first `$constant` command in the above script.



*Figure 14.156 Switching pattern for three phase SRM*

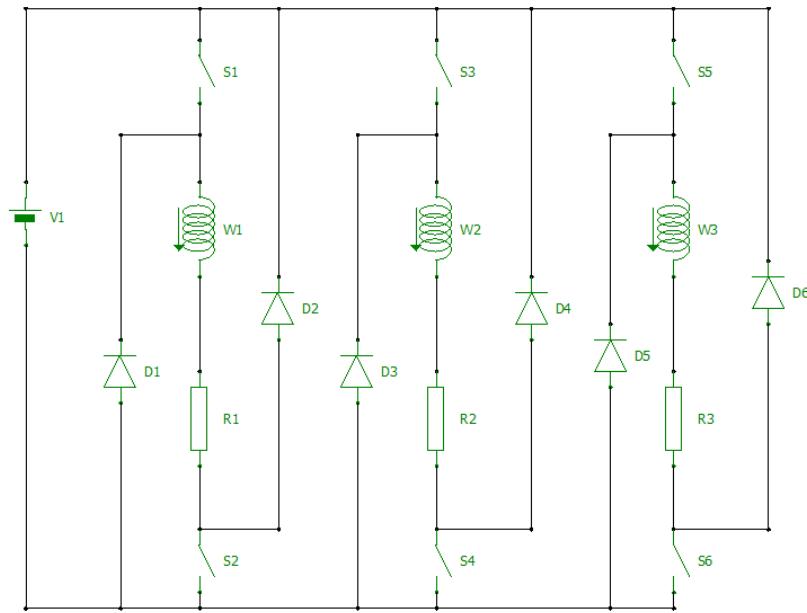


Figure 14.157 Full bridge asymmetric circuit for three-phase SRM

## Example 2

A hysteretic current following technique is implemented in this example, which uses a sinusoidal wave as the reference current input. The phase winding is fed from a DC bus and 4 switches provide the dual polarity needed (Figure 14.158).

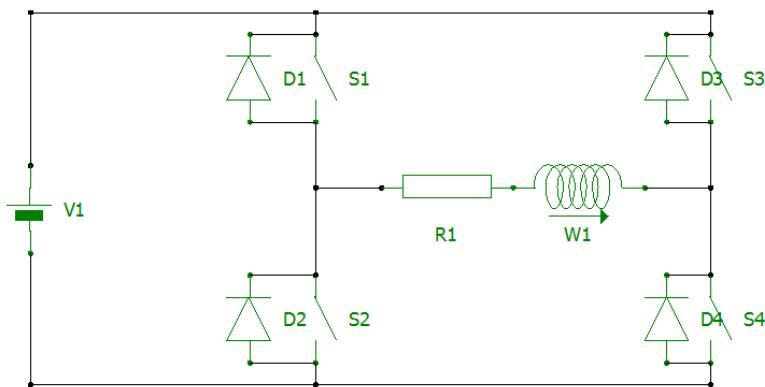


Figure 14.158 External circuit for sinusoidal hysteretic following

The reference sinusoidal current is defined in the control comi and the tolerance for the hysteretic band is set at 5% of the peak value.

```
// Reference sinusoidal current
$constant #I_peak 20
$constant #I_ref #I_peak*cos(2*PI*50*TTime)

// Hysteresis tolerance [%]
$constant #tol 5 Description='Error [%]'
```

The value of the actual current is compared with the reference at the end of each time-step and the state of the switches is updated accordingly.

```
$if !%compare(&Timestep_Stage&,END)
$if R1_I>#I_ref+#I_peak*#tol/100
$constant #S1 0
$constant #S3 1
$elseif R1_I>#I_ref
$constant #S1 0
$constant #S3 0
$else
$constant #S1 1
$constant #S3 0
$end if

$constant #S2 !#S1
$constant #S4 !#S3

$end if
```

The time-step method is set to fixed and the time-step value to **4E-4**. For a frequency of 50 Hz this translates into 50 steps / cycle. The resulting current wave is presented in [Figure 14.159](#).

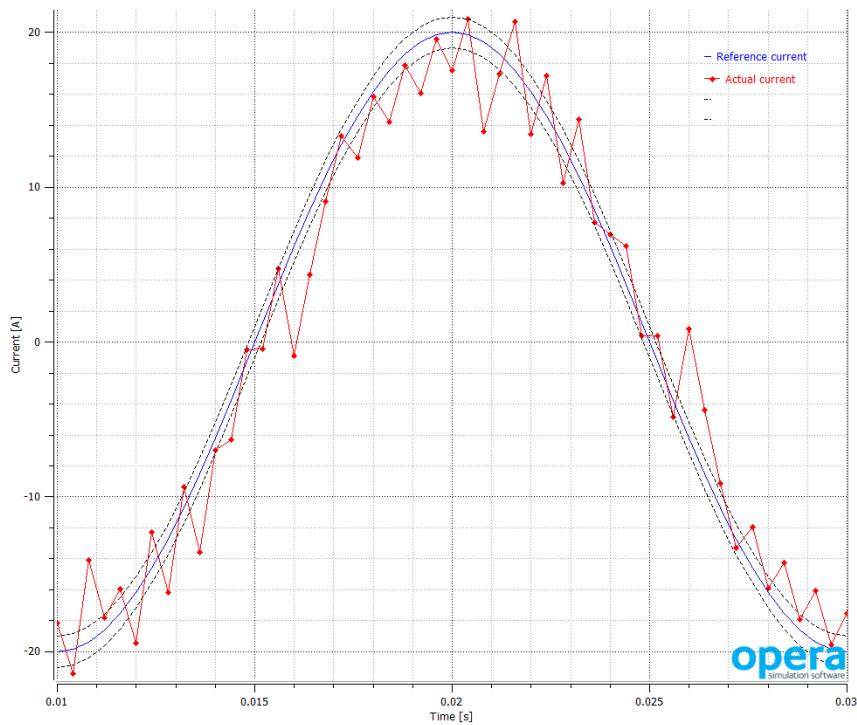


Figure 14.159 Hysteretic current control- *fixed* time-step

As can be seen, the time-step is not fine enough to correctly maintain the current within the hysteresis band. Using the same control file, the model is solved with an adaptive time-step, with an initial time-step of **4E-4** and a maximum error for the adaptive updates of 1%. Figure 14.160 shows that the current is maintained within the error limits much more effectively; however, the time needed to solve the same model using adaptive time-stepping is more than 6 times that for the fixed time-step method.

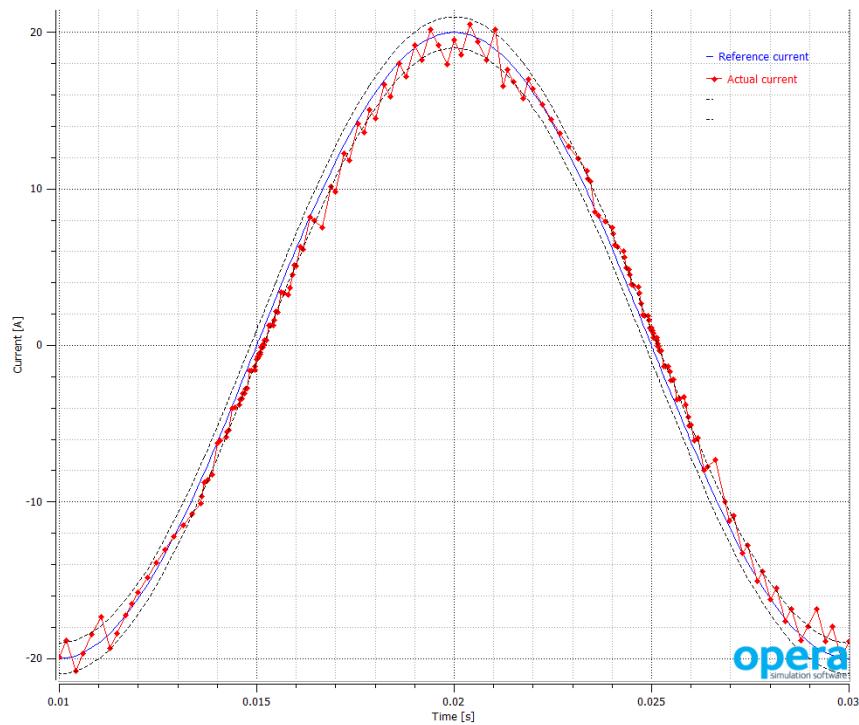
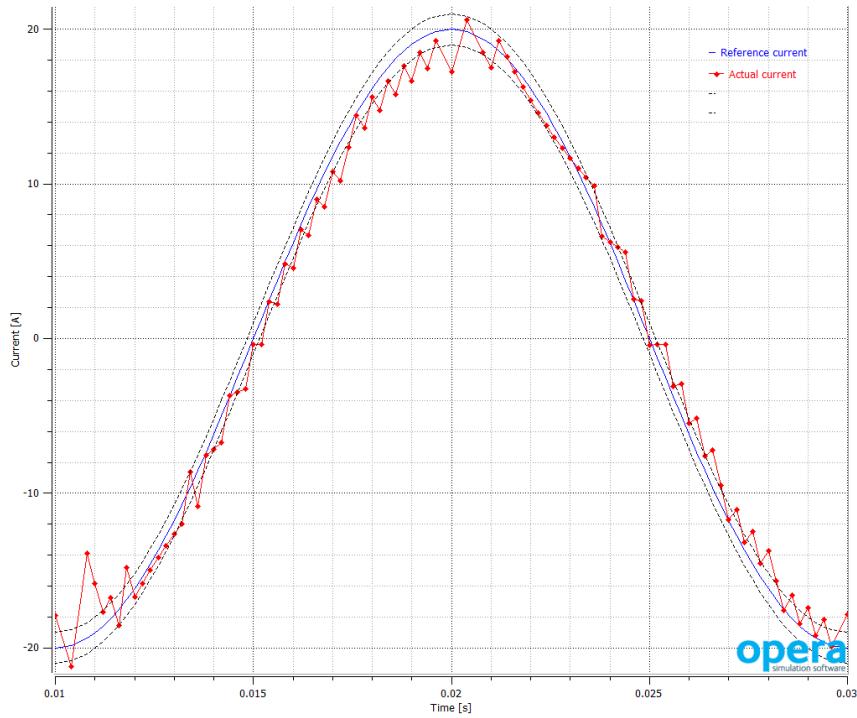


Figure 14.160 Hysteretic current control - *adaptive* time-step

Using the fixed time-stepping method with an improved control strategy that halves the time-step size when the current slope is higher than an imposed limit, a similar accuracy level can be achieved as in the adaptive time-step case (see Figure 14.161), while increasing the solution time by only 15%, when compared to the fixed time-step case.



*Figure 14.161 Hysteretic current control - fixed time-step with improved control strategy*

The step size is adjusted at the beginning of the fixed time-step based on the time derivative of the reference current:

```
$if !%compare (&Timestep_Stage, START)
$if abs(#crt_deriv)>1000
$constant #delta #step/2
$else
$constant #delta #step
$end if
$string Timestep_DeltaT OVERRIDE
$constant #Timestep_DeltaT #delta
$end if
```

The current derivative is updated at the end of each time-step along with the state of the switches.

```
$if !%compare (&Timestep_Stage, END)
$constant #crt_deriv (#I_ref-#prev_I)/#delta
$constant #prev_I #I_ref
$if R1_I>#I_ref+#I_peak*#tol/100
$constant #S1 0
$constant #S3 1
$elseif R1_I>#I_ref-#I_peak*#tol/100
$constant #S1 0
$constant #S3 0
$else
$constant #S1 1
$constant #S3 0
$end if
```

```
$constant #S2 !#S1  
$constant #S4 !#S3  
  
$end if
```

## Discussions

In order for the control file to be executed correctly when running using an adaptive time-stepping method, it is important that any changes to the state of the switches be done at the **END** of the time-step and changes to the time-step value or method be done at the **START**. For further information on the time-stepping control, please see the Time-Stepping section in the ***Opera-3d Reference Manual***.

The time-step size adjustment during the simulation can provide a much more efficient solution than the adaptive time-stepping for some applications, by refining the time-step only in regions of interest. Circuit and system variables, as well as mechanical information related to the rotor position can all be used to define the limits of these regions and the drive strategy.

## Opera in Simulink® System Analysis

Opera can be used as a block in a Simulink® system analysis. This section assumes that users are familiar with both Opera and Simulink and describes the interface between them.

Opera provides a Windows DLL (the Simulink block is not available on Linux systems) that exports a set of functions which can be called to create and control an Opera Application object. The Opera Application object has the ability to start and communicate with the Opera solvers, allowing external program control of analyses through this interface. A description of these functions is given in the ***Opera Manager User Guide***.

The external library is used in a Simulink block which allows a user to include Opera models in Simulink models (see Figure 14.162).

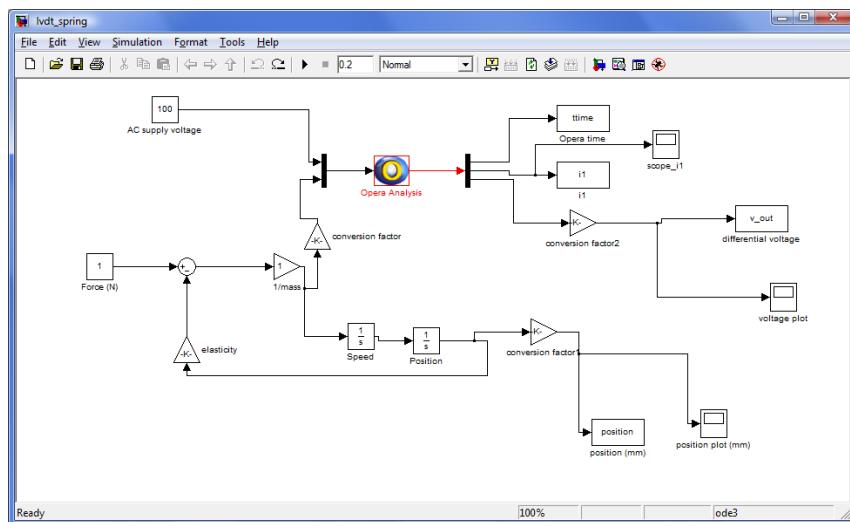


Figure 14.162 An Opera Block in a Simulink Model

This block is used like any other Simulink block and it is stored in a Simulink library called **Opera Analysis**. When this block is used, all the Opera simulation features are available, such as log files, output times and other options used by the solver.

In order to utilize the Simulink block, the Matlab® path must contain both the **bin** and **simulink** folders of the Opera installation, **%VFDIR%\bin** and **%VFDIR%\simulink**, where **%VFDIR%** is the path to your Opera installation folder.

## Opera Analysis Block

### What is it?

The Opera Analysis block is a masked Level-2 S-Function block. It calls functions which contain all the necessary calls to the Opera Application object, for example, to

- define simulation parameters,
- start the simulation,
- read some of the system variables at each Opera time-step and output them to the Simulink model,
- read some values from the Simulink model at each Simulink time-step and send them to the Opera solver,
- stop the simulation.

Each Opera Analysis block creates its own Opera Application object, so there can be more than one Opera Analysis block in a Simulink model.

### How does it work?

The Opera and Simulink solvers are independent from each other and only exchange their results. The Opera Analysis block starts the Opera simulation first with input values at time zero given by initialization commands. Opera runs for one time-step. At the end of each Opera time-step the following happens.

1. The output values are sent to the Opera Analysis block outputs.
2. Simulink runs to the same time as Opera. If the time-step in Simulink is smaller than the time-step in Opera, Simulink will run enough steps to get to the Opera time.
3. The user variables in the Opera model are updated from the Opera Analysis block inputs.
4. The Opera Analysis block runs Opera for the next time-step.

### What types of simulation are available?

All Opera transient solvers can be used. The time-step can be either **Fixed** or **Adaptive**.

The value of the fixed time-step can be chosen to be the same as the Simulink time-step, but this is not necessary: it is possible to choose a smaller time-step in Simulink by using a Rate Transition block, for example.

Adaptive time-stepping requires a variable time-step to be set in Simulink's options. It is recommended to limit the maximum time-step in Opera to a reasonable value, as any change in the Simulink model is taken into account only in the next Opera time-step.

## How are Simulink and Opera inputs and outputs connected?

The Opera Application object interface is similar to and replaces the use of *comi* scripts.

- Opera input variables are user variables with names beginning with #, e.g.
  - `#xx_ALPHAZ`, `#xx_OMEGAZ` for coupled motion or
  - `#v` for functional drives.
- Opera output variables can be:
  - system variables describing motion, e.g. `xx_TORQUEZ`, `xx_ALPHAZ`, etc.,
  - system variables for measuring the current in circuit components, e.g. `W1_I`, or
  - user variables.

## What are the other options available in Opera Analysis block?

The options are set by the **Function Block Parameters** dialog, shown in Figure 14.163.

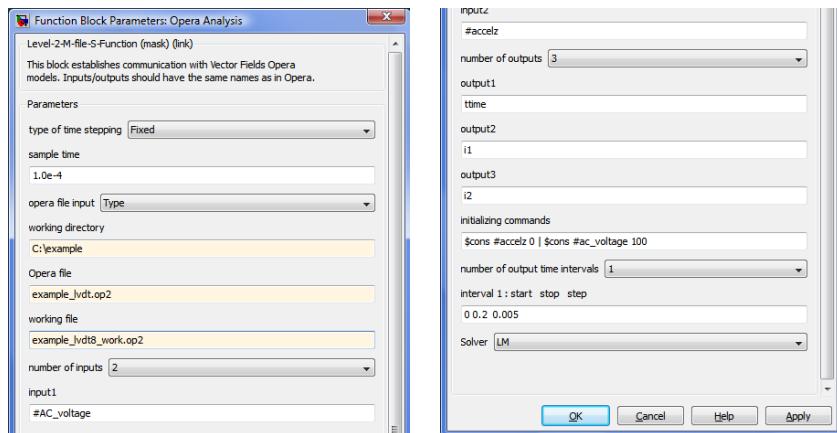


Figure 14.163 The Function Block Parameters Dialog

Here is the list of all the parameters:

Parameter	Meaning and Options
<b>type of time-stepping</b>	<b>Fixed or Adaptive</b>
<b>sample time</b>	The fixed time-step or maximum adaptive time-step.
<b>opera file input</b>	3 options: <b>Type</b> file and folder names. <b>Browse</b> to select file and folder names. <b>Lock</b> file and folder names so that they cannot be changed.

Parameter	Meaning and Options
<b>working directory</b>	The folder where the Opera file is saved.
<b>Opera file</b>	The filename of Opera model, including extension.
<b>working file</b>	The filename for a working copy of the Opera file which will be used for the simulation.
<b>number of inputs</b>	Any number of inputs.
<b>input 1</b> <b>input 2</b> <b>input 3</b> <b>input 4</b> <b>input 5</b>	The names of user variables which are set by the Opera Analysis block and used in Opera model.
<b>input 6...</b>	If there are more than 5 inputs, the names of the 6 <sup>th</sup> and subsequent inputs should be entered as a space separated list.
<b>number of outputs</b>	Any number of outputs.
<b>output 1</b> <b>output 2</b> <b>output 3</b> <b>output 4</b> <b>output 5</b>	The names of variables which are in the Opera model for use in the Opera Analysis block.
<b>output 6...</b>	If there are more than 5 outputs, the names of the 6 <sup>th</sup> and subsequent outputs should be entered as a space separated list.
<b>initializing commands</b>	Use standard Opera commands (c.f. a <i>comi</i> file). Put multiple commands on one line, separated by   (vertical bar). These commands are executed only at time zero.
<b>number of output time intervals</b>	Each time interval has a different output time-step. There can be up to 3 output time intervals.
<b>interval n: start stop step</b>	3 values specify the start time, the end time and a step time for each interval. These values are used to define the output times in the Opera model.
<b>Solver</b>	op3solve

# Estimating Iron Loss in Opera-3d

## Introduction

This Application Note reviews:

- Typical methods of quantifying the iron loss in a particular device when operating at steady-state:
  - User Defined Function method (using Loss curves)
  - Steinmetz Method
    - 1. Simple two term formulation;
    - 2. Quadratic Hysteresis Exponent formulation; and
    - 3. Bertotti-Steinmetz formulation
- Specific Pre-Processing options required in order to evaluate iron losses; and
- Post-Processing options to analyse the results.

All of the methods outlined herein are implemented for synchronous rotating machines within an example post processing script which can be found through the Opera Manager using:

**File -> Open examples folder-> 3D\Synchronous-machine\_IRON-LOSS-Calculations.comi.**

## Background

Iron loss calculation is of great importance in the computation of performance in electrical machines, actuators and other electromagnetic devices such as transformers and reactors. Accurate iron loss calculation becomes more important as drive frequencies increase as iron loss becomes the dominant form of loss, leading to significant reduction in machine efficiency.

## Harmonic Evaluation

The most basic implementations of the methods mentioned above assume a sinusoidal flux density variation at frequency  $f$  in the different parts of the machine. This is often too simplistic an approximation to make.

- In AC machines (e.g. synchronous, induction), higher order harmonics introduced by slotting can significantly contribute to the loss component and must therefore be accounted for.
- In stepping machines (e.g. brushless DC, switched reluctance) the problem is more fundamental. Square wave voltage excitation leads to non-sinusoidal current (and therefore flux density) distribution. In addition, different parts of the machine (stator and rotor teeth and yoke) undergo a different flux density variation cycle.

These problems are addressed by using following methods:

- The non-sinusoidal waveform problem is addressed by carrying out a full harmonic analysis on the waveform, to an order which is user specified. The number of steps (output times) is also

user-specified, and should be a function of the highest harmonic order chosen by the user, ensuring that the stored flux density variation with angle (time) contains enough information for correct harmonic amplitude extraction.

- The variation in the flux density waveforms in the different parts of a machine is addressed by obtaining flux density values not only in each machine section, but in each element in that section.

The result is a highly accurate evaluation of losses everywhere in the machine, and the ability to plot and compute losses as a function of frequency.

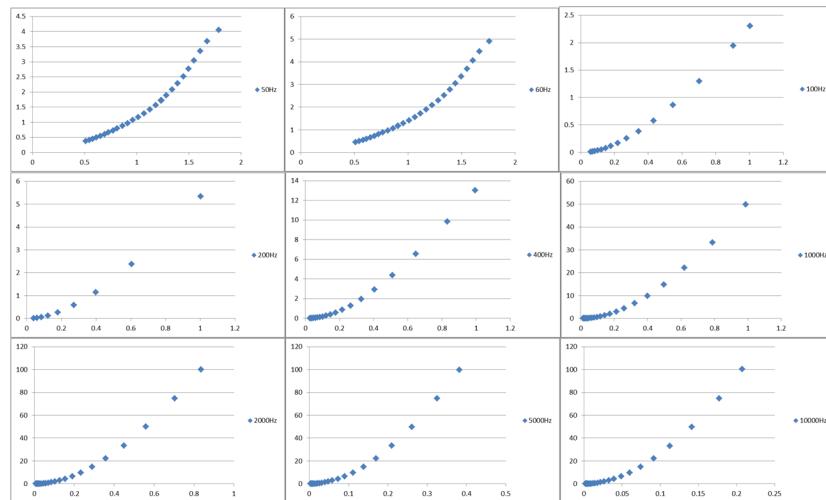
## Methods for Quantifying Iron Loss

### Method 1: Using User Defined Function

- Formulation

Manufacturers often supply loss data for materials as a series of frequency specific loss curves.

[Figure 14.164](#) shows an example of such data with frequencies ranging from 50 - 10000 Hz. Axes are B(T) and W (W/kg).



*Figure 14.164 Manufacturers loss curve data for magnetic steel at frequencies ranging from 50-10000 Hz*

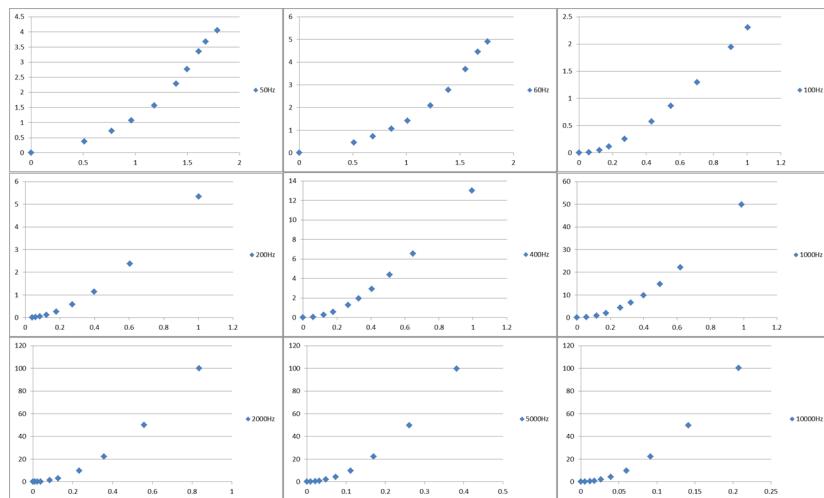
The standard method of fitting to this data would be through the Steinmetz formulation which includes terms which have a corresponding coefficient and varying dependency on the magnetic field and frequency ([See "Method 2: Using Steinmetz Formulae" on page 721.](#)). This method is useful in separating the contributions to the total loss, particularly in providing a good estimate of

the losses in soft magnetic material. In certain situations however, (e.g. when there is a wide range of excitation frequencies or highly non-linear materials), it has limited use as the losses generated within a material for a given field and frequency become too complex to be accurately represented with a function of the form used in the Steinmetz formulae.

The method outlined here makes use of the manufacturers' loss data explicitly and uses modified curves imported into Opera as a 2 dimensional function which covers both the magnetic field and frequency domains. Interpolation is performed using the nearest 8 data points supplied to the function. A downside of this method is that it doesn't split the loss into contributing components.

- Data Preparation

In order to create the function for Opera, each of the data series must contain the same number of data points but the values on either axis do not have to correspond. From the data shown in Figure 14.164 the 200 Hz data has only 9 data points. The other data is reduced to having 9 points by removing points not required to provide a reasonable description of the data. The crossing point at  $|B|=W=0$  is also added to each data set (as shown in Figure 14.165).



*Figure 14.165 Modified loss curve data for magnetic steel at frequencies ranging from 50-10000 Hz*

An alternative method would be to add extra points to the original data so that each series had the same number of points. However, it is clear from comparison of Figure 14.164 and Figure 14.165 that the removal of data points from the original data does not significantly impair the quality of the data.

Additionally it is necessary to add a zero frequency series (in order to allow the function to correctly calculate the DC field losses). This series must again include the same number of points and cover a similar range of field values whilst having zeros for the loss values.

The data is compiled into a table file (for more information on the format of and creating table files please refer to the section "Post-Processor Data Files" in chapter 7, "Opera-3d Post-Processor" of

the ***Opera-3d Reference Manual*** and to "Tabular Functions" on page 576). An example of this format is shown below.

6	5	1	2
1 F [1.0]			
2 B [TESLA]			
3 Wh [1.0]			
0			
0	0	0	
0	0.1	0	
0	0.2	0	
0	0.3	0	
0	0.4	0	
0	0.5	0	
400	0	0	
400	0.1	14.66	
400	0.2	49.05	
400	0.3	110.3	
400	0.4	215.5	
400	0.5	389.3	
1000	0	0	
1000	0.05	15.43	
1000	0.1	49.83	
1000	0.15	105.3	
1000	0.2	188.9	
1000	0.25	314.5	
5000	0	0	
5000	0.05	134.9	
5000	0.07	249.5	
5000	0.1	522.7	
5000	0.13	956.4	
5000	0.15	1376	
8000	0	0	

8000	0.01	18.47
8000	0.03	107.4
8000	0.05	260.5
8000	0.07	508.5
8000	0.1	1139

User defined table

This data can be imported into Opera-2d or Opera-3d as a user defined function which can be referenced in any functional data field (such as the volume integral and plot commands). The process for import is as follows:



User Defined Functions > Select table file > New

## Method 2: Using Steinmetz Formulae

- Formulation

The term iron loss refers mainly to two loss components, namely the hysteresis loss and eddy current loss. The well-known two term Steinmetz formulae separates the loss into components from different phenomena: hysteresis loss and eddy current loss. Hysteresis loss can be expressed as:

$$P_h = \frac{K_h B_m^n f}{\rho} \quad (14.47)$$

where

- $K_h$  is the hysteresis loss constant,
- $B_m$  is the maximum flux density,
- $f$  is the frequency,
- $\rho$  is the material density and
- $n$  depends on the material and is typically in the range 1.5 - 2.5.

Eddy current loss is expressed as

$$P_e = \frac{K_e B_m^n f^2}{\rho} \quad (14.48)$$

where  $K_e$  is the eddy current loss constant.

There have been many variations to the two term Steinmetz equation described in literature in attempts to improve the overall accuracy of losses calculated by this method. Here we present two such variations, both of which are implemented within the example comi file provided. The example is open to the user's preference of loss model and may be customized to suit if one of the four methods implemented does not provide a suitable representation of the iron losses in a particular system.

1. The Bertotti-Steinmetz equation supplements the two term Steinmetz equation with a third term often described as the anomalous or excess loss term. This additional term is expressed as:

$$P_x = \frac{K_x B_m^{3/2} f^{3/2}}{\rho} \quad (14.49)$$

where  $K_x$  is the excess loss constant.

2. A modified version of the two term Steinmetz equation replaces the constant exponent  $n$  from the hysteresis component term with a variable with quadratic dependency on the peak B field. This allows the two term Steinmetz equation to better represent highly non-linear materials.

As for Method 1, loss data in watt/kg for different lamination materials is provided by manufacturers for a range of frequencies. These curves are used to extrapolate values for  $K_h$  and  $K_e$  (and  $K_x$ ) and also define the exponent  $n$  in the hysteresis loss term for the modified two term Steinmetz formulation. It is important to note that it is the user's responsibility to use appropriate values of  $K_h$ ,  $K_e$ ,  $K_x$  and  $n$  (where applicable), for the materials in their model.

## Preparation/Pre-Processing

The **Transient Motion** solver must be used to model the electrical machine in transient (rotating) mode. At least one complete period (excitation cycle) of rotation at steady-state should be included.

### Output times

A series of output times in one excitation cycle of the machine must be recorded. It is important for the user to identify the angular span of this excitation cycle. In a permanent magnet machine this may be a straight forward calculation (e.g. 2 poles) while in a switched reluctance machine, this will depend on the number of teeth per phase and the excitation pattern.

The number of output points in time period chosen dictates how many harmonics can be calculated. For instance, if the user wants to evaluate  $n$  orders of harmonics then the model should have at least  $2n+1$  output times defined.

### Groups

The command file requires that appropriate region groups have already been defined in the model, namely:

- the stationary lossy parts should be grouped together in a group called **STATORIRON**; and
- the moving lossy parts should be grouped as **ROTORIRON**.

## Post-Processing

The example comi file carries out two main stages:

- the first stage performs the Fourier analysis of the flux density; and
- the second stage evaluates losses.

## Input to the script

The user specifies:

- Name of the Opera-3d database
- Simulation data:
  - Total number of poles (`#numpoles`)
  - Number of poles modelled (`#polesMod`)
  - Rotor speed in RPM (`#revm`)
  - Whether the model has axial symmetry (`#hasAxSym`).
- Analysis data:
  - Harmonic to start (`#harStart`)
  - Harmonic to stop (`#harStop`)  
In Opera-3d Post-Processor, depending on the size of the model, the post-processing can take quite a lot of time. In order to expedite this process, `#harStart` and `#harStop` have been implemented so that users can launch multiple post-processing jobs (in parallel) over different ranges of harmonic orders.
  - Case number corresponding to the start of the period (`#Case_start`)
  - Number of cases in one period (`#Ncases`)
  - Step in case number (`#Case_step`)  
The users can use `#Case_step` to specify whether the calculation needs to be performed on every consecutive output time or for instance every second output time.
  - Where to calculate losses (Stator or Rotor)
- Types of loss calculation:
  - Simple 2 term Steinmetz formulation;
  - Quadratic Hysteresis Exponent Steinmetz formulation;
  - Bertotti-Steinmetz formulation;
  - User defined function method.
- Each of the material coefficients for the implemented methods are stored separately to allow all of the methods to be calculated in parallel from a successful harmonic evaluation. A separate set of coefficients (signified by Stat/Rot) are available for rotor and stator to allow different materials to be used.
  - `#KhStat` and `#KhRot`: Hysteresis loss constant;
  - `#KeStat` and `#KeRot`: Eddy current loss constant;
  - `#npStat` and `#npRot`: The exponent  $n$  is calculated as

$$n = aB^2 + bB + c \quad (14.50)$$

where  $a$ ,  $b$  and  $c$  are fixed constants `#fac1`, `#fac2` and `#fac3` respectively.

## Process

The following procedure is implemented:

- A series of read/write table commands is used to record the values of flux density in each element over the excitation cycle. In this way, a flux density waveform over the excitation cycle is stored for each element.
- A harmonic analysis on the flux density waveform is performed, identifying the n-order harmonic content in the waveform.
- The iron loss per harmonic order is computed, and accumulated to the total iron loss.

## Output from the script

All the values calculated for losses (Stator, Rotor and Total) are stored in appropriate text files depending on the methods chosen. The text files are stored under a sub folder called **Results** followed by the range of the harmonic order (e.g. **Results0-4**). The values are also presented in a summary dialog ([Figure 14.166](#)).

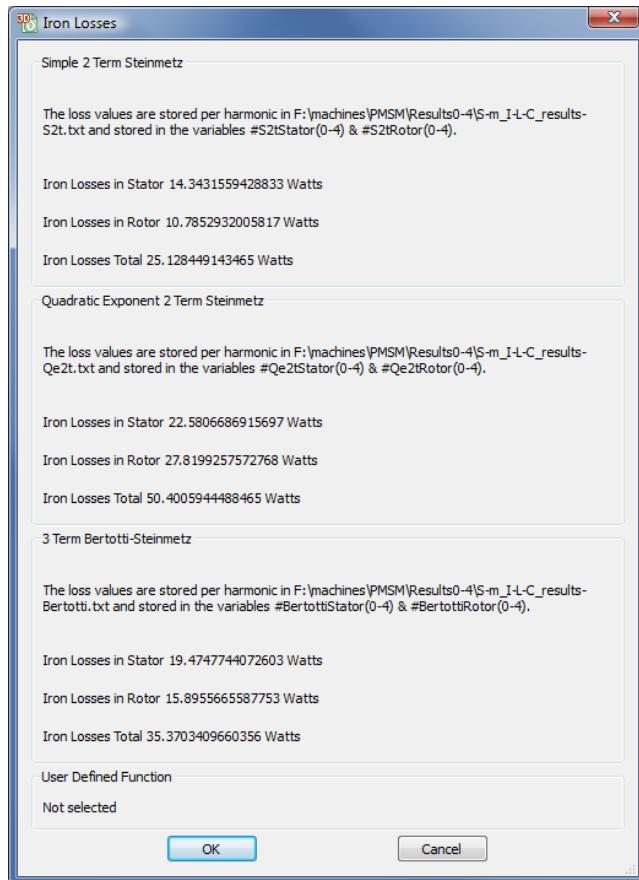


Figure 14.166 Summary of the results

## Modelling DC Current flow

---

### Introduction

Analysis of current flow in conducting materials is important in applications such as cathodic protection, non-destructive testing and metals production. As well as computing the distribution of current flowing in the conducting media, the Post-Processor can calculate the magnetic field distribution from the currents. This enables Lorentz force calculations on the conducting media to be performed.

The current flow simulation can also be used in conjunction with other Opera-3d simulations. The magnetic field distribution obtained from the current flow can be used as the source magnetic field in a magnetic field analyses. There is also the option to include lossy dielectric material in electrostatic and current flow simulations. These slightly conducting dielectrics are used primarily in high voltage power engineering applications to reduce electric stress levels. When the lossy dielectrics option is activated for electrostatic or current flow analysis, the simulation runs in two stages - initially solving the DC current flow in the lossy materials and then using the computed potential distribution (and other boundary conditions) for the electrostatic analysis. Note that the lossy dielectric option requires an additional license.

This application note shows some examples of all the features discussed above.

### DC Current Flow in a Conducting Medium

Figure 14.167 shows part of a DC current bus-bar arrangement in a metals production facility. As a part of the mechanical fixing, it is necessary to drill 3 holes in the bus-bar. The user wishes to determine how much the DC resistance increases as a result of the reduction in copper volume. The volume of the holes is filled with a second material whose properties can be varied - so that with the same model it is possible to examine both the undrilled and drilled bus-bar. In fact, this is good practice in all finite element calculations where the result required represents a small change between configurations. Changes in the mesh can sometimes produce changes in results that are of the same order as the differences being evaluated.

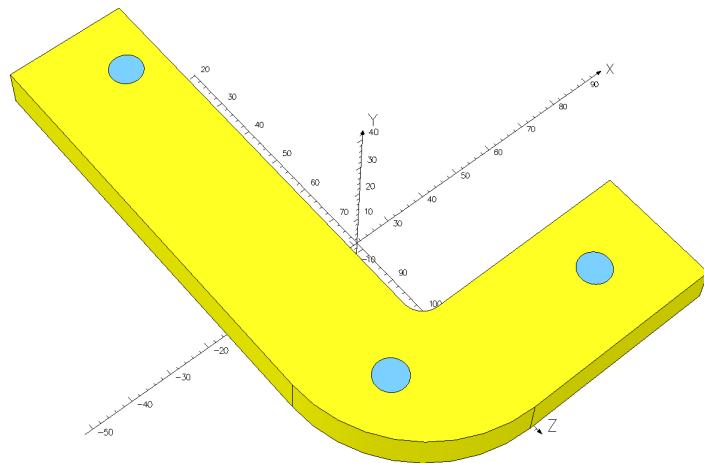


Figure 14.167 Geometry of bus-bar

The model only consists of the two media - the surrounding air is implied by the default (Neumann) boundary condition at the external surfaces. The model is excited by applying voltage boundary conditions at the two ends. As it is the resistance that is required, making the voltage difference of 1 volt between the two ends, means the inverse of the current flowing will be the resistance (from Ohm's law). [Figure 14.168](#) shows the two surfaces where the boundary conditions are applied.

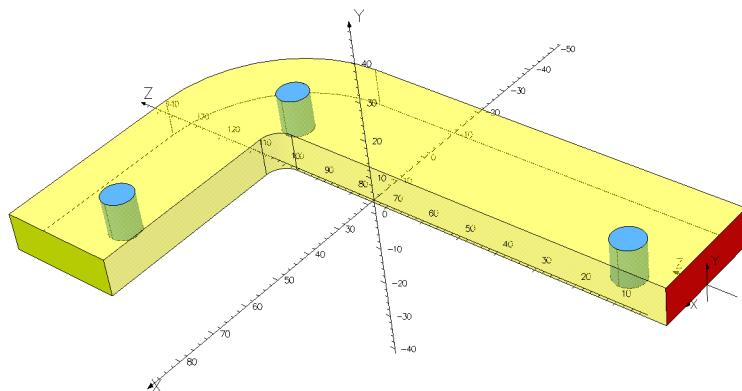


Figure 14.168 Voltage boundary condition surfaces

After meshing, defining the material properties (only conductivity is important) and boundary condition values, the database for a **Static Current Flow** analysis database (**.op3** file) can be created. The material properties in the hole can then be changed and a second case added to the

database by selecting  **Add simulation to existing database** option, as shown in Figure 14.169.

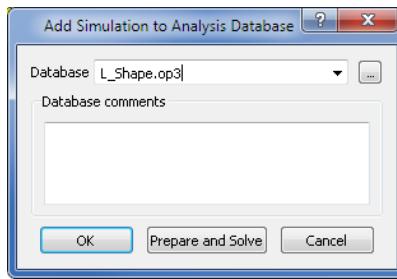


Figure 14.169 Dialog for additional case in database

## Current flow direction

After the analysis has completed, the Post-Processor can be used to examine the two solutions. Figure 14.170 and Figure 14.171 show the distribution of  $|J|$  in the two solutions, clearly showing how the holes interrupt the current flow.

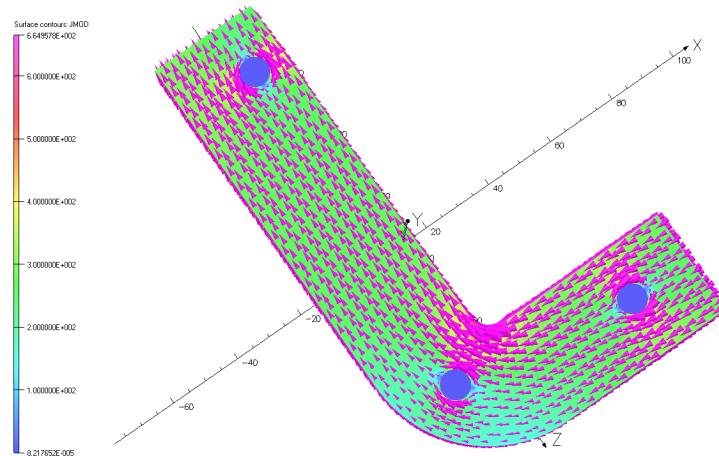


Figure 14.170 Current flow with holes in model

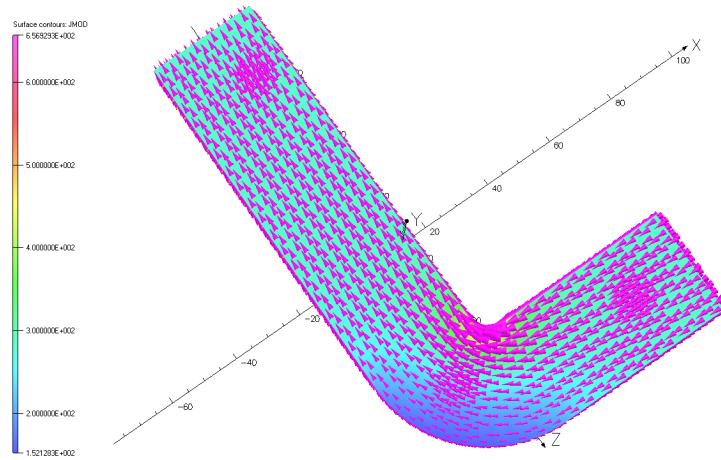


Figure 14.171 Current flow without holes in model

## Total current integral

To determine the current flowing in the models, the surface integral of the normal component of the current density on one voltage boundary condition surface can be calculated. Use the  **Select** dialog to select one of the boundary condition surfaces (see Figure 14.172) and then compute the surface integral of  $\int \mathbf{J} \cdot \mathbf{N}$  using the  **Other surface integrals** toolbutton.

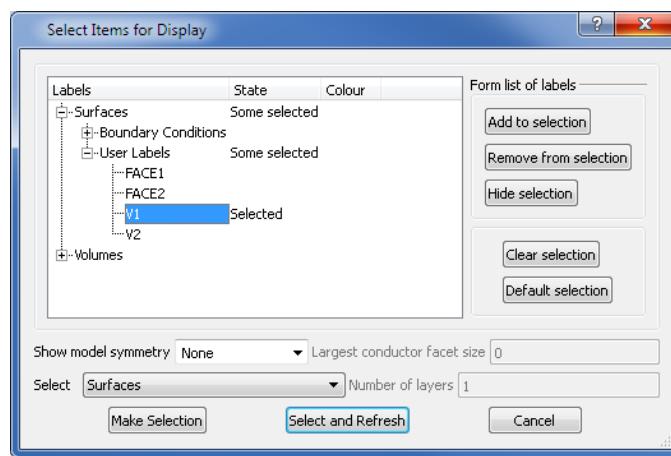
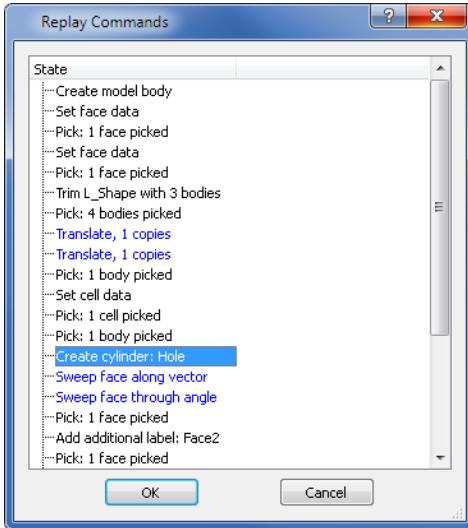


Figure 14.172 Dialog for selecting voltage surface

The result is stored in system variable **INTEGRAL**, which can then be inverted to give the resistance. The resistance in the two cases are 12.41 and 11.78 microhm with and without the holes respectively.

If this change was unacceptable, then the model could easily be parameterized using the **Re-build** facilities in the Modeller to adjust the hole radius until suitable values were obtained. [Figure 14.173](#) shows how the cylinder dialog would be re-accessed for this model.



*Figure 14.173 Dialog for selecting cylinder in re-build*

## Magnetic fields from current flow

Magnetic fields can be evaluated by performing Biot-Savart integrations over the volume of the geometry. In this model only part of the conductor has been modelled; in reality there must be conductors to deliver current at one end of the bus-bar and to take it away from the other. Obviously it is necessary to model sufficient of the conductor so that the omitted sections will not contribute significantly to the local field values. In the bus-bar example, the current entering and leaving the model is assumed to be uniform across the cross-section, so will not contribute to the local field on the bend.

 Switch the **Field Point Options > Field Calculation Method** to **Fields calculated by integration**. It is then possible to ask for the fields anywhere in space. For example, the effect of the holes on the local field can be seen on a line at 45 degrees across the corner, 1 mm above the surface of the bus-bar. [Figure 14.174](#) shows the position of the line and [Figure 14.175](#) shows BY from the two cases (black-solid with hole, red-dashed without).

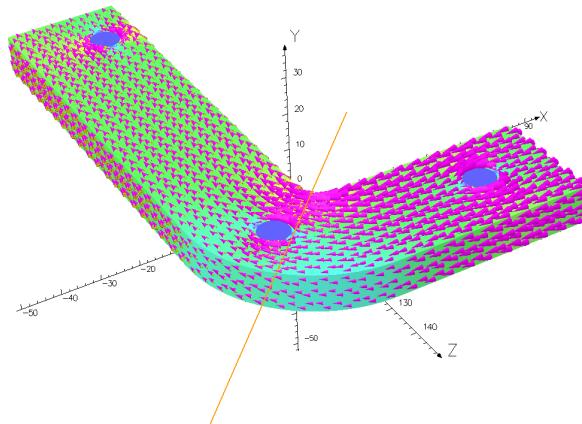


Figure 14.174 Position of line for evaluation of fields

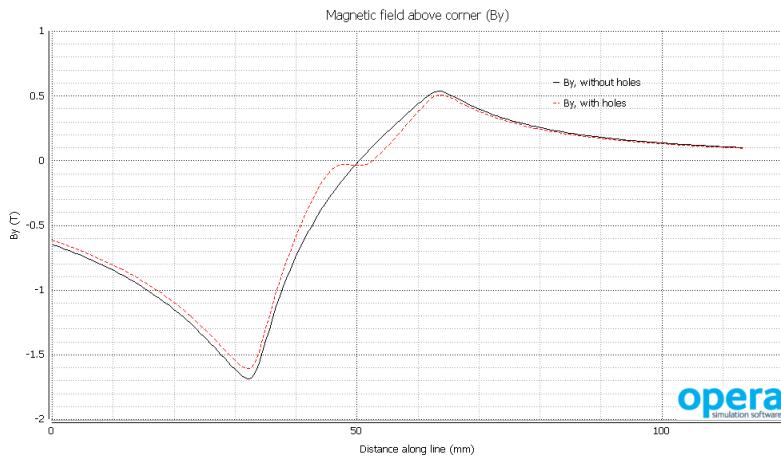


Figure 14.175 Fields at corner of bus-bar

## Evaluating forces

Magnetic fields can be evaluated using Biot-Savart integration, as shown in the previous section. However, current density is evaluated using nodal averaging. This presents a quandary when Lorentz force densities are to be evaluated, as they are the cross-product of these two quantities ( $\mathbf{F} = \mathbf{J} \times \mathbf{B}$ ). For current flow models, a third method of field evaluation is supported - **integration with nodal**. The Post-Processor will calculate magnetic field quantities using integration but will compute nodal values of current density. Note that this calculation can be very time consuming as the Biot-Savart integral expressions become nearly singular when the source ( $\mathbf{J}$ ) and evaluation points ( $\mathbf{B}$ ) are in the same element, and the program has to take special action to ensure accurate results. Figure 14.176 shows contours of the Z-component of Lorentz force density on the configuration with the holes.

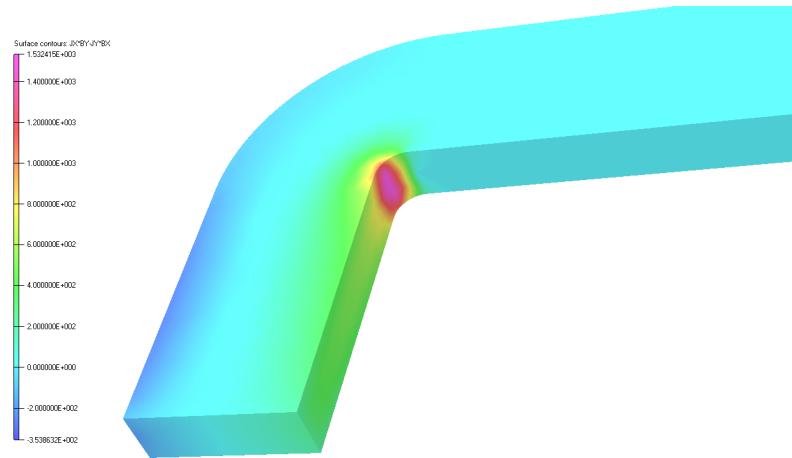


Figure 14.176 Force density contours

## Using Field Values from Current Flow Simulations

The magnetic fields computed in the current flow simulation can be used to "excite" other applications. The bus-bar is now assumed to emerge from some steel trunking just before the 90 degree bend, as shown in Figure 14.177.

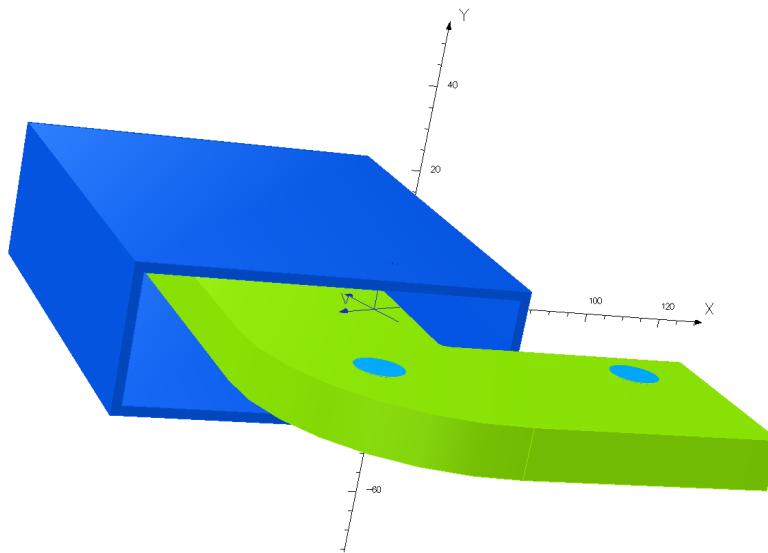
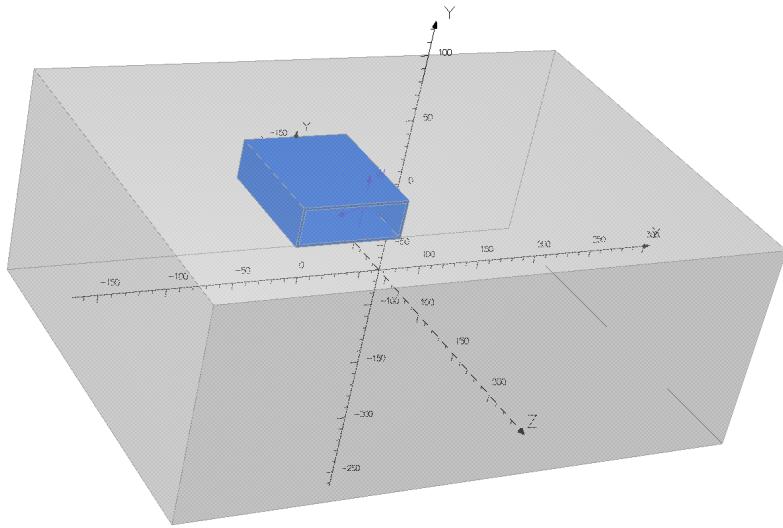


Figure 14.177 Bus-bar with steel trunking

To evaluate the flux density in the trunking, it is no longer necessary to include the geometry of the bus-bar in the model. So the Magnetostatic model only consists of the trunking and surrounding free space, as shown in [Figure 14.178](#).



*Figure 14.178 Geometry for magnetostatic model*

After meshing and creating the **op3** file database (use **Simple Conductor Line Integrals**) for the magnetostatic simulation, launch the Post-Processor and read in the unsolved database. The fields from the current flow simulation are added as RHS terms for the magnetostatic model using table files.

- Create a table file of the coordinates of the nodes in the magnetostatic model.
- Load in the solved current flow database. Set the field evaluation method to integration.
- Create a second table file using the first table file as the input coordinates. The second table file should contain 6 columns - **X**, **Y**, **Z**, **H<sub>X</sub>**, **H<sub>Y</sub>**, **H<sub>Z</sub>**. However, the H field components should be re-named as **RHS<sub>X</sub>**, **RHS<sub>Y</sub>**, **RHS<sub>Z</sub>** to denote they are **Real Source** field components.
- Re-load the magnetostatic model database.
- Import the table file of **RHS<sub>X</sub>**, **RHS<sub>Y</sub>**, **RHS<sub>Z</sub>** to the database.
- Close the Post-Processor and start the analysis.

[Figure 14.179](#) shows the resulting flux density in the trunking.

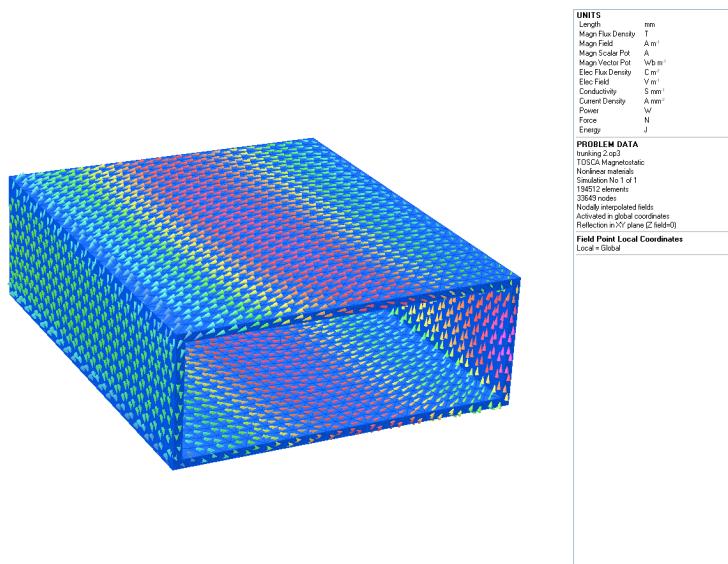


Figure 14.179 Magnetic flux density in trunking

## Lossy Dielectric Simulation

In many power generation and distribution applications, semi-conducting dielectric materials are used to reduce electrical stresses. Figure 14.180 shows a typical bushing arrangement with an HV conductor at 10 kV (red) emerging from a grounded tank lid (yellow) with a green insulating bushing between them. The permittivity of the bushing is 5.7.

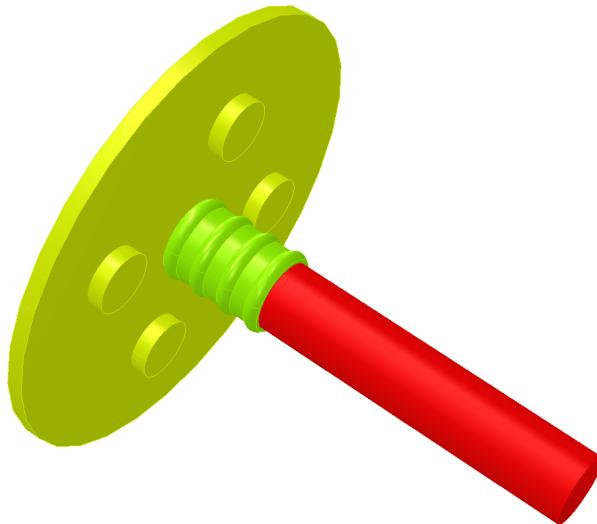
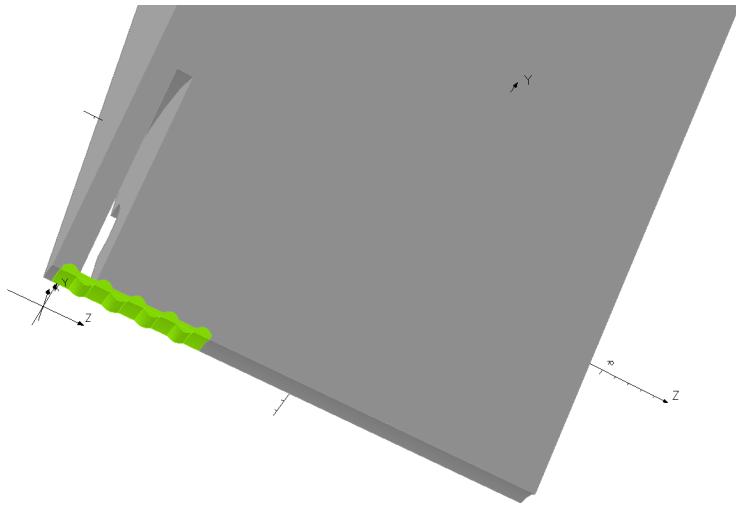


Figure 14.180 Geometry of bushing

Only the dielectric and the surrounding air need to be included in the model. The model has 8-fold symmetry so can be further simplified by including only 45° ([Figure 14.181](#)).

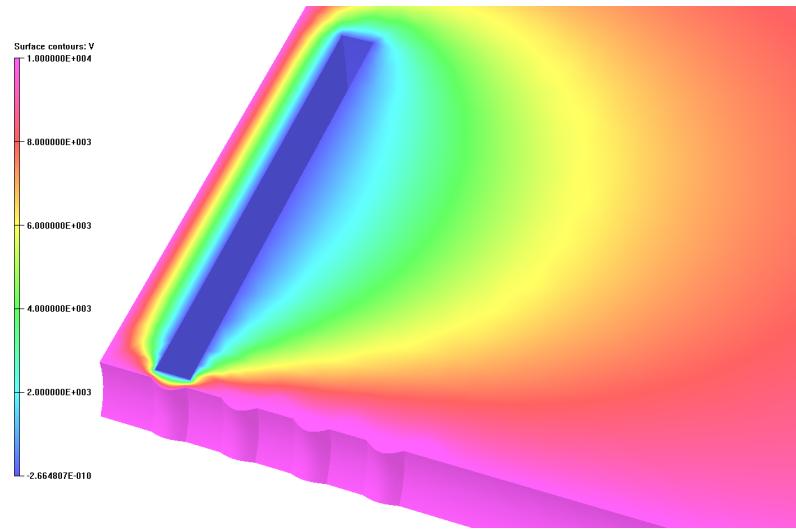


*Figure 14.181 3d model of bushing*

## Model without stress relief

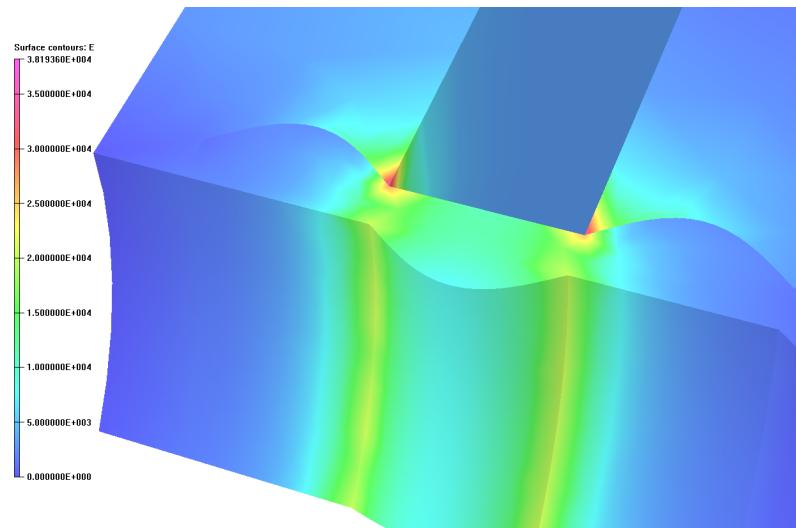
Symmetry allows a 45 degree segment of this model to be simulated and a background air region is

added using the  **Model symmetry** dialog to allow the electric field to be obtained, using the electrostatic analysis. [Figure 14.182](#) shows the geometry of the model that is solved with the potential solution, **V**, added.



*Figure 14.182 Potential on the model*

The electric field just in the air outside the insulator near the lid is close to  $40 \text{ kV cm}^{-1}$ , as can be seen in Figure 14.183. This can lead to breakdown as it exceeds  $33 \text{ kV cm}^{-1}$ .

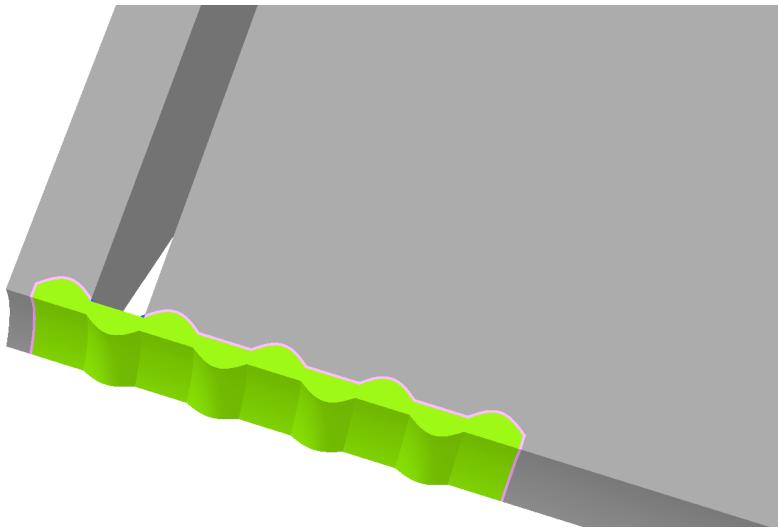


*Figure 14.183 Region of high electric field strength*

## Model with stress relief

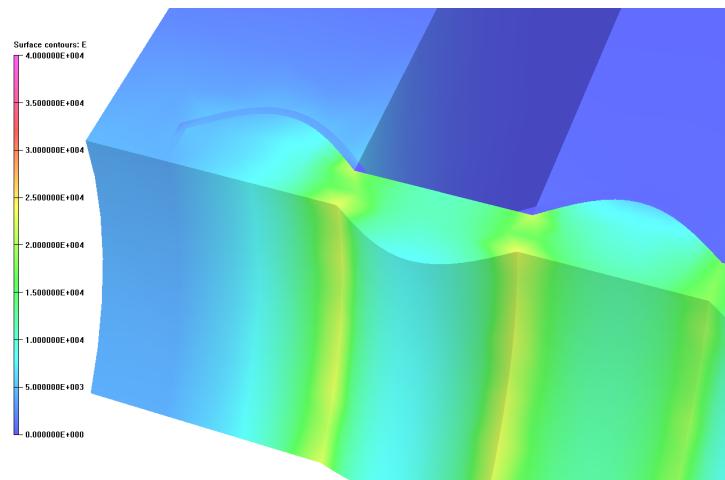
To overcome the high electric fields, a coating of stress-grading material can be applied over the insulator. Figure 14.184 shows the 45 degree section model with the stress grading material included. The properties of the material are  $\epsilon = 2$ ,  $\sigma = 10^{-6} \text{ S/m}$ . The material connects the high

and low voltage boundaries allowing a small current (because of the low conductivity) to flow and thus redistributes the electric field ([Figure 14.185](#)).



*Figure 14.184 Stress-grading material added*

**Static Current Flow** solves models with lossy dielectrics as a 2-stage process. This type of analysis must be selected by ticking the **Lossy dielectrics** option in Analysis Settings dialog of analyses that support this feature (**Static Current Flow** or **Electrostatic**). The current flow simulation is completed first and the resultant potential distribution assigned to the nodes for the electrostatic simulation. The resultant change in the electric field can be seen in [Figure 14.185](#).



*Figure 14.185 Electric field with addition of stress-grading*

The current density in the lossy dielectric material can also be evaluated, as shown in [Figure 14.186](#).

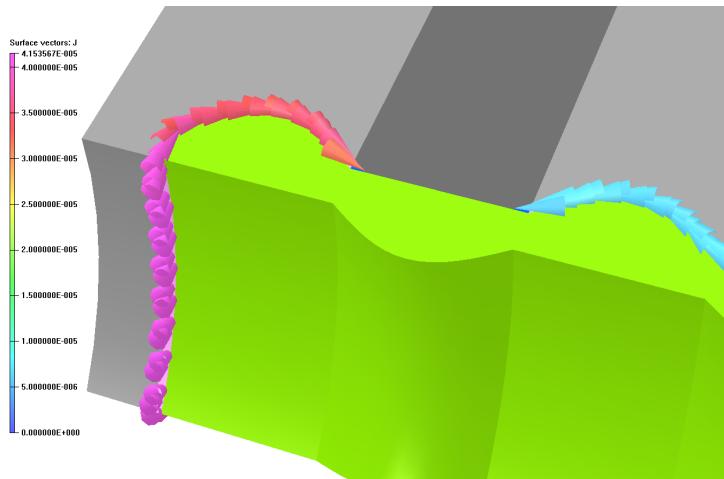


Figure 14.186 Current density vectors in lossy dielectric

## Multiphysics Analysis

A number of analyses require a combination of physical effects to be modelled, each needing a different type of solver. One example of this would be computing stresses due to a change in temperature and the resultant thermal expansion or contraction. Opera allows this type of analysis to be set up and run in a similar manner to a conventional, single solver simulation.

This application note will illustrate the analysis process with two simple example problems, namely combined thermal-stress and combined electromagnetic-stress (magnetostriction). The models for each example are available in the Opera examples folder, and include all of the data and settings required to perform the simulations. The first example will be considered in some detail, particularly where the process differs from that of a single solver simulation. The second example will only be introduced, and its use will largely be left for the user to explore.

It should be noted that these models are intended only to illustrate the process of performing multiphysics analyses in Opera; they have not been set-up to produce highly accurate results.

### Coupled Thermal and Stress

To demonstrate the way in which a combined thermal-stress multiphysics analysis can be performed, a model has been set up that analyses a cryostat support member. In this model, **TieRod.opc**, a glass fibre rod has been defined with two brass end pieces. One end is to be rigidly mounted on the OVC (outer vacuum case) at room temperature, 300K, while the other is rigidly mounted on a liquid nitrogen vessel at 77K. The model is shown in [Figure 14.187](#). The analysis will allow the temperature and stress distributions within the structure to be examined.

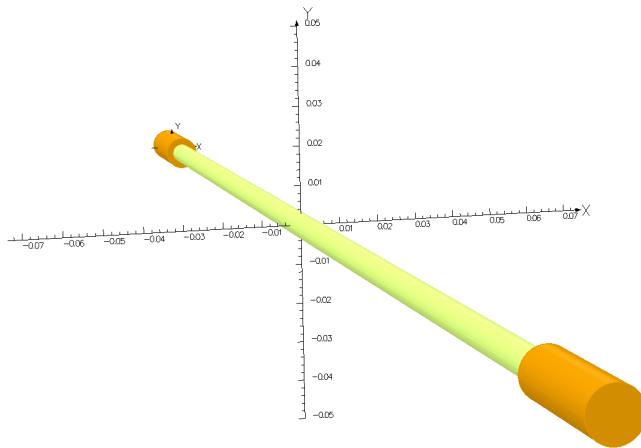


Figure 14.187 Cryostat Tierod model

A multiphysics analysis is setup by adding, one after another, different types of solvers in the Analysis Settings window. The individual settings for each of the solvers can be set independently, in the usual way.

In this example, set a **Static Thermal** solution followed by a **Static Stress**. The stress analysis will use the results of the thermal simulation to define the loading that is applied. In general, the sequence may be extended to add further analyses as required.

The following settings need to be changed for the **Static Thermal** case:

Field	Value
<b>Material Options</b>	<b>Use nonlinear properties</b>
<b>Maximum number of iterations (for a Newton-Raphson update)</b>	<b>21</b>
<b>Convergence criteria</b>	<b>1.0E-03</b>

Next, for the **Static Stress** case set the following options:

Field	Value
<b>Gravity / Direction</b>	<b>Ignore gravity</b>
<b>Use Thermal Expansion Coefficients</b>	<b>Yes (checkbox)</b>
<b>Reference Temperature</b>	<b>300</b>
<b>Operating Temperature</b>	<b>T</b>

As can be seen, non-linear material properties have been enabled to allow a user-defined material property variation with temperature.

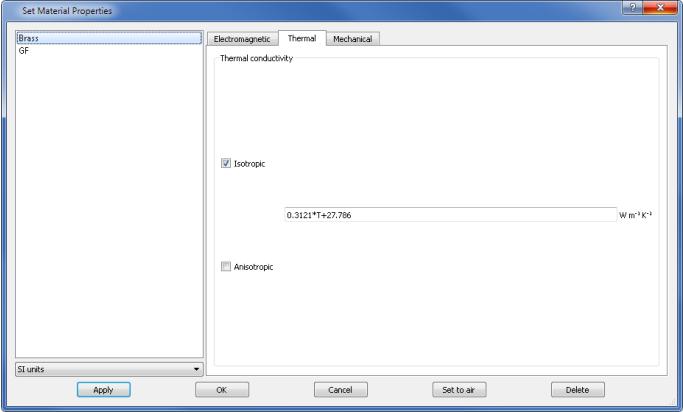
Material properties may now be set in the usual way by selecting  **Set Material Properties**. The resulting dialog allows the properties to be defined for all types of analysis that could be used in multiphysics analysis.

Material definitions for the various types of analysis are set in the relevant tabs within the dialog. In this example, the material **Brass** is given the thermal and mechanical properties shown below; the electromagnetic properties are not relevant in this case.



**Set Material Properties**

Select **Brass** then, on the **Thermal** tab, set the **Isotropic Thermal conductivity** to  $0.3121*T+27.786$ .



On the **Mechanical** tab, set the **Isotropic** properties as follows:

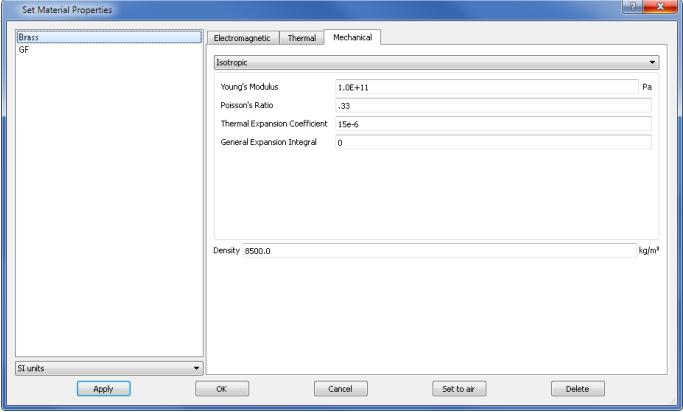
**Young's Modulus:**  $1.0E+11$

**Poisson's Ratio:** 0.33

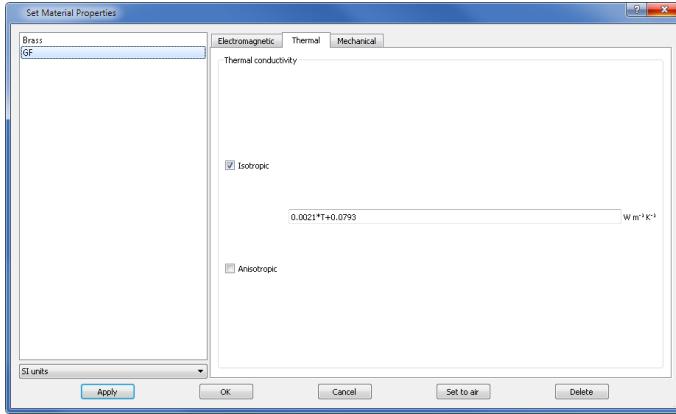
**Thermal Expansion Coefficient:**  $15E-6$

**General Expansion Integral:** 0

**Density:** 8500



Similarly, for the **GF** material: on **Thermal** tab, set the **Isotropic Thermal conductivity** to  $0.0021*T+0.0793$ .



On the **Mechanical** tab, set the **Anisotropic (orthotropic)** properties as follows:

**Young's Modulus:**  $1.0E+10$ ,  $1.0E+10$ ,  $2.0E+10$

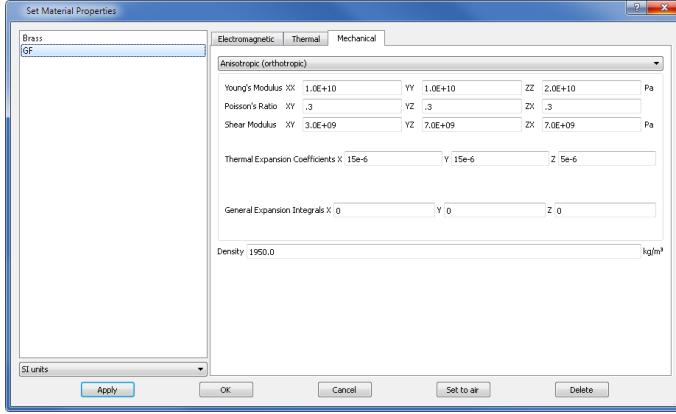
**Poisson's Ratio:**  $0.3$ ,  $0.3$ ,  $0.3$

**Shear Modulus:**  $3.0E+09$ ,  $7.0E+09$ ,  $7.0E+09$

**Thermal Expansion Coefficient:**  $15E-6$ ,  $15E-6$ ,  $5E-5$

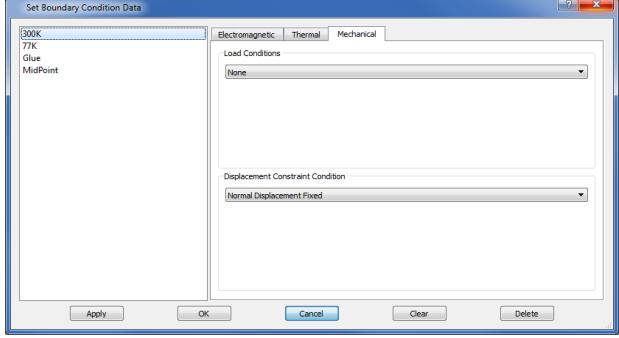
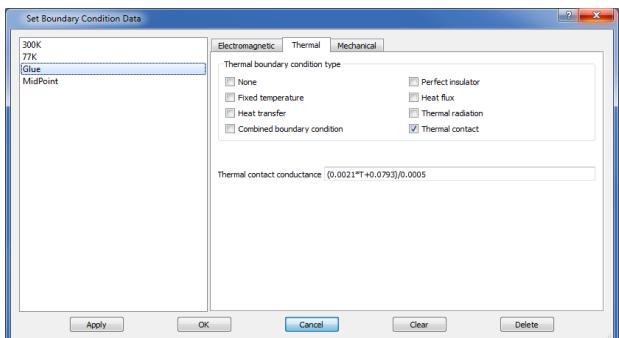
**General Expansion Integral:**  $0$ ,  $0$ ,  $0$

**Density:**  $1950$



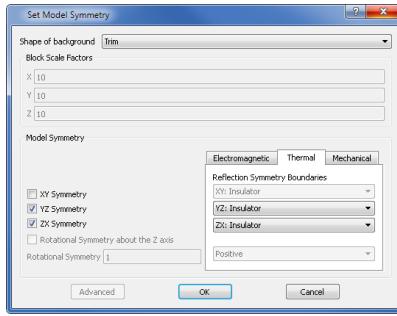
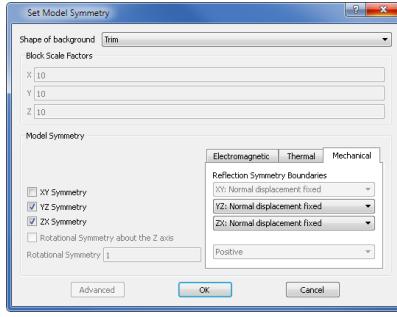
As can be seen, the material properties have been set to vary with temperature. The thermal conductivity functions have been defined to reflect the change in thermal conductivity with temperature: for the brass material the original data is a linear variation from  $59 \text{ W/(m K)}$  at  $100\text{K}$  up to  $113 \text{ W/(m K)}$  at  $273\text{K}$ ; and for the glass fibre of the tierod values of  $0.1 \text{ W/(m K)}$  at  $10\text{K}$  up to  $0.7 \text{ W/(m K)}$  at  $300\text{K}$ . Hence, simple linear functions have been specified to define these in the model.

Both thermal and mechanical boundary conditions need to be applied. To enable this, the end faces of the brass pieces have been labelled, one end being given the label **77κ**, the other **300κ**. The faces of the bore in the brass pieces that interface with the glass fibre rod are labelled as **Glue**, and represents imperfect thermal contact between the components.

 <b>Set Boundary Conditions</b>	<p>The thermal and mechanical boundary conditions are shown in the relevant tabs for each boundary label.</p> <p>On the Thermal tab, set the <b>300K</b> and <b>77K</b> boundaries to have fixed temperatures of 300 and 77K respectively.</p> <p>On the Mechanical tab, set the <b>Displacement</b> conditions for both to <b>Normal Displacement Fixed</b>, representing axially rigid mounting of the device.</p>  <p>Set the <b>Glue</b> boundary to have a <b>Thermal contact conductance</b> of <math>(0.0021*T+0.0793)/0.0005</math></p>  <p>All other boundary conditions should be set to <b>None</b>.</p>
---	--

The model has symmetry in the YZ and ZX planes. Taking advantage of this reduces the simulation time.

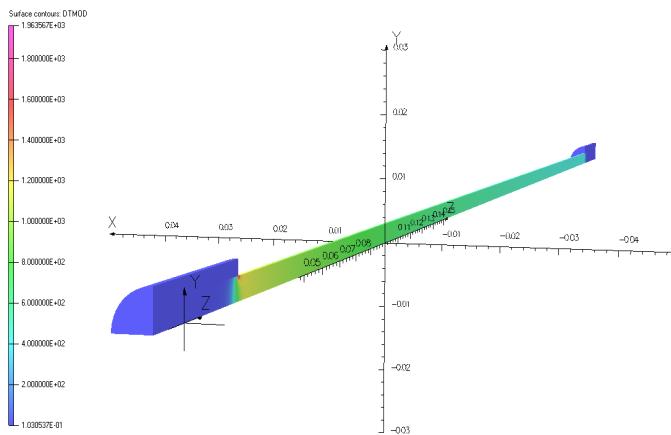
The model symmetry and thermal and mechanical symmetry conditions are applied using **Model Symmetry**.

 <b>Model Symmetry</b>	<p>Select <b>YZ Symmetry</b> and <b>ZX Symmetry</b>. On the <b>Thermal</b> tab, select the only choice: <b>YZ: Insulator</b> and <b>ZX: Insulator</b></p>  <p>On the Mechanical tab select <b>YZ: Normal Displacement Fixed</b> and <b>ZX: Normal Displacement Fixed</b></p> 
--	--

The model body may now be created and the mesh generated. For the example model, a surface mesh size of 0.00075 was chosen, and the simulation run in SI units.

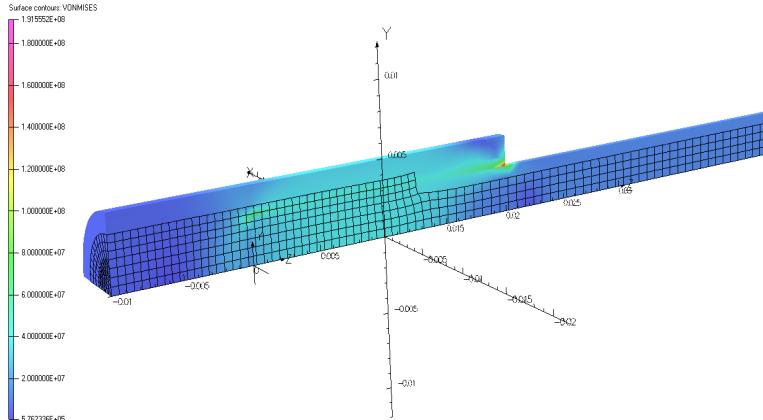
Following the simulations, the database will contain two simulation cases, the first being the thermal solution, the second the stress. When opened in the Post-Processor, the thermal solution loads first by default.

All of the usual functions of the Post-Processor relevant to a thermal solution are available. In [Figure 14.188](#) the temperature gradient is displayed at the cold end of the model. The effect of the nonlinear thermal conductivity is visible as an increase in the thermal gradient towards the cold end.



*Figure 14.188 Thermal gradient towards the 77K end*

With the second case in the database, all of the Post-Processor's stress-solution features may be used. For example, Figure 14.189 shows the von Mises stress contours and displaced mesh solution for the final model.



*Figure 14.189 Stress model results*

It should be noted here that the distortion of the device, indicated by the outline view of the geometry, is highly exaggerated. This is achieved by setting an **Automatic scale** with a relative factor of 0.2 in the  **3d Display** dialog.

## Coupled Electromagnetic and Stress (Magnetostriction)

The second multiphysics example shows the coupling between a magnetostatic and stress analysis, leading to the effect of magnetostriction on a sample. The model that has been constructed to show this is **Magnetostriction\_Example.opc**. This model contains a bar of Invar, which expands in the presence of an applied magnetic field, which is generated by a surrounding solenoid. Opera will compute the magnetic fields in the material and use them to calculate the material deformation.

The difference from the previous example comes from the requirement to surround the magnetic field model by an air volume, on the outside faces of which may be placed the far-field magnetic boundary conditions. This air volume takes no part in the subsequent stress analysis.

The model is shown in Figure 14.190.

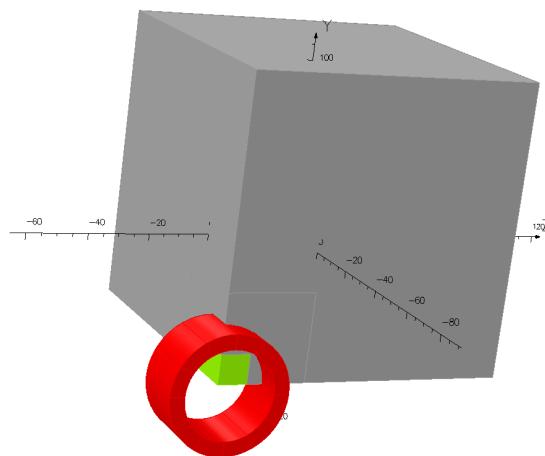


Figure 14.190 Magnetostriction model

The process of setting-up the model for multiphysics analysis is the same as for the previous case. If the settings in the magnetostriction model are examined, it will be seen that the main differences are the presence of electromagnetic boundary conditions - required since the **Magnetostatic** solver is used in place of the **Thermal** one- and the use of a **General Expansion Integral** in the **Material Properties** dialog to describe the effect of the magnetic field on the Invar material. In this example, the **General Expansion Integral** is defined in terms of an expression consisting of a user defined function, **Invar\_alpha(B)**. This function was generated from a table of values of flux density and expansion integral; its use allows the software to interpolate values. The data used are contained in **magnetostriction\_Invar.table**, which may be found in the Opera Examples folder.

Once the analyses have been run, the deformation may be observed as in the previous example. This is shown in Figure 14.191.

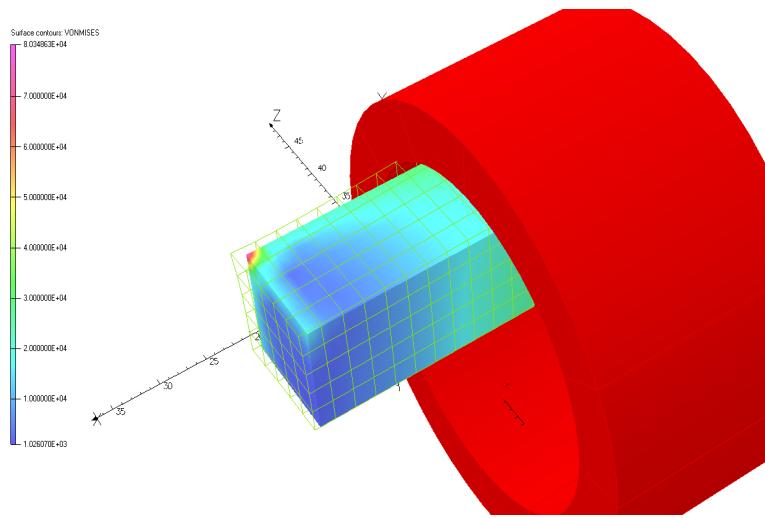


Figure 14.191 Magnetostriiction stress model results

## Using Biot-Savart Coils in Multiphysics (Magnetostatic and Stress) Simulations

A "pot core" model, similar to the one created in the 'Steel Plate Deformed by Energized Solenoid' example, is used to investigate the stresses and deformations in a Biot-Savart conductor. Along with the forces acting on the steel plate and core, the forces acting on the conductor can also be calculated.

The necessary steps needed to modify the **potCore\_noModelBody.opc** example to include forces on the Biot-Savart conductor are described below.

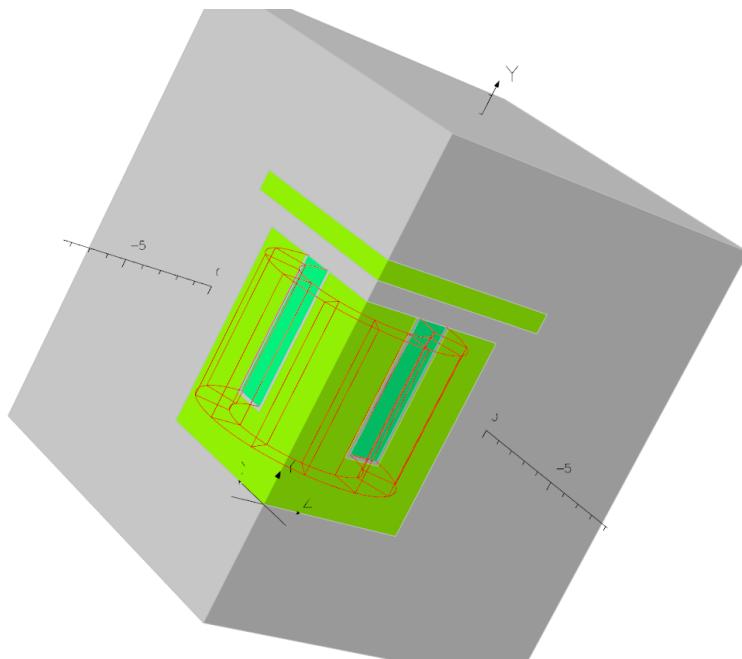


Figure 14.192 Pot core example with meshed Biot-Savart conductor region



If the Biot-Savart conductor is defined as a meshed region in the **Model Body Create** stage, cells are created in the shape of the conductor. The material label used for these cells is taken from the **Drive label** set for the conductor.

Load the **potCore\_noModelBody.opc** file from the examples directory. Make sure that the conductor properties are set as shown in the next dialog.

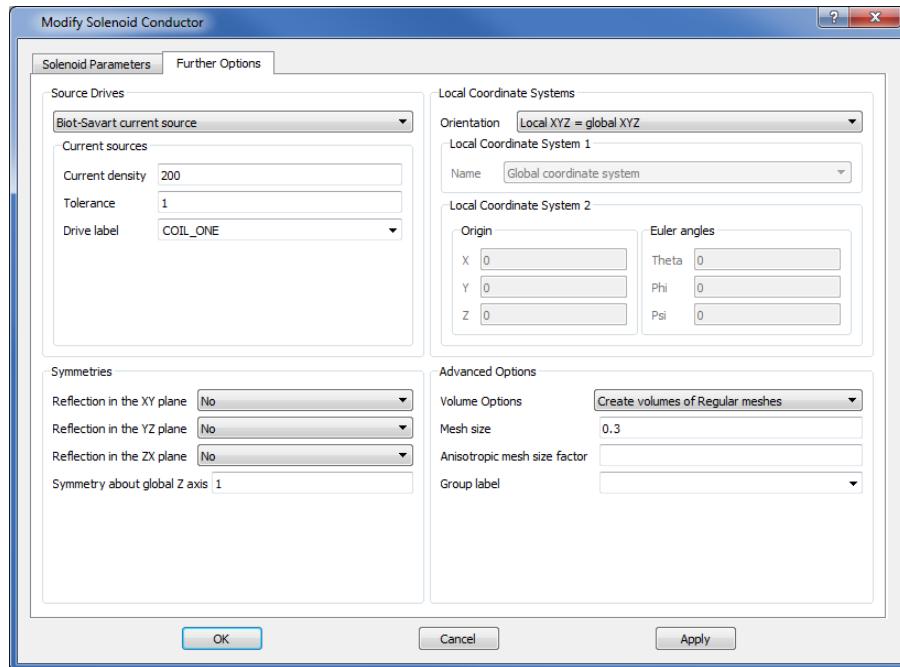


Figure 14.193 Meshed Biot-Savart conductor definition



### Material Properties

Once the model body is created, the new material label appears in the **Material Properties** dialog, where the mechanical properties can be defined. The format of this label is `<driveName>_Material` and the name assigned in this case is `COIL_ONE_Material`. It is worth noting that this material should not have any electromagnetic properties assigned to it since it is only supposed to represent the conductor in a stress analysis.

The material properties of the `COIL_ONE_Material` are shown in Figure 14.194.

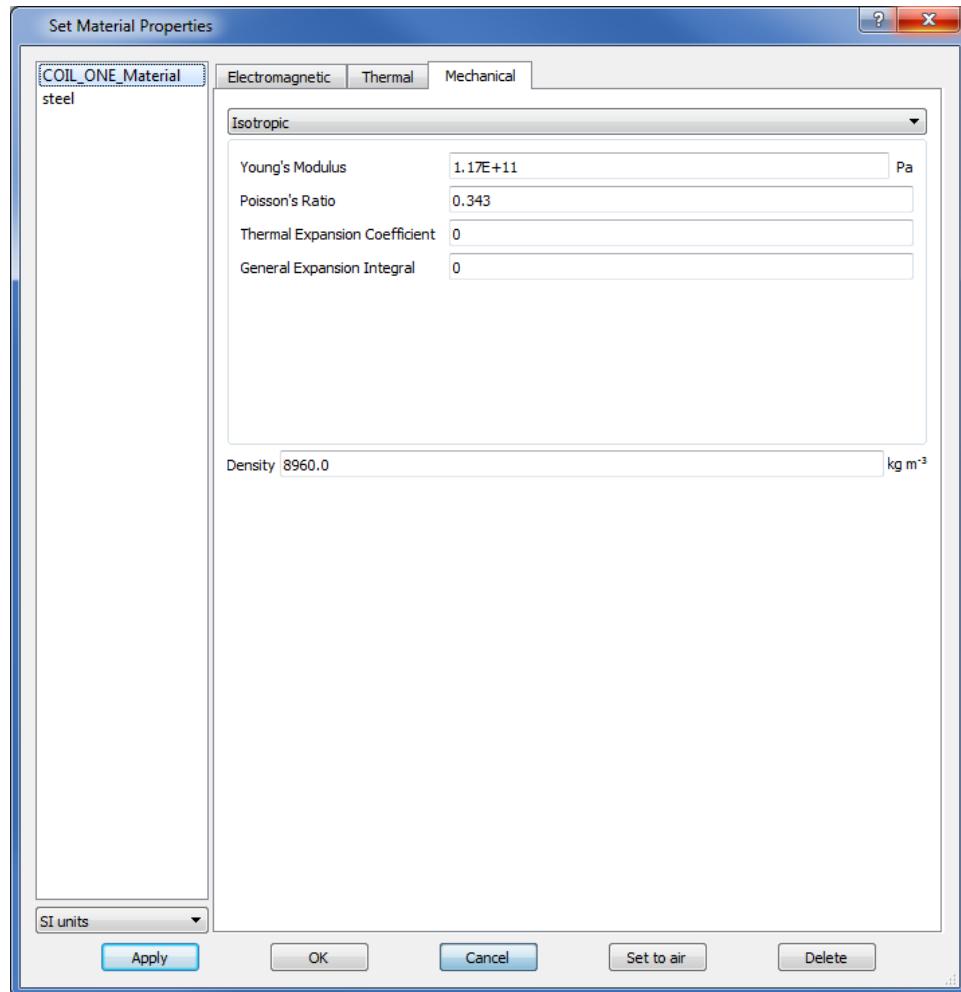


Figure 14.194 Mechanical properties assigned to the conductor region



Similarly, **Boundary Conditions** can be applied to the faces of the conductor. In this case a **Normal Displacement Fixed** boundary condition should be applied to the top face of the coil to provide mechanical support of the coil.



In order to do this press, **Select**, followed by **Clear all**, then select **Material COIL\_ONE\_Material** and **Display**. Make sure that the material body has already been created for this option to be available.

**Pick Faces** and pick the remaining top face of the coil, as shown in [Figure 14.195](#). Perform a right-click on the selected face to obtain the **Face Properties** menu. Enter a name for a boundary condition on the face, for example `coil_support`.

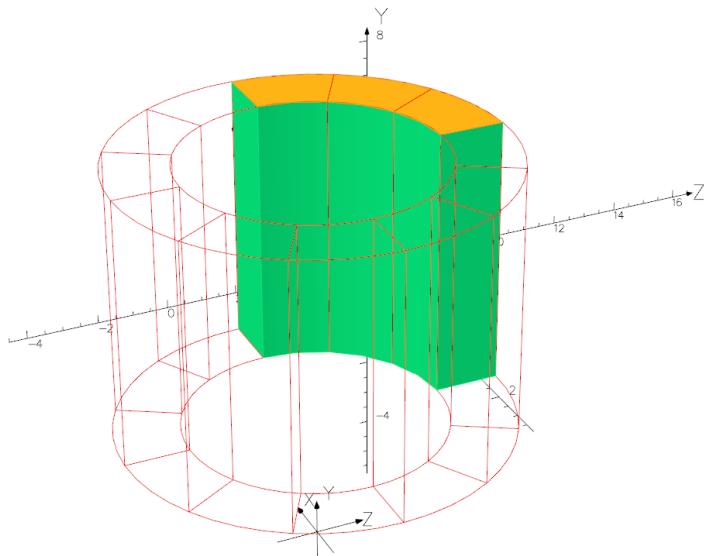


Figure 14.195 Coil support boundary condition

Then go to the **Boundary Conditions** menu to show the following dialog.

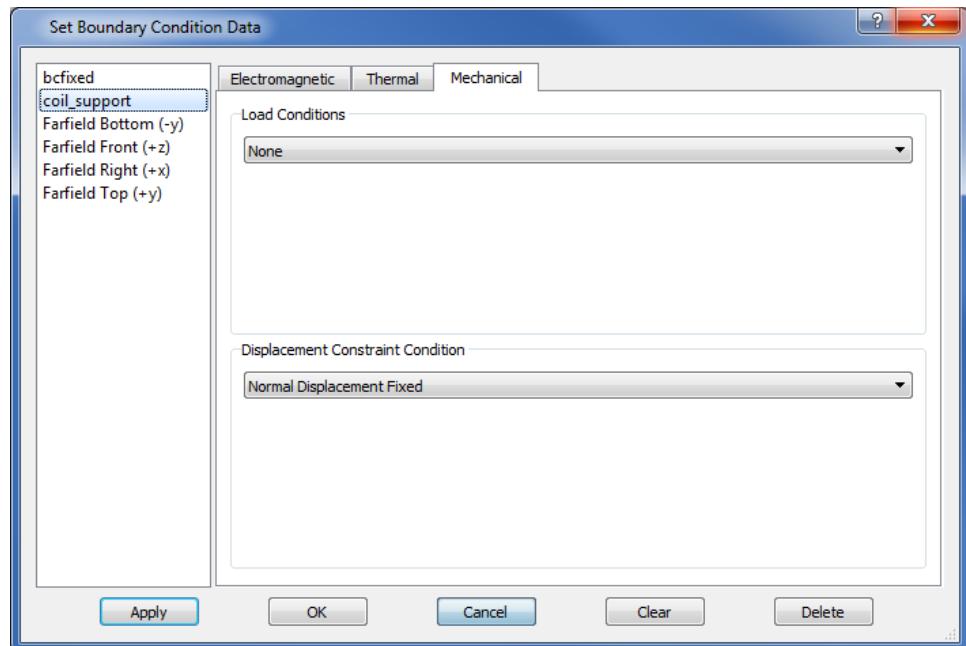


Figure 14.196 Normal Displacement Fixed boundary condition set on the top face of the coil

Set `coil_support` in the **Mechanical** tab to **Normal Displacement Fixed**.

Create a mesh with Mesh Generator Type I to achieve a hexahedral mesh in the coil.

Once the database is created and solved, stresses and deformations can be visualised in the **Post-Processor**. Along with the deformations of the core and plate (shown in the 'Steel Plate Deformed by Energized Solenoid' example) the conductor is now also deformed due to the forces computed in the **Magnetostatic** analysis.

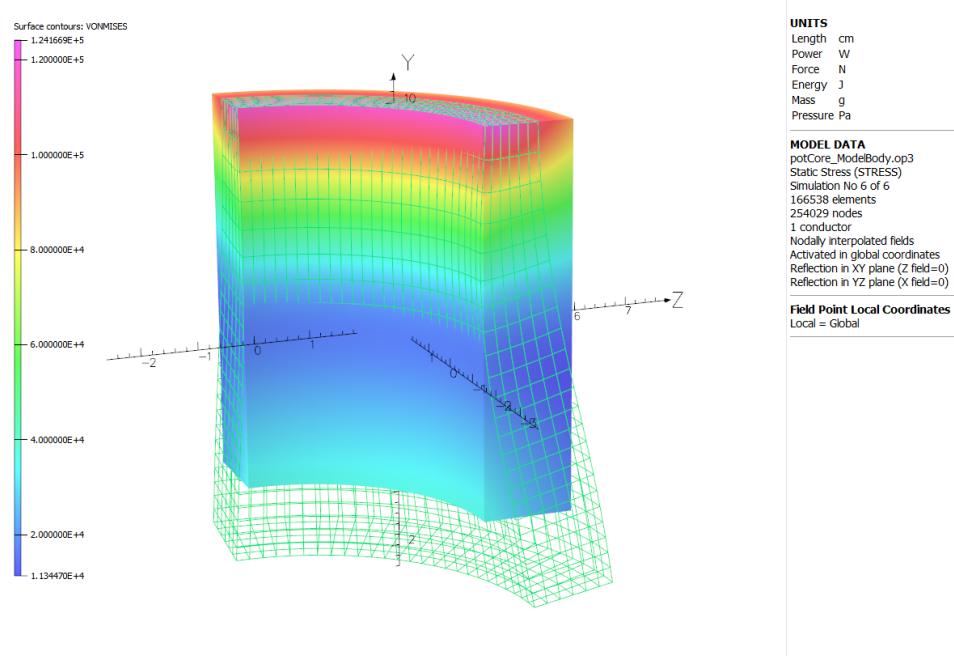


Figure 14.197 Von Mises stress and deformations on the Biot-Savart conductor

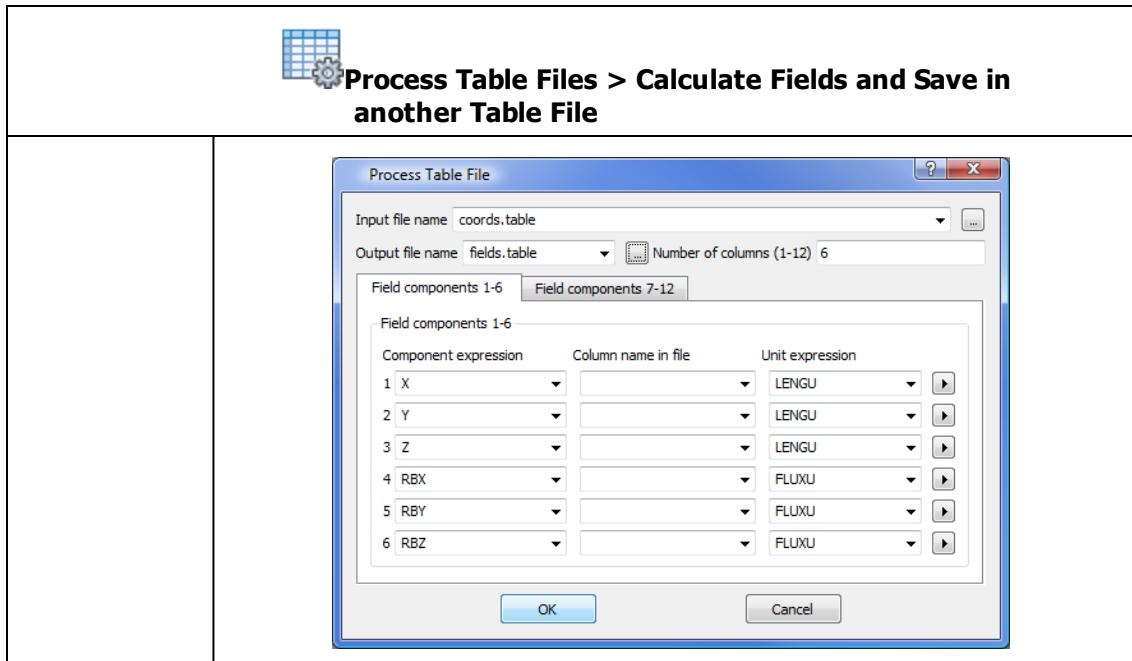
## Combined Magnetic and Electric Fields

The following describes the procedure required to calculate particle trajectories in combined magnetic and electric fields in the Post-Processor.

1. Solve the magnetostatic and electrostatic models separately; ***mag.op3*** and ***elec.op3*** will be used as the respective file names.
2. Now form the table of the node coordinates of the electrostatic model. Open ***elec.op3*** and then use the **TABLE** command as follows:



- Select a filename, for example ***coords.table*** (any filename is valid).
3. Open the ***mag.op3*** file and use the **TABLE** command to calculate the fields at the nodes of the electric field model by selecting:



For the **Input file name**, use ***coords.table*** (the file created in step 2), and for the **Output file name**, for example ***fields.table*** (any filename is valid). On the **Field components 1-6** tab set the output components to **X, Y, Z, RBX, RBY, RBZ**.

The entry in the second column is used as the column header in the table file. If the second column is left empty, the field components entered in the first column will be used instead.

Click on **OK** to process the table.

4. Re-Open **elec.op3** file and use the **TABLE** command to add the magnetic fields:



**Process Table File > Import Fields at all Nodes or Elements**

For the filename, specify **fields.table** (the file created in step 3). The magnetic field values will be automatically added and stored in the **elec.op3** database (note that there is no **Save** option in the Post-Processor).

5. The combined magnetic and electric fields option will now be available and may be switched on, when required, in the trajectory menu.

## Notes

A similar procedure can be used to add magnetic fields to an unsolved **Charged Particle** simulation prior to analysis. In this case, the table file of field values should contain values of **RHX**, **RHY** and **RHZ**. The **Charged Particle** solver uses the magnetic fields provided in this way in its trajectory calculations.

The magnetostatic and electrostatic models may be different. However, the electrostatic model must lie entirely within the volume of the magnetostatic model. If they weren't originally co-located this can be achieved when loading the models using the local coordinates options.

## Q and g (R/Q) Factors from a Modal HF Solution

The following describes the methods used to calculate Q and R/Q values for a resonant cavity solution solved using the **Modal High Frequency** solver.

### Definitions

The geometric impedance, g (sometimes known as R/Q) characterizes the efficiency of the geometry for particle acceleration. The units are SI.

$$g = \frac{R}{Q} = \frac{|V|^2}{PQ} \quad (14.51)$$

where

P is the power dissipated in the metallic surfaces,

Q is the quality coefficient,

V is the accelerating voltage seen by the particles, assumed to cross the whole cell at the speed  $\beta c$  and

$$V = \int E_z \cos\left(\frac{jwz}{\beta c}\right) dz, \quad (14.52)$$

c is the speed of light (m/s),

$\beta$  is the ratio of the speed of the particle to the speed of light,

$\omega$  is the angular resonant frequency and

$E_z$  is the z-component of the electric field on the axis.

The Q factor may be calculated as:

$$Q = 2w \frac{W}{P} \quad (14.53)$$

where W is the energy in the cavity.

The geometric impedance g is associated with the shunt impedance per unit length,  $R_l$  as

$$R_l = \frac{gQ}{L} \quad (14.54)$$

where g is the geometric impedance,

$$g = \frac{R}{Q} = \frac{\left\{ \int_0^{\frac{l}{4}} E_z \cos\left(\left(\frac{wz}{\beta c}\right)\right) dz \right\}^2}{PQ}, \quad (14.55)$$

L is the cavity length and

P is calculated over all the cavity wall in  $\lambda/4$  period.

## Command Files to Compute Q-factor and Geometric Impedance

The following **comi** file calculates the Q factor of a **Modal HF** resonant cavity<sup>1</sup>:

```
/Set SI units
unit len=metr,flux=tesl,fiel=am,elec=vm,
s=amp,v=wbm,co=sm,cu=am2,p=watt,
fo=newt,ener=joul

/ Select the PEC (metallic) surfaces

select type=surface label=pec

/Define variables for |Hxn|**2, omega,
/conductivity (for copper say 5e7 S/m), etc.

$parameter #hxnm (nz*hy-ny*hz)**2+(nx*hz-nz*hx)**2+(ny*hx-nx*hy)**2
$parameter #omeg 2*pi*freq
$constant #cond 5e7
$parameter #fact sqrt(#omeg*mu0/2/#cond)
$parameter #loss #fact*#hxnm

/Calculate surface integral and save in variable #i

surface -taverage #loss
$constant #i integral

/Calculate magnetic energy

energy

/Calculate Q-factor

$constant #q #omeg*energy*2/#I

/As a check, calculate the electric energy and Q-factor

volume -taverage comp=epsilon0/2*emod**2
$constant #q2 #omeg*integral*2/#i
```

The following commands can be used in conjunction with the above **comi** file. They calculate the geometric impedance.

```
/Calculate line integral for quarter wavelength length
/cavity setting #zend occurs at the peak field
/say 0.15 m) and beta (say 0.7)

$constant #zend 0.15
$constant #beta 0.7
line 0 0 0 0 #zend n=100
colour set code=9 1 1 1
plot comp=ez*cos((z-#zend)*#omeg/#beta/c)
```

---

<sup>1</sup>The same method is used by the **Cavity Q factor** toolbutton .

/Calculate the geometric impedance (R/Q),  
/using #symm (the symmetry copies around the  
/z axis say 8).

```
$constant #symm 8
$constant #imp integral**2/#i/#symm/2/#q
```

## Coupled EM and Thermal Analysis in Opera-3d

### Introduction

Opera-3d includes the **Harmonic Electromagnetic** and the **Harmonic High Frequency** modules for time varying electromagnetic field simulations and the **Static** and **Transient Thermal** modules for thermal simulations. A tutorial in this User Guide describes using the **Harmonic Electromagnetic** solver to calculate the heat sources for a **Transient Thermal** simulation. This application note shows how temperature dependent electromagnetic properties can be taken into account by coupling together the two simulations.

In most applications the time constants associated with changing electromagnetic fields are much shorter than the time constants of the thermal simulation. A weak coupling of the thermal and electromagnetic simulations is therefore suitable. The transient thermal calculation can proceed with a fixed heat source, until a significant temperature rise has occurred, at which time, the heat source must be recalculated as the electromagnetic properties will have changed significantly.

The necessary exchange of data between the **Transient Thermal** and **Harmonic Electromagnetic** simulations is performed in a fully automated way using scripts. The detection of the temperature change, and hence when the electromagnetic simulation must be recalculated, can however be performed in several ways. The example shown here demonstrates two such possibilities:

- Stopping the thermal simulation at specified intervals and loading it into the Post-Processor to monitor the temperature rise.
- Using the Opera Python interface to extend the solver functionality to monitor the temperature rise within the simulation itself.

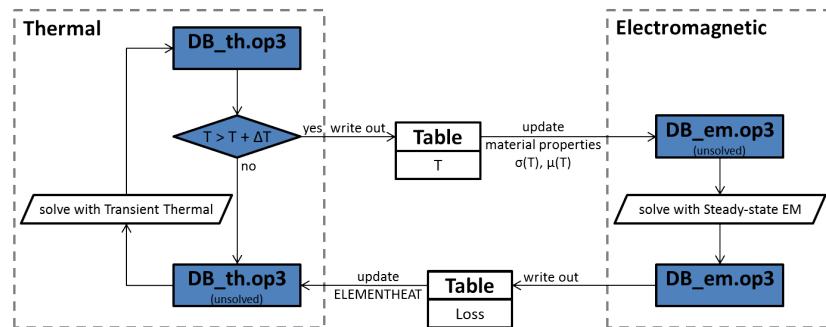
### Implementation

The Post-Processor command scripts for this example perform the following calculations:

- At an initial temperature, a metal work-piece has an initial permeability and an initial electrical conductivity. Using the **Harmonic Electromagnetic** solver, the ohmic losses in the metal are calculated.
- The ohmic losses from the **Harmonic Electromagnetic** solution are used as heat sources for the **Transient Thermal** simulation.
- The **Transient Thermal** simulation performs time-steps, using a constant (though spatially variant) heat source, until the next output time is reached. At this output time the maximum temperature rise in the metal is determined and compared with some pre-defined maximum.
- If the maximum temperature in the metal is higher than the defined maximum, a table file is written containing the distribution of temperatures, which is subsequently passed to the **Harmonic Electromagnetic** model to be included in the simulation.
- Material properties of the **Harmonic Electromagnetic** model are updated using the new temperature distribution, and the **Harmonic Electromagnetic** model is solved again.

- Heat sources are recalculated and exported to the **Transient Thermal** model.

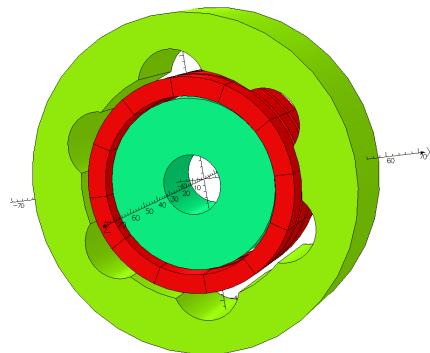
The process is repeated until a final output time is reached. [Figure 14.198](#) illustrates this process.



*Figure 14.198 Flowchart of the coupling between a **Transient Thermal** and an **Harmonic Electromagnetic** model*

## Induction Heating Example

In this example, an application of induction hardening in the automotive industry is presented. It is a constant velocity joint casing, where the inner surfaces are to be heated by eddy currents. [Figure 14.199](#) shows the assembly. A strong primary field is generated by 3 water-cooled solenoids.



*Figure 14.199 Assembly for inductive heating*

The solenoids are driven by a 10 kHz alternating current.

Due to the high frequency, the induced eddy currents are concentrated on the inner surface of the metal ring (know as the skin effect). As the temperature increases, the electrical conductivity of the metal reduces. This effect is taken into account in the modelling as follows:

$$\sigma_{op} = 4 \cdot 10^6 \cdot \frac{1}{T_{op}^{1.1861}} \quad (14.56)$$

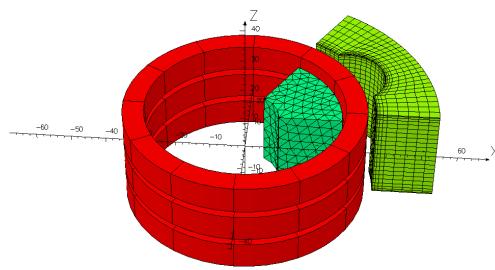
The numbers used above are derived from empirical data.

When the temperature reaches the Curie temperature of about 800° C, the permeability of the metal drops to 1. This is implemented in the script as follows:

$$\mu_{op} = 1 + \sqrt{\frac{800 - \min(T_{op}; 800)}{800}} \cdot 361 \quad (14.57)$$

The Finite Element model covers 60 degrees, as shown in [Figure 14.200](#). The geometry allows for tangential magnetic boundary conditions at 0 and at 60 degrees. A flux concentrator inside the coil bore increases the system efficiency.

Not visible in [Figure 14.200](#) are 3 additional solenoids, that are carrying negative currents. They have a smaller conductor cross-section than the original solenoids, and co-exist within the original solenoids. Where the solenoids overlap, the currents cancel out to zero, as if the original solenoids had hollow cross-sections. This is an easy way to model a water cooling channel inside a Biot-Savart conductor.



*Figure 14.200 FE-model for **Harmonic Electromagnetic***

## Scripts provided

Several command scripts (.comi), and a Python file (.py), are provided in the 3D examples folder, which can be found through the Opera Manager using

**File -> Open examples folder.**

Two of the command scripts are run in the Opera-3d Modeller. They create Finite Element models to be run in **Harmonic Electromagnetic** and **Transient Thermal** solvers, before sending the thermal model to be solved.

The other files (including the optional Python file) are used to control the coupling between the two analyses as described in the flowchart [Figure 14.198](#).

## Command script for 3d Modeller

The filename of the script for the Modeller is ***Coupled\_EM-TH\_Model-Build.comi***. The script builds the geometry from scratch, based on the dimensions given in ***Coupled\_EM-TH\_Model-Dimensions.comi***. All steps are documented with comments; the script is divided into uniquely named sections to ease navigation. The script generates 2 unsolved OP3 databases before sending the thermal database file **DB\_th.op3** to be solved using the \$exec command with the post-processing file **post-TH.comi**.

## Command scripts for 3d Post-Processor

Two of the command scripts are used to handle the data transfer between the two simulations. The filenames of the thermal and electromagnetic Post-Processor scripts are ***post-TH.comi*** and ***post-EM.comi*** respectively. These scripts are run automatically as part of the analysis. They are triggered at the post-processing stage of the solution.

## Solver.comi & Python

**Solver.comi** is used to register the `onSimuStart` function contained in the Python file **my\_hooks.py** to the `on_simu_start` hook within the software. The solver hooks allow user defined code to be run at specified places within the normal operation of the software, in this case at the initialization of the solution process. The function uses a user variable (`#usePython`) specified in the ***Coupled\_EM-TH\_Model-Build.comi*** to decide whether to load the additional functions `onTimestepStart` to the `on_timestep_start` and `onTimestepEnd` to the `on_timestep_end` hooks within the software. The function added to time-step start tells the solver to store the results of the next time-step temporarily in the database so they can be interrogated on time-step end. The function added to time-step end replaces the need to stop the thermal simulation at each output time in order to determine the temperature rise in the conducting parts of the model. Instead, the Python commands extract the temperature while the simulation is still running, and the thermal simulation only stops when a re-evaluation of the heat source is required.

Setting the value of `#usePython` to 0 (at the top of the ***Coupled\_EM-TH\_Model-Build.comi***) will use the traditional method of monitoring the temperature using the post-processor, while a value of 1 will enable the monitoring in Python.

## Notes

All parameters used in the scripts are only examples; the user will have to provide sensible numbers for frequency, current density, electrical conductivity, thermal conductivity, permeability, and Curie temperature.

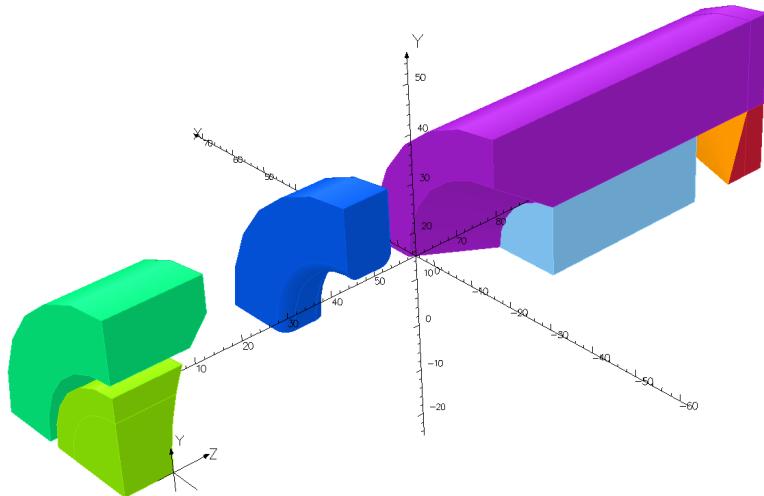
When running the scripts, it is recommended that the Opera Manager batch option to **Show Solver Window** is unticked so that the multiple solver windows are not shown.

## Particle Collision and Scattering in the Charged Particle solver

The **Charged Particle** solver includes a number of options for simulating the production of secondary particles, from both surface collisions and scattering within volumes. This application note describes these options, and indicates the physical meaning of the parameters that are required for the simulations. It is assumed that the user is familiar with the use of the Opera Modeller, **Charged Particle** solver and Post-Processor. The description below indicates the use of the GUI for data entry; all features and functions may also be specified from command files.

### Example geometry

The addition of secondary emission is demonstrated using a modified version of the model used in [A Charged Particle Example \[page 343\]](#). The geometry of the model has four-fold symmetry, a quarter section of which is shown in [Figure 14.201](#); some air regions have been hidden for clarity.



*Figure 14.201 Geometry of space charge model*

For the purposes of this note, the conical surface of the anode, coloured orange, is assigned surface secondary emission properties. The region of the beam line shown in pale blue has been given volume secondary emission properties.

## Surface Secondary Emission

Four types of surface secondary emission model are available in a Charged Particle analysis, namely:

- Backscattered
- Advanced Backscattered
- True Secondary Emission
- Advanced True Secondary Emission

Backscattering occurs when the trajectory of an incident particle is perturbed by the nucleus of an atom in a surface such that it re-emerges from the surface. This type of scattering is elastic or near-elastic, the backscattered particle then emerging with close to its incident energy, and often having been deflected through a large angle, perhaps nearly 180 degrees. All types of particle can undergo this type of scattering (for protons, the process is usually known as Rutherford backscattering).

True secondary emission occurs when an incident particle interacts with, and loses energy to, the atoms in a target material. If particles in the target gain sufficient energy they may be emitted. Unlike backscattered particles, true secondaries usually have low energy. At relatively low incident energies, electrons are the most probable secondary particles.

Secondary emitted particles might hit another emitting surface, in which case further secondaries could be produced. As will be described later, particles are represented in a Charged Particle analysis as beamlets, and the number of secondary beamlets produced at a secondary emitter is deterministic - it does not depend on the statistics of the event - these are embodied in the effective current and energy calculated for each beamlet. Allowing an unrestricted number of emission events for each beamlet has the potential to produce a very large number of particle tracks, many of which are likely to represent a negligible current. This is particularly so for true secondary emission, where the emitted particles usually have insufficient energy to cause further emission. To avoid this, secondaries are assigned a generation number, which is one higher than the generation number of the parent particle, and the maximum generation number may be limited. The limit is set in the Analysis Settings dialog for the **Charged Particle** analysis.

In the example used here, the maximum generation number is set to one (note that the generation number of particles produced by a **Primary** emitter is zero).

A secondary emission surface is defined in a similar way to a primary emitter. The surfaces required to be secondary emitters are given boundary condition labels in the **Face Properties** dialog ([Figure 14.202](#)). In the **Set Emitter Properties** dialog, shown in [Figure 14.203](#), a new emitter is added to the relevant surface and the **Emitter type** is set to **Secondary**.

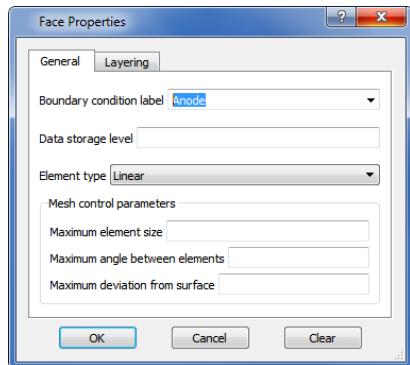


Figure 14.202 Add boundary condition labels in the **Face Properties** dialog

Multiple emitters may be defined on any labelled surface to allow the simulation of multiple particle species and/or multiple secondary emission models.

For each emitter, the required emission type is chosen from the four options given in the **Secondary Emitters** drop-down box in the dialog.

## Backscattered

**Backscattered** (Figure 14.203) is the simplest of the secondary emission models used within **Charged Particle** simulations.

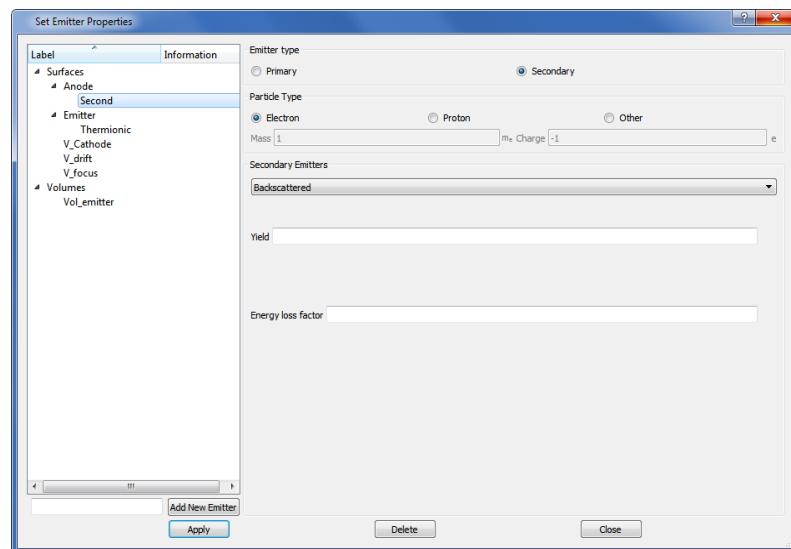


Figure 14.203 **Set Emitter** dialog for Backscattered emission

In the **Backscattered** secondary emission model, the secondary particles are emitted or reflected from the emission surface at the specular angle.

The only user supplied parameters are the **Particle Type**, **Yield** and the **Energy loss factor**.

**Particle Type** specifies the emitted particles. Since the mechanism simulated here is backscattering, a particular particle type will only be emitted if the incident particle is of the same type.

In a backscattering event, the incident particle can only produce a maximum of one secondary particle. The **Yield** represents the probability that a particle will be backscattered; each incident beamlet represents the aggregate current from a region of the emitting source. In the backscattering model, each incident beamlet produces exactly one secondary beamlet, and the **Yield** ( $\gamma$ ) is represented by the ratio of the current in the backscattered beamlet ( $I_o$ ) to the current in the incident beamlet ( $I_i$ ), i.e.,

$$\gamma = \frac{I_o}{I_i} \quad (14.58)$$

The collision process may be elastic or inelastic. In the latter, the incident particle loses energy to the surface. This is defined by the **Energy loss factor** ( $\eta$ ), which represents the ratio of the energy of the scattered and incident beamlets, i.e.,

$$\eta = \frac{E_o}{E_i} \quad (14.59)$$

Typically, the energy loss would be relatively low for **Backscattered** particles, giving values of  $\eta$  close to unity.

Both the **Yield** and **Energy loss factor** may be defined to be functions of the input beamlet parameters, the most useful of which are the incident polar angle (system variable **INTHETA**) and energy (system variable **INENERGY**). A full list of available system variables may be found in the **Opera-3d Reference Manual**.

## Advanced backscattered

In the previous emission model, the secondary particles are emitted only at the specular angle. The **Advanced Backscattered** model (Figure 14.204) allows both the energy and angular distributions of the secondary particles to be specified.

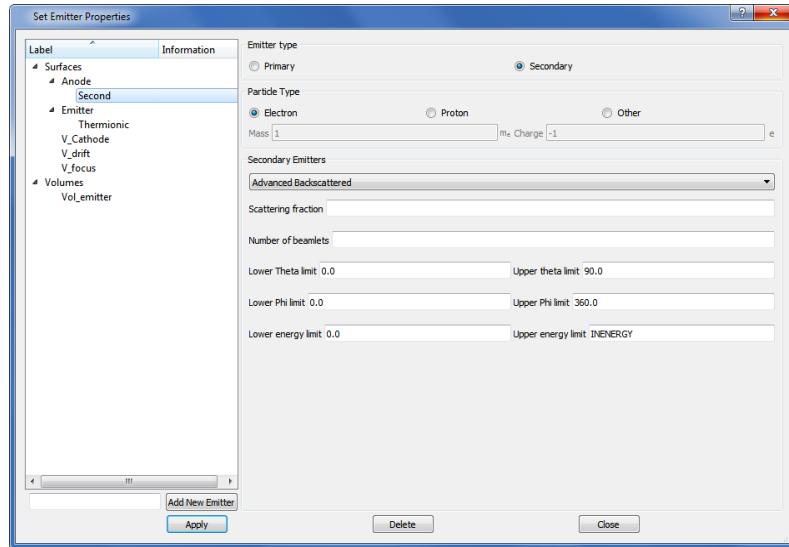


Figure 14.204 **Set Emitter** dialog for Advanced Backscattered emission

**Particle Type** specifies the particle that will be emitted, and as for the previous case, a particular particle type will only be emitted if the incident particle is of the same type

For this emitter type, the backscattered particles are distributed over angle and energy with a user defined distribution function, which is specified by the **Scattering fraction**. This represents the differential yield of the emitted particles (with units of  $\text{eV}^{-1}\text{sr}^{-1}$ ); its integral, with respect to relevant variables, is the total yield. In general, for a given emitter, the **Scattering fraction** will depend on the energies of the primary and secondary particles (**INENERGY** and **OUTENERGY**), and their incidence and emission polar angles with respect to the emitting surface normal (**INTHETA** and **OUTTHETA**).

The distribution is sampled by allowing each incident beamlet to give rise to a number of emitted beamlets (**Number of beamlets**). These beamlets are distributed over angle and energy space, within the bounds defined by the limits entered in the relevant fields. Each beamlet is assigned a value determined by the **Scattering fraction**.

Since this is a backscattering mechanism, one incident particle can only produce a maximum of one backscattered particle. Hence the **Number of beamlets** does not represent the actual number of backscattered particles; it is merely used as a way to populate the angle/energy space. Increasing its value will improve the accuracy of the solution, at the cost of increased run time. However, it should be noted that since a large number of primaries will usually be incident on a significant secondary emitter, the **Number of beamlets** (i.e. emitted) does not itself have to be large to populate the space adequately.

## True secondary emission

In the **True Secondary Emission** model (Figure 14.205), incident particles of one type can produce secondaries of the same or of different particle type. As in the previous cases, **Particle**

**Type** refers to the emitted particle that is produced by the specified **Incident particle type**.

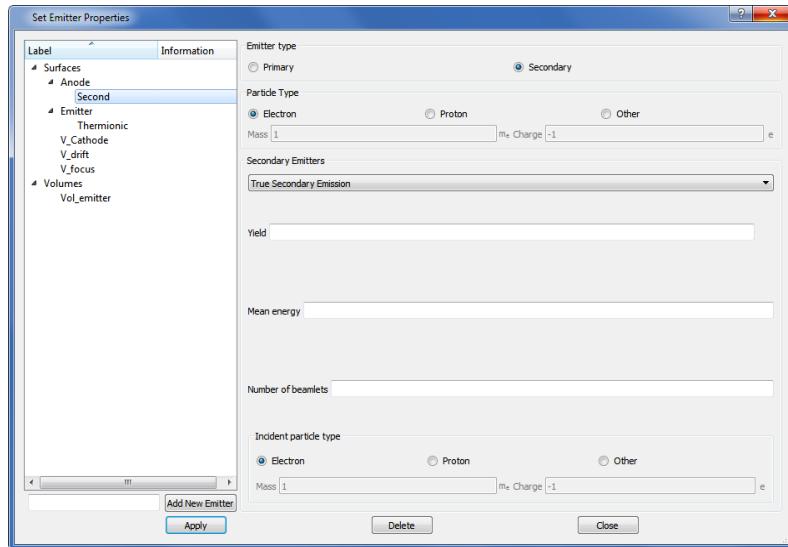


Figure 14.205 **Set Emitter** dialog for True Secondary Emission

As in the **Backscattered** case, the **Yield** is the total yield, and in terms of beamlets, is equivalent to the ratio of the emitted to incident current. Only the **Mean energy** of secondaries, rather than the distribution of energy, may be defined. The **Mean energy** should be specified in eV.

Secondaries are emitted at random angles, with an overall distribution that is a cosine function of the polar angle, with respect to the normal to the emission surface, and uniform over azimuthal angles. The distribution is independent of the incidence angle of the primary particle. As with **Advanced Backscattered** emission, each incident beamlet is considered to give rise to a defined **Number of beamlets**, which serve only to populate the angular space; they do not affect the total emitted current, which is determined by the value of **Yield**.

### Advanced true secondary emission

The definition of the parameters **Particle Type**, **Emission fraction**, **Number of beamlets** and the **Theta/Phi** limits for the **Advanced True Secondary Emission** model (Figure 14.206) is the same as those for the **Advanced Backscattered** model. However, in the latter, a backscattered particle is constrained to be of the same type as that of the incident particle, whereas in **Advanced True Secondary Emission**, the primary and secondary particles may be selected independently. As for **True Secondary Emission**, **Particle Type** refers to the emitted particle that is produced by the specified **Incident particle type**.

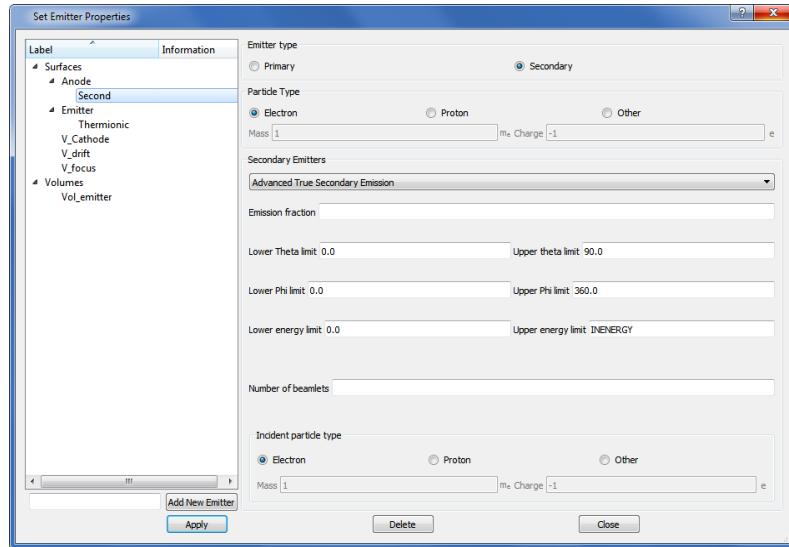


Figure 14.206 **Set Emitter** dialog for Advanced True Secondary Emission

## Volume Secondary Emission

If a volume in a **Charged Particle** model is given a volume data label, it may be assigned volume emission properties<sup>1</sup> in the **Set Emitter Properties** dialog, in the same way that surface emission is defined. Note that in this instance, the material of the emitter should be set to **Air** to allow beamlets to propagate. Alternatively, the volume may be given the additional label **BEAMPASS** (using **Additional Labels** on the Modeller menu bar).

Three types of volume secondary emission model are available in the **Charged Particle** solver, namely:

- Backscattered
- True Secondary Emission
- Advanced True Secondary Emission

The parameters specifying these scattering models are analogous to those used for surface scattering.

### Backscattered

In a volume **Backscattered** simulation (Figure 14.207), a beamlet propagating through a region of volume emission will progressively lose both current and energy; there is no concept of emission. The incident beamlet is not deflected in this process, the change in beamlet current and energy will result in a modified trajectory.

---

<sup>1</sup>The use of volume emitters in Opera Charged Particle solver is a separately licensed feature.

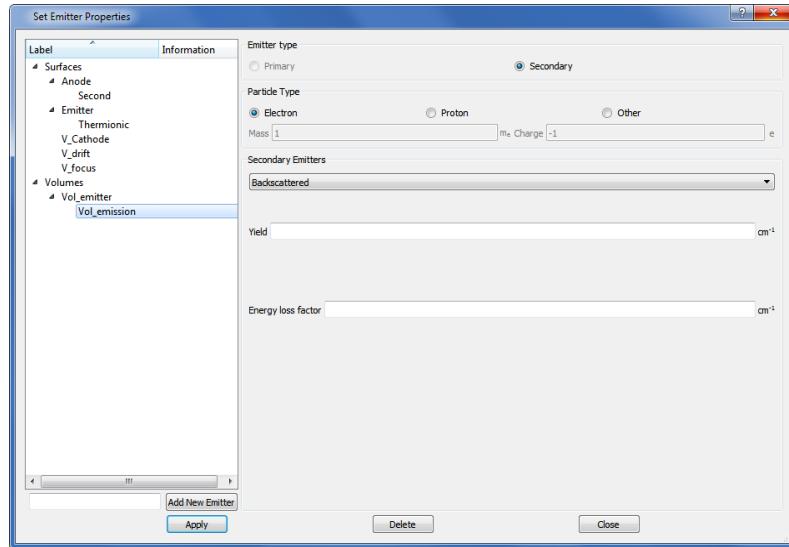


Figure 14.207 **Set Emitter** dialog for volume Backscattered Emission

As in surface **Backscattered**, the only user supplied parameters are the **Particle Type**, **Yield** and the **Energy loss factor**.

**Particle Type** specifies the emitted particles. Since the mechanism simulated here is backscattering, a particular particle type will only be emitted if the incident particle is of the same type.

The **Yield** represents the probability that a particle will be backscattered per unit length (s) of the propagation path, so the **Yield** ( $\gamma$ ) determines the rate of change of current (I) in the incident beam, i.e.,

$$\gamma I = \frac{dI}{ds}. \quad (14.60)$$

The scattering involves a loss of incident beam energy (E) per unit length, specified by **Energy loss factor**,  $\eta$ , where

$$\eta E = \frac{dE}{ds}. \quad (14.61)$$

Note that both **Yield** and **Energy loss factor** are defined with units of  $\text{cm}^{-1}$ , regardless of the units selected for the simulation. Both should be negative, and would normally be small.

## True secondary emission

In a volume **True Secondary Emission** simulation (Figure 14.208), a single particle will produce a number of particles per unit propagation length in the emitting medium. This is specified by the **Yield**. These secondaries are emitted with a uniform random distribution about the incident particle, with an average energy given by **Mean Energy** (in eV). Within the simulation, the number of

particles emitted per unit length is known, allowing the total secondary yield and energy to be calculated.

Scattering events are randomly distributed along the primary beam trajectory, and at each event, a **Number of Beamlets** may be specified to improve the population of the emitted distribution.

As previously, the trajectory of the incident beamlet will be modified by the change in current and energy.

Note that **Yield** is defined with units of  $\text{cm}^{-1}$ , regardless of the units selected for the simulation. The **Mean energy** of true secondaries would normally be small.

As for surface **True Secondary Emission**, the **Particle Type** and **Incident particle type** may be chosen independently.

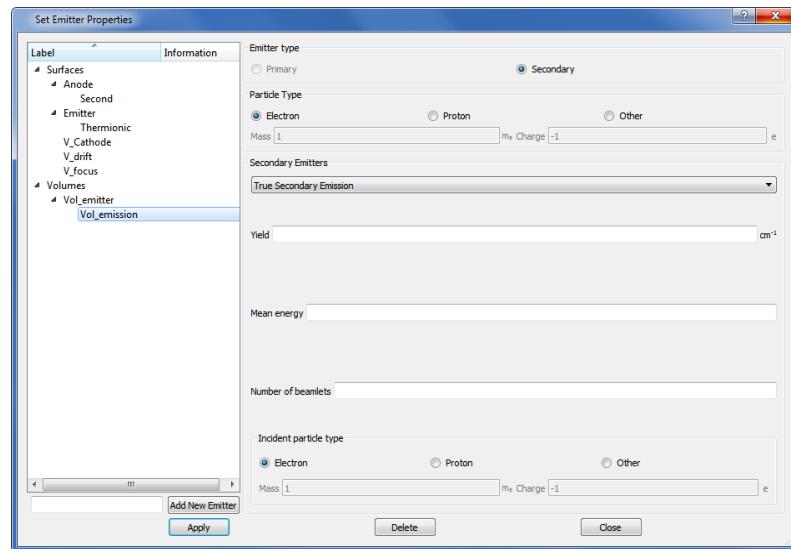


Figure 14.208 **Set Emitter** dialog for volume **True Secondary Emission**

## Advanced True secondary emission

In a volume **Advanced True Secondary Emission** simulation (Figure 14.209), the distribution of secondaries is defined by the functional variable **Emission fraction**. The emission may be restricted to lie between defined theta, phi and energy limits, where the polar angle, theta, is measured from the primary beam direction.

Scattering events are randomly distributed along the primary beam trajectory, and at each event, a **Number of Beamlets** may be specified to improve the population of the emitted distribution.

As previously, the trajectory of the incident beamlet will be modified by the change in current and energy.

The **Upper energy limit** of true secondaries would normally be small.

As for surface **Advanced True Secondary Emission**, the **Particle Type** and **Incident particle type** may be chosen independently.

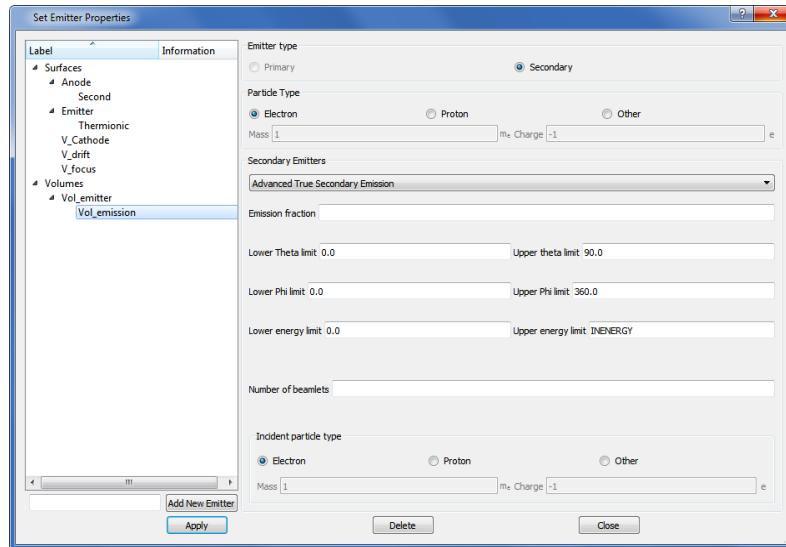
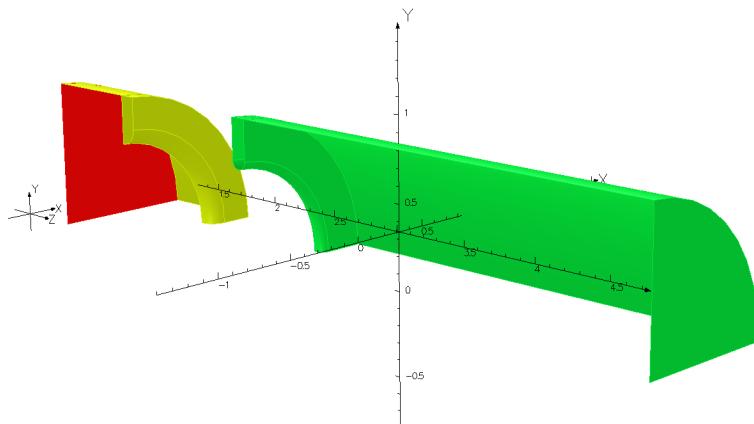


Figure 14.209 **Set Emitter** dialog for volume Advanced True Secondary Emission

## Plasma Free Surface - Type 103 Emitter

The type 103 emitter is used to simulate the emission of ions from the meniscus of a plasma, which forms in the vicinity of the aperture of the plasma chamber. In this type of emitter, the emission surface (the meniscus) does not coincide with any physical geometry - the emission surface is determined by the potential distribution from both the electrode fields and the beamlet space charge. The Type 103 emitter definition is typically attached to the rear wall of the plasma chamber. A separate boundary condition label from the rest of the plasma chamber must be defined on this wall so that an emitter may be attached to it, but these boundaries must be at the same potential. For example, in [Figure 14.210](#), the red and yellow boundaries are both set to ground potential, but they are specified separately so that an emitter may be attached to the red surface.



*Figure 14.210 Applied boundary conditions for a plasma free surface model*

The user defines a Plasma Surface Type 103 emitter on this surface and specifies the Bohm current density, meniscus voltage and ion temperature. A typical set of emitter parameters is shown in [Figure 14.211](#). The program determines the position of the plasma boundary, by tracking beamlets normal to the emitter surface until they reach a position where the voltage (relative to the emitter surface) is equal to the specified meniscus voltage. The ions are then tracked from the plasma boundary.

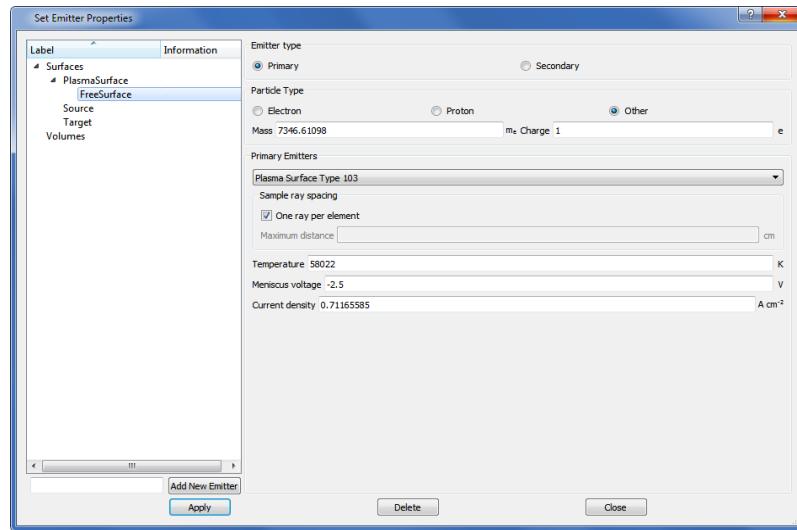


Figure 14.211 Emitter dialog with typical parameter values

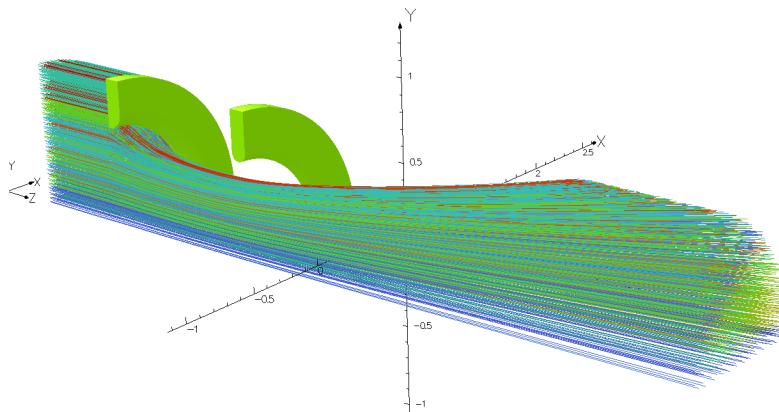
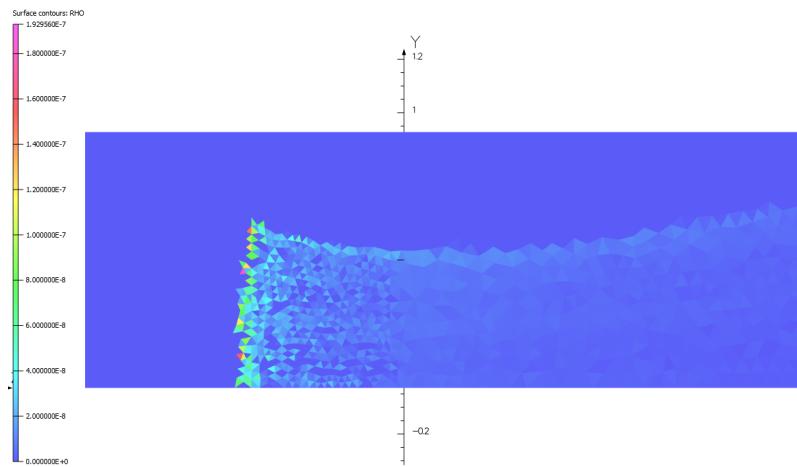


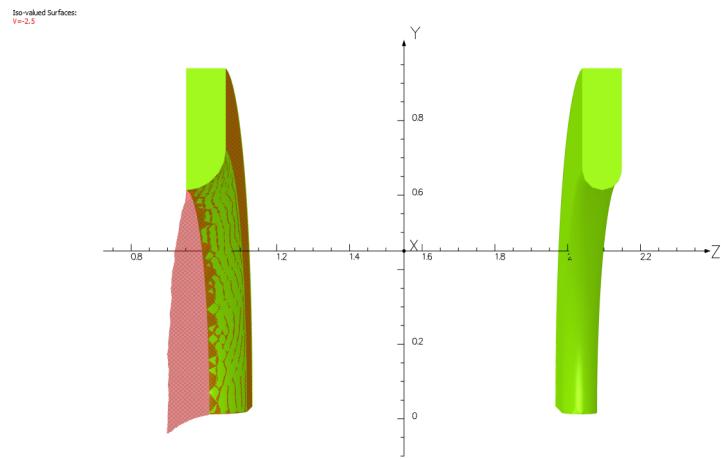
Figure 14.212 Trajectories in the solved plasma free surface model

The meniscus potential used by the Charged Particle solver is specified as the position beyond which ions begin to contribute space charge to the solution, i.e. it is the boundary of the plasma. The potential should be of the order of  $kT_e/2q_e$  (where  $T_e$  is the electron temperature), although the exact value is left for the user to specify. No space charge is considered behind this boundary from any source within the plasma. Note that the temperature specified by the user is used only to determine the initial energy of the ions at the plasma boundary and is not linked to the specified meniscus potential; this allows an additional flexibility.



*Figure 14.213 Space charge in the solved plasma free surface model*

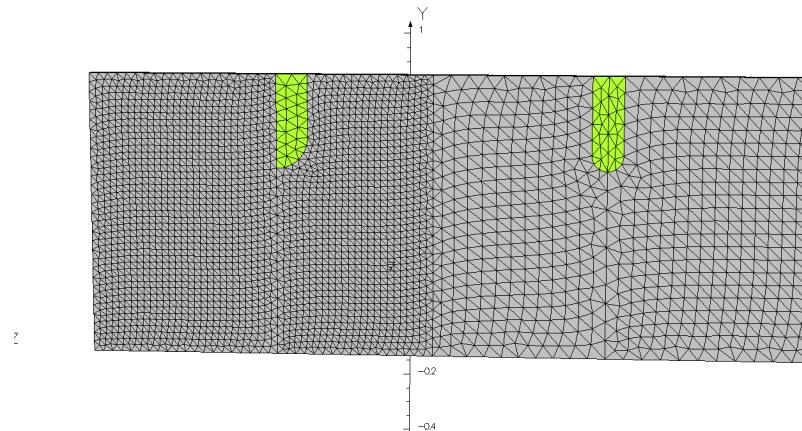
On subsequent iterations the space charge associated with the ion trajectories modifies the voltage distribution and therefore the position of the plasma boundary. After a number of iterations this process converges on a self-consistent space charge/plasma boundary.



*Figure 14.214 Isopotential surface at the meniscus voltage*

For the type 103 emitter, convergence is dependent on the solver being able to accurately model the meniscus, and also on being able to capture the change in meniscus position between iterations. In order to do this, it is advised that the expected meniscus region (typically close to, and slightly behind or ahead of, the aperture of the plasma chamber) be finely meshed with quadratic elements. A mesh

size of 1/20 of the smallest aperture dimension is usually sufficient, but a smaller element size may be required if there is significant curvature of the meniscus.



*Figure 14.215 Mesh around the plasma chamber and meniscus*

Because the plasma free surface emitter ignores much of the behaviour behind the meniscus, it is not necessary to model the entire plasma volume. Instead, the plasma chamber can be truncated so that the emitting surface lies behind the source electrode (i.e. the chamber aperture electrode) by a distance approximately equal to the largest aperture dimension, as shown in [Figure 14.215](#). This keeps the model size small while ensuring that the fixed potential boundary condition of the emitting surface is far enough away from the electrode that it does not interfere with the potential distribution in that region - having the surface too close to the aperture may prevent the meniscus from forming far enough into the plasma chamber, leading to problems with convergence or accuracy of results.

## Plasma Emitter

In a general plasma sputtering process, a plasma is generated in a working gas, typically argon, between two electrodes contained within a chamber. The relatively heavy argon ions from the plasma are accelerated towards the cathode - known as the target - where they cause emission (sputtering) of neutral target material atoms, ions and electrons. The neutrals, in particular, travel ballistically and deposit on internal surfaces, eventually forming a thin film coating. The chamber also contains the item intended to be coated; this is usually known as the substrate, and is typically held at anode potential.

This simple device suffers from a number of drawbacks, in particular, a low pressure is required to maintain a high quality coating, but this results in very low sputtering rates, and correspondingly low rates of deposition. The magnetron sputtering technique is intended to overcome this by magnetically confining the electrons in the plasma close to the target, where they experience an extended interaction length. This increases the rate of generation of gas ions at a given gas pressure, so increasing the rate of sputtering and deposition.

The Plasma Emitter has been developed to simulate low density magnetized plasmas, and is intended to allow modelling of such devices as the magnetron sputter coater. The simulation employs a self-consistent plasma boundary method that is a development of the technique used in Opera to model emission from a [Plasma Free Surface - Type 103 Emitter \[page 772\]](#).

The method is described as self-consistent because the solutions for the plasma boundary surface, ion and electron currents are consistent with the potential distribution caused by the space charge outside the plasma volume. Like the Type 103 emitter, the emission surface does not coincide with any physical geometry. Ions from the plasma are assumed to be emitted from the plasma meniscus. However, unlike the Type 103, electrons are tracked into and in the plasma, where they are taken to be the dominant source of gas ionization.

### Magnetron geometry

A cut-away view of an example magnetron is shown in [Figure 14.216](#). It comprises a disc-shaped target (green), surrounded by a conducting case (light grey), above which is a similarly shaped substrate (blue). An air region surrounding these defines the chamber volume in which the plasma will form. The model file, **magnetron\_example\_model.opc**, is available in the Opera Examples Folder.

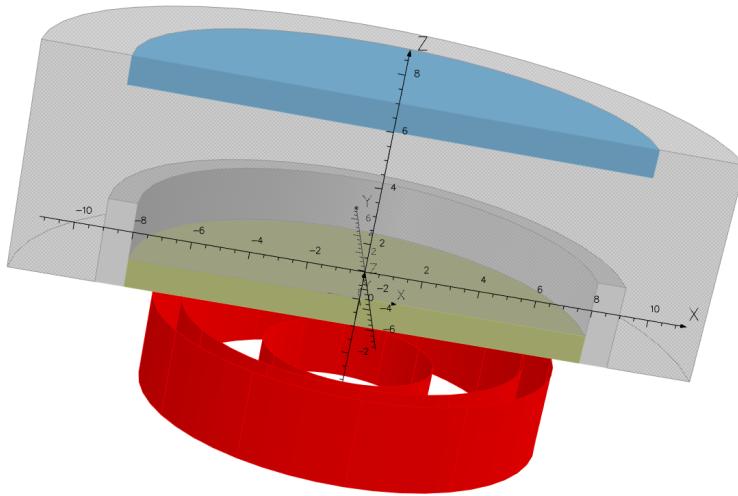


Figure 14.216 Geometry of the magnetron example

The outside faces of the air volume are given conducting boundary conditions to represent the chamber walls. Since both the target and substrate are taken to be conducting in this example, their faces and the faces of the case are also given conducting boundary conditions. The target and case are assigned a potential of -350V, while the substrate and chamber walls are set to 0V.

A magnetron requires both electric and magnetic fields, the latter most often being provided in practice by an array of permanent magnets arranged below the target. This can easily be simulated in Opera by importing the magnetic fields from a **Magnetostatic** solution into the **Charged Particle** database before running the latter simulation. However, for the simple case illustrated here, it is convenient to use Biot-Savart conductors to generate the magnetic field during the **Charged Particle** simulation. The current density in the conductors has been set to suitable values.

The model is set up in the same way as for any other **Charged Particle** simulation. However, a number of the default analysis settings might need to be changed. In particular, the maximum number of iterations might need to be increased, and the nonlinear iteration convergence tolerance loosened. The default convergence tolerance of 0.001 is overly stringent in this type of simulation, and a value of 0.1-0.2 may be used without significant loss of results accuracy; this corresponds to a tolerance in the residual of the order of  $10^{-4}$ .

In this particular example, the maximum number of iterations is set to 41 and the convergence tolerance to 0.2.

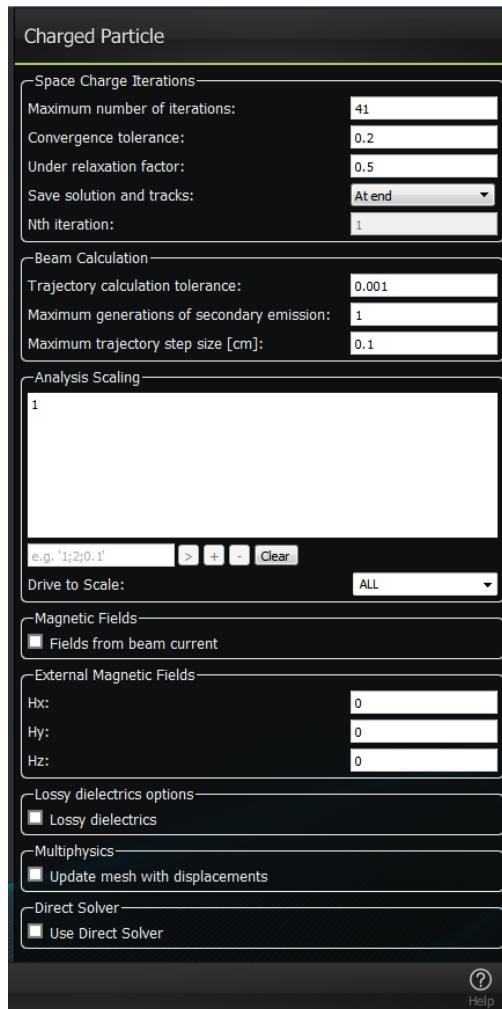


Figure 14.217 Analysis options

In addition, it should be noted that the voltage distribution in the plasma chamber is strongly affected by the plasma. Under these circumstances, the allowed voltage change between iterations should be limited to avoid spurious oscillations and subsequent failure to converge. This can be effected by defining the user variable **#MaxVoltChange**. The chosen value for this should not be too small, or convergence will be overly slow; in the present example a value of 50V is suitable.

During the simulation, electrons will be tracked, including in the plasma region. The electrons will spiral around the magnetic field and will tend to drift around the central axis of the model. In principle, the electron tracks could be extremely long, requiring a large computational resource. However, while in the plasma region, the electrons are interacting with the gas, causing ionization and losing current and energy; once the current falls to a low value, the beamlet may be terminated. The user variable **#MaxtrackLen** can be set to achieve this. A value of 10000 (steps) is used in this example.

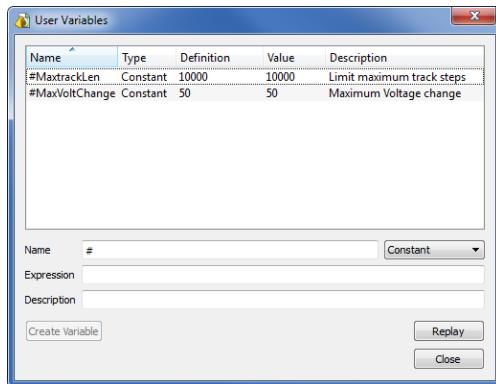


Figure 14.218 User variables set for the example

## Emitters

Four emitters are used in this model, two attached to the upper face of the target, and two attached to the chamber air volume.

The surface emitter parameters are shown in Figure 14.219 and Figure 14.220. Both emitters are true secondary emitters and represent generation of particles from incident argon ions. For the first, labelled **E\_Ar**, the secondaries are electrons; for the second, **Sputtered**, they are neutral, sputtered target atoms.

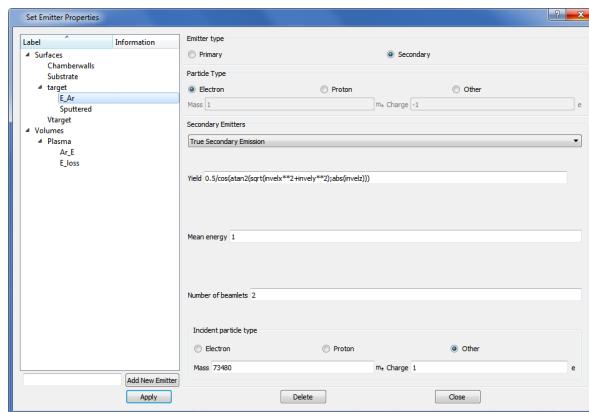


Figure 14.219 Surface emitter **E\_Ar**

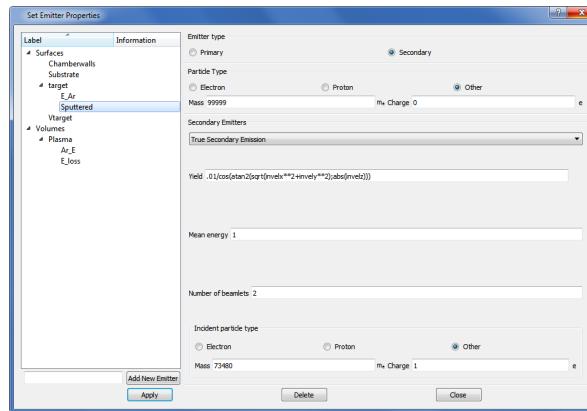


Figure 14.220 Surface emitter *sputtered*

**Figure 14.221** and **Figure 14.222** show the volume emitters. The first, **Ar\_E**, is the Plasma Emitter, which models the interaction of the electrons with the magnetized plasma. The parameters required for this emitter are relatively simple:

**Ionization energy** depends on the working gas; here the first ionization energy of argon is used.

**Ionization as a fraction of losses** accounts for mechanisms other than ionization, such as excitation into metastable states, that lead to electron energy loss. Its value depends on the physical properties of the working gas and of the ionizing particles.

**Electron Temperature** is set to a typical value for a low number density plasma.

**Total plasma current** is a basic input parameter - it is the user defined operating current, and together with the electrode potentials, defines the power requirement of the device.

The final emitter, **E\_loss**, describes the rate at which electron beamlets lose current and energy by interaction with the plasma.

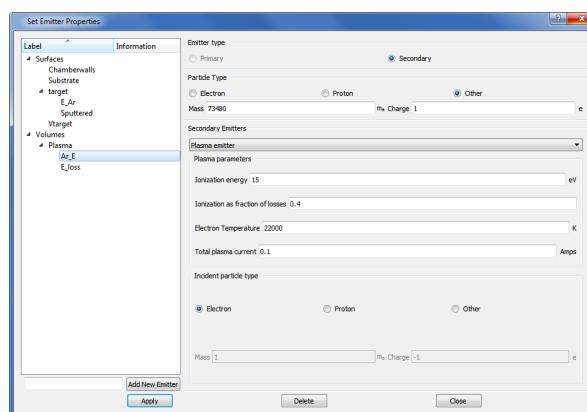


Figure 14.221 Volume emitter *Ar\_E*

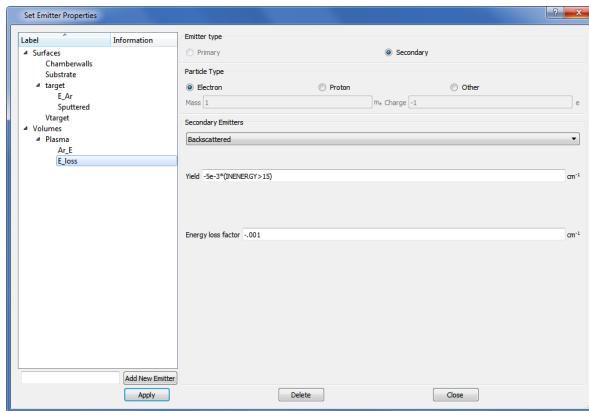


Figure 14.222 Volume emitter  $E_{loss}$

A simple approach has been taken in the definition of the emitters for this example. Additional sophistication may be incorporated by using functional emitter parameters, and by the addition of further emitters to represent other phenomenon in the plasma.

Note that this model uses only secondary emitters; the Plasma Emitter has the property of being able to self-start, avoiding the need to seed the simulation with an artificial particle distribution.

It is usual to exploit the symmetry of a model to reduce the run time. However, with combined electric and magnetic fields, and the presence of charged particles, additional care must be taken over the selection of symmetry boundary conditions. An appropriate choice in this case is positive rotational symmetry of even degree; here eight-fold is used.

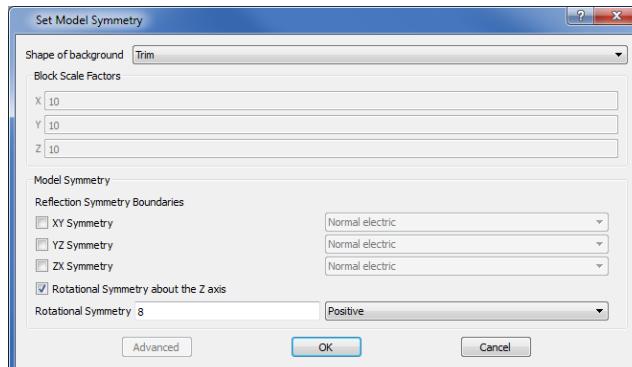


Figure 14.223 Model and field symmetry

As can be seen in Figure 14.224, a simple tetrahedral mesh, without local refinement has been used. However, in general, it might be necessary to refine the mesh close to the expected location of the plasma meniscus, as discussed for the [Plasma Free Surface - Type 103 Emitter \[page 772\]](#).

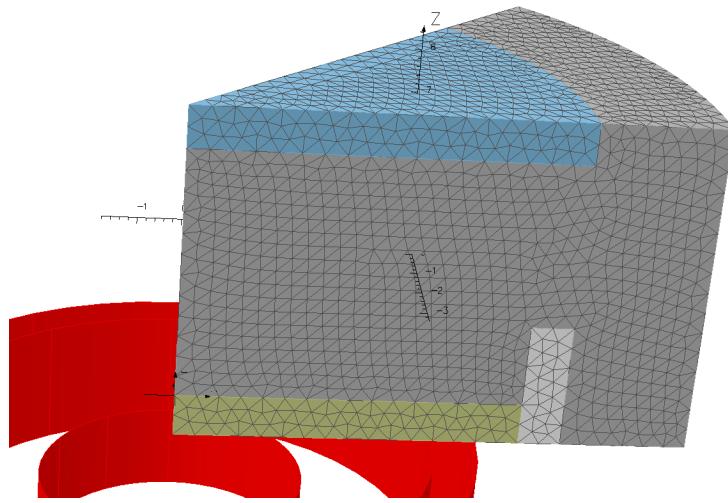


Figure 14.224 Surface mesh

In addition, the use of quadratic elements should be considered where the field gradients are high, as they are around the plasma boundary. Although quadratic elements are computationally expensive, and should only usually be used where required, the relatively small size of this example model allows them to be used throughout the geometry.

## Simulation

Once solved, the standard data extraction tools in the Post-Processor may be used to interrogate the solution. The electron tracks, for example, may be confirmed to show the expected trajectories, as seen in [Figure 14.225](#). In this figure, only those electrons that are produced in the primary symmetry section of the model are shown (i.e. in the 45° wedge). However, these electrons are tracked around the full geometry.

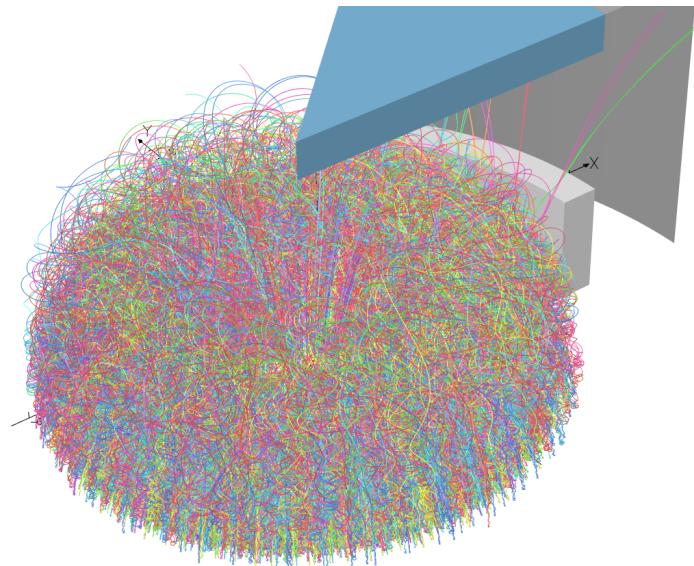


Figure 14.225 Electron tracks in the magnetron

Figure 14.226 and Figure 14.227 show, respectively, the tracks of the sputtered neutrals and the number density of neutrals approaching the face of the substrate - this directly relates to the deposition profile. On the other hand, the number density of the neutrals leaving the target, Figure 14.228, describes the target erosion profile.

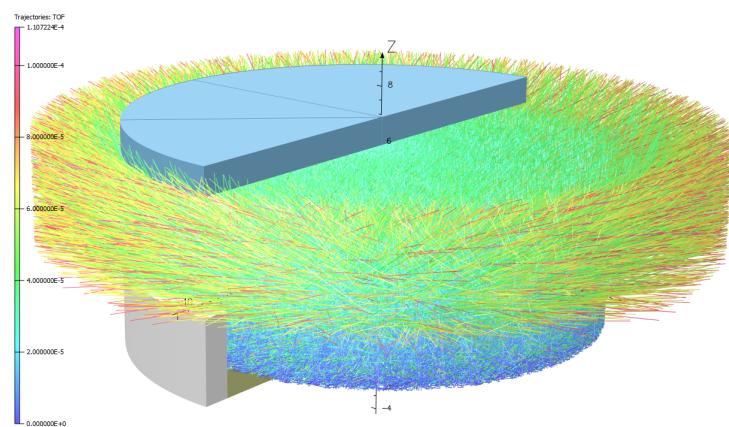


Figure 14.226 Tracks of the sputtered neutrals

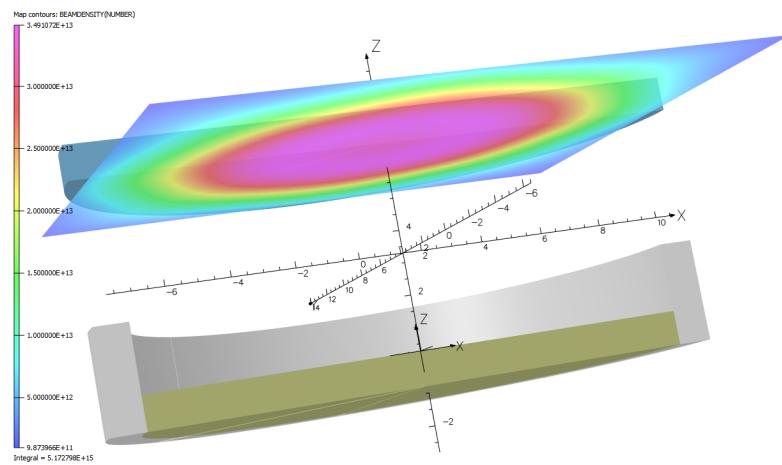


Figure 14.227 Number density of neutrals incident on the substrate

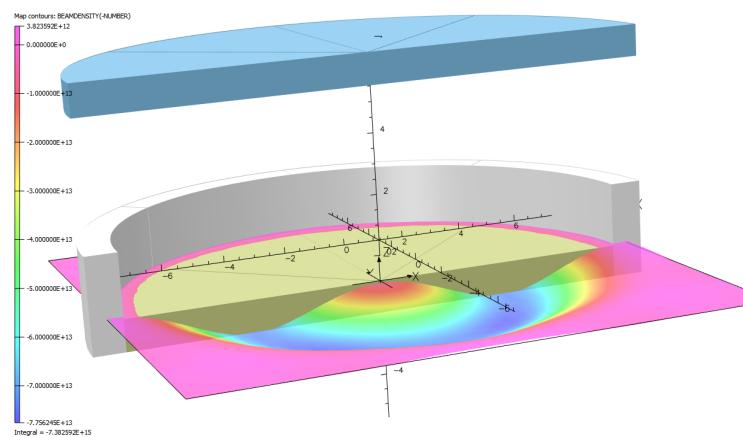


Figure 14.228 Number density of neutrals emitted from the target

## Parameterized Models in the Pre-Processor

### Geometric Modeller

For parameterizing in the Modeller, see [Parameterizing Models and Rebuilding \[page 169\]](#) and the section on "The Optimizer" in the *Opera Manager User Guide*.

### Pre-Processor

1. Create a base "standard" model with no parameterizing in the normal way. Write out the file as a Pre-Processor file, say **base.oppre**.
2. Clear and reset the Pre-Processor. Read the **base.oppre** model into the Pre-Processor.
3. Select **MODIFY**
  - Select **Point coordinates**
  - Set the *plane number* to the plane in which the parameterized coordinates are required.
  - Select **Select/de-select point** and pick the points you wish to move in this plane.
  - Select **Transform points** and choose a method of moving the points.
  - As an example, select **Displace** and enter three values (say) 1.0, 0, 0.
4. Close the menus and write out the new file, say **param.oppre**.
5. Use a system editor to edit the **oppre** file.
  - At the top of the file (in the line above the first command) the user defined constant for the displacement **#xdis** should be set. For example to set **#xdis** to 2.54 enter<sup>1</sup>  
**\$constant #xdis 2.54**
  - Move to the bottom of the file and then move back up to the line  
**MODIFY STAR=n | POINT**  
where **n** is the plane you have chosen<sup>2</sup>.
  - Move down to the **DISPLACE** command and edit the line to include the parameter for the x displacement:  
**DISPLACE #xdis 0 0**
  - Save the **oppre** file. By editing the **#xdis** value you may create any offset required each time the **oppre** file is read.

An even more powerful way of parameterizing a Pre-Processor model is to use the **Label** facilities under **MODIFY**. This allows the user to label points, lines, faces and volumes in the model. A hierarchy exists such that points inherit labels from the lines joining them, lines inherit labels from the

<sup>1</sup>Multiple parameters may be defined in the same file. Using the **\$ASK** command (rather than the **\$CONS** command) will make the **oppre** file prompt the user for a value of the user constant each time the file is read.

<sup>2</sup>Using a do-loop in the **oppre** file will allow the parameterized values to be used in many planes automatically.

faces to which they form edges and faces inherit labels from the volumes that they form. So it is possible for a point to inherit labels from all the volumes, faces and lines to which it belongs as well as have labels applied to it individually.

Having constructed and saved a base model which includes all the point labels that are needed for the parameterizing, the user can then create parameterized models using the **TRANSFORM** command in a similar way to the above example. For example, the top of the file could contain some lines such as:

```
$constant #xfac 1.05  
$constant #yfac 1.0  
$constant #zfac 0.95
```

and the points that are to be parameterized given labels **XLAB**, **YLAB** and **ZLAB** depending on whether the scaling to be applied is in the x, y or z direction respectively.

The parameterizing would then include statements such as:

```
TRANSFORM XLAB x*#xfac 1 1  
TRANSFORM YLAB 1 y*#yfac 1  
TRANSFORM ZLAB 1 1 z*#zfac
```

to enlarge the parameterized object by 5% in the x-direction and reduce it by 5% in the z-direction.

## Single Phase Inductor, featuring 3 types of losses

This 3d application note shows some of the features in transient multiphysics Opera simulations. The model is setup as a coupled **Transient Electromagnetic / Static Thermal** problem. The transient simulation is run until steady-state operation has been detected; then average results are produced for iron loss in the yoke, copper loss in the winding and eddy current loss in a conducting plate.

Given is a single phase racetrack conductor and an iron yoke to return the flux, sitting on a plate made of stainless steel. Exploiting symmetry, one quarter of the configuration is meshed as shown in Figure 14.229.

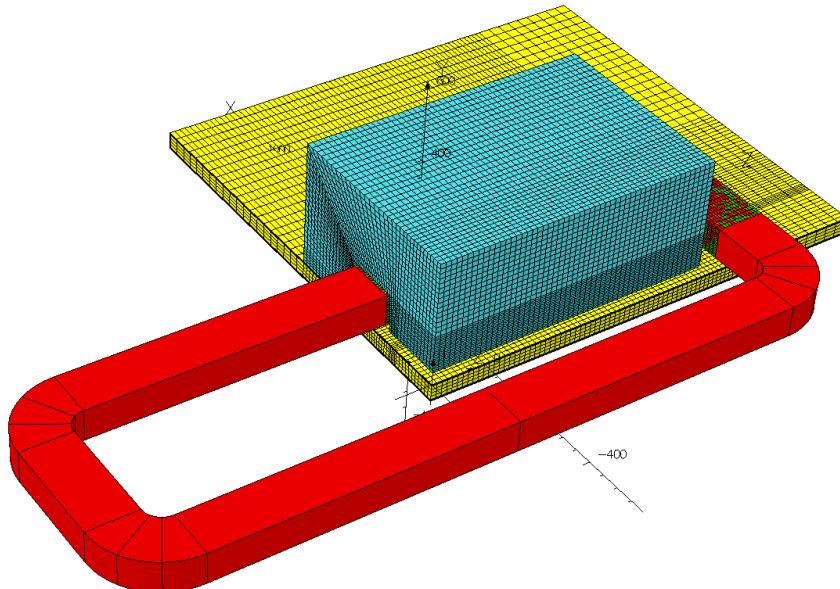


Figure 14.229 Single Phase Inductor above a stainless steel plate

Using this model, we want to calculate 3 types of losses:

- Eddy current losses in the stainless steel,
- Ohm losses in the racetrack conductor, and
- Iron losses in the return yoke.

The racetrack conductor is driven by an AC voltage of 240 Volt (cosine, 50 Hz); it is switched on at 2 ms ( $\text{TTIME}>0.002$ ). Alternatively the switch could be removed, and the peak voltage could be defined as:  $240*\text{RANGE}(\text{TTIME};0.002;333)$ .

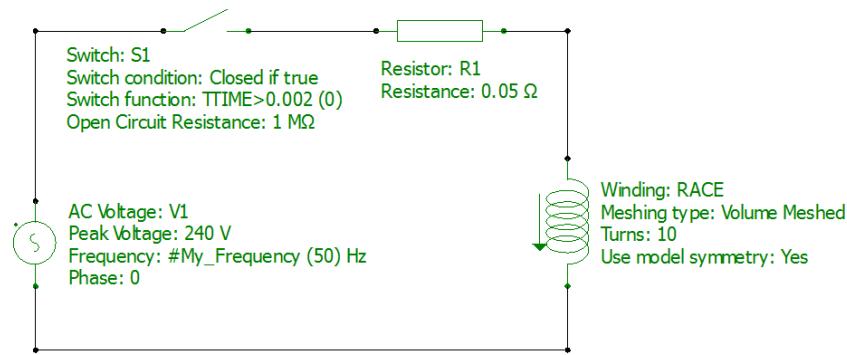


Figure 14.230 Circuit that drives the racetrack conductor

The model should take a few cycles to reach steady-state performance over a cycle. A Python script is used to detect steady-state; when this is reached, the Joule heat density in the stainless steel is calculated over one electrical cycle. The averaged Joule heat density is then used to drive a simple thermal steady-state model that only contains the stainless steel plate.

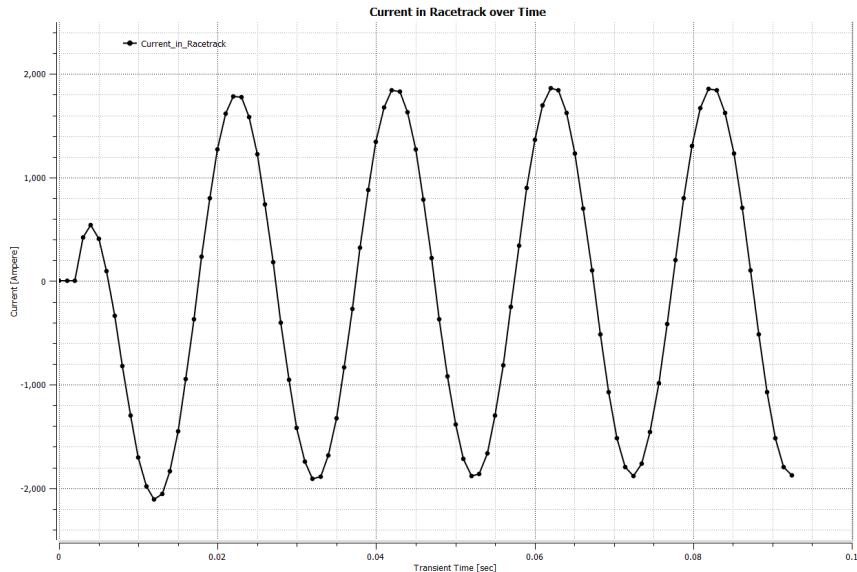


Figure 14.231 Current through the racetrack conductor as a function of time

## Thermal model

Although the thermal model only requires the stainless steel plate, the rest of the mesh is still present but with thermal properties of air. It is therefore ignored by the thermal solver.

A heat transfer coefficient of 3.0E-04 W/(mm<sup>2</sup>K) is applied to the outer surfaces of the plate, apart from the face where the iron yoke touches. The ambient temperature outside the plate is 20 °C. The

yoke is assumed to be watercooled and kept at a constant temperature of 30 °C. A heat transfer coefficient between the stainless steel plate and the yoke is assumed to be 3.5E-02 W/(mm<sup>2</sup>K).

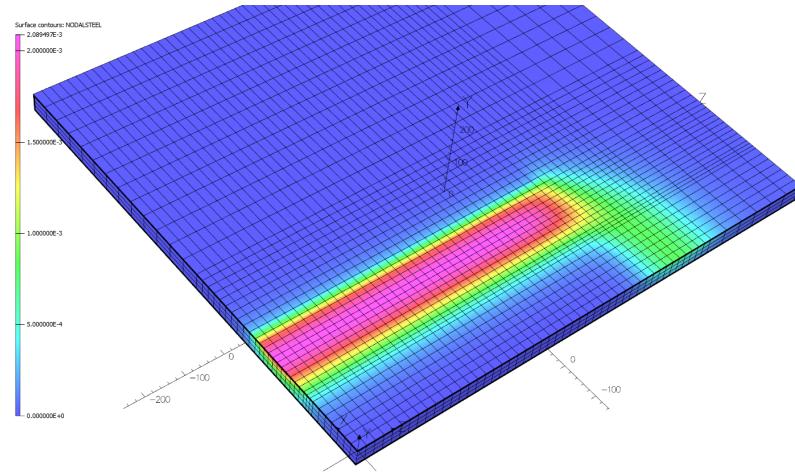


Figure 14.232 Time averaged Eddy Current Losses that are used to drive the thermal mode

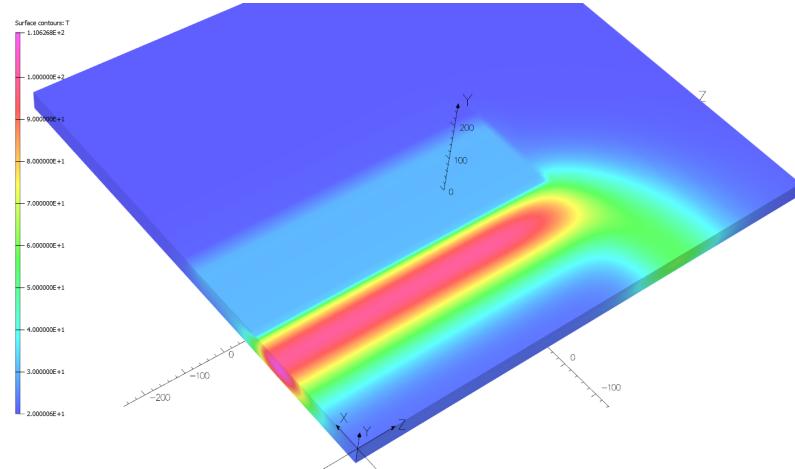


Figure 14.233 Temperature distribution in the thermal model

## Files provided - How to run the model

Copy the following files into a fresh directory

Make_Transient_Racetrack_Inductor_HexMesh.comi	This script prepares an unsolved database file: Racetrack_Inductor_MULTIPHYSICS.
Circuit_ELEKTRA_TR.vfc	Needed for the main script
Cross_2d_Extrusion1.op2	Needed for the main script
Cross_2d_Extrusion2.op2	Needed for the main script
Cross_2d_Extrusion3.op2	Needed for the main script
control_elektratr.comi	The Transient EM solver runs this automatically at each timestep when found in the working directory.
initialize_python_classes.py	Python script called by the control file. It sets up the Python functionality for steady-state detection and loss calculation
Make_PostProcessing_Multiphysics.comi	Script to calculate the 3 types of losses, also generate some of the screenshots.

# **Chapter 15**

# **Supplied Python Functionality**

## **Utility Classes**

---

In addition to the `operafea` module (documented in the Reference Manual), several Python packages are supplied, located in the ***Lib / site-packages*** directory of the distribution. These generally include Python class definitions designed to perform various tasks, such as acquiring indices of node and element numbers related to specific property labels, calculation of time averaged losses, and steady-state detection.

### **Use**

The documentation supplied herein is intended to allow the use and extension of the distributed Python modules. The Python modules are supplied as examples of how to use Opera's Python functionality and may evolve as Opera develops. As such there is no guarantee that scripts utilising these modules will not need to be updated for use with future versions of Opera.

Some examples in this document make use of the modules for performing the functionality listed above:

- Single Phase Inductor, featuring 3 types of losses [page 787]
- Further Work II - Extracting Field Data During the Simulation [page 219]

### **Contents**

To see the contents of a package, the `help` command may be used after it has been imported. For example, the following snippet will print any introductory documentation included in the `OperaDatabaseIndex` package, and list all associated modules:

```
import OperaDatabaseIndex as odi
help(odi)
```

To import the functionality of a package, it is recommended to use the "`from ... import ... as ...`" syntax. For example, a `multi_index` object may be created as follows:

```
from OperaDatabaseIndex import indexclasses as odic
multi_index= odic.OperaMultiIndex()
```

The provided Python packages are:

- **OperaDatabaseIndex** – provides functionality to determine the node and element numbers relating to nodes and elements of specific volume, material or user labels.
- **OperaDatabaseExtraction** – provides functionality to store specified components (such as "RBX", "RJHD"), at user defined labels.
- **OperaDatabaseUpdate** – provides functionality for adding arrays to a database in a manner consistent with Opera's naming schemes, for use, for example, in subsequent analyses.
- **OperaLossCalculation** – provides functionality for calculating time averaged losses, as well as a driver module for calculating multiple different types of loss simultaneously.
  - **OperaLossCalculation.ironloss** – provides specific functionality for time averaged iron loss calculations.
- **OperaSteadyStateDetection** – provides functionality for steady-state detection and for subsequently determining sample output times.
  - **OperaSteadyStateDetection.sequentialanalysis** – provides steady-state detection functionality using various methods involving sequential analysis.

## Database Index

### Purpose

The `OperaDatabaseIndex` package contains functionality to determine the node and element numbers associated with nodes and elements in a labelled section of the model. For example, one may determine the node numbers of all nodes inside a labelled ferrite region of a rotor in a Motional EM simulation. Python classes are provided to extract such information from the database and to subsequently store such information in an easily referable manner ("Database Extraction" on page 796).

### Terminology

In this package, the following terminology is used.

<code>index</code>	A <code>NumPy</code> array containing node or element numbers.
<code>index_key</code>	The label of the section in the model for which an index contains information.
<code>index_key_type</code>	The database association of an index vector, indicating whether it is associated to node numbers (NODAL) or element numbers (ELEMENT).
<code>index_keys</code>	A set of labels, each one of which is an <code>index_key</code> .
<code>multi_index</code>	An instance of a Python class containing the functionality to generate, store and subsequently return the index vectors associated to a supplied <code>index_key</code> set.

### Available classes

- `OperaElementDataRetriever`:  
An example of a `multi_index` implementation which provides functionality for determining and subsequently accessing index vectors with nodal and element associations for a set of `index_keys`.
- `OperaElementDataRetriever`:  
An example of a `multi_index` implementation which provides functionality for accessing index vectors on elements on labeled model faces.

### Typical usage: Opera-3d/Solver

To use a `multi_index` object in the Solver, the solver hooks must be attached to the appropriate `multi_index` functions. Typically, indices need only be generated once. However if the sections of

interest in the mesh change between time-steps, for example if the GAP region of a Motional analysis mesh is under investigation, then the indices will need to be regenerated at each time-step.

In a Python module (`indexModSolver.py`):

```
import operafea
from OperaDatabaseIndex import indexclasses as odi

multi_index= odi.OperaElementDataRetriever(
    ["LABEL_1","LABEL_2",...])

def onStoreTimeOutput():
    simu = operafea.currentSimulation()
    if simu.getId() == 1:
        multi_index.generate_indices(simu)

index_el_1 = multi_index.get_index(index_key="LABEL_1",
    index_key_type="ELEMENT")
index_nod_2 = multi_index.get_index(index_key="LABEL_2",
    index_keys_type="NODAL")
```

In a command file for use in the `Solver`, to instantiate a `multi_index` object:

```
$IF %COMPARE(&Timestep_Stage&,INIT)==0
    / initialize multi_index object
    $PYTHON COMMAND=import operafea
    $PYTHON COMMAND=import indexModSolver as im
    / set custom index keys if required.
    $PYTHON COMMAND=im.multi_index.set_index_keys(["LABEL_1",
        "LABEL_2"])
    / attach functions to appropriate python hooks.
    $PYTHON=operafea.registerHookCallback(
        'on_store_time_output',im.onStoreTimeOutput)
$END IF
```

## Typical usage: Opera-3d/Post

In a Python module (`indexModPost.py`):

```
import operafea
from OperaDatabaseIndex import indexclasses as odi
multiIndex = odi.OperaElementDataRetriever(
    ["USER_LABEL_1","USER_LABEL_2",...])

def post_create_multi_index():
    simu = operafea.currentSimulation()
    multi_index.generate_indices(simu)
```

In a command file for the Post-Processor with a simulation loaded, to construct a `multi_index` object:

```
$PYTHON COMMAND=import opeafea
$PYTHON COMMAND=import indexModPost as im
/ set custom index keys if required.
$PYTHON COMMAND=im.multi_index.set_index_keys(["LABEL_1","LABEL_
2"])
/ create the multi_index
$PYTHON COMMAND=im.post_create_multi_index()
```

## Typical usage: general

Once instantiated, a `multi_index` object may be used to access segments of complete arrays extracted from a simulation stored in the database.

In a Python script (accessed via the `$PYTHON` command):

```
import numpy as np

# access the current simulation in python
simu = opeafea.currentSimulation()

# extract an array from the current simulation with the same
# association as multiIndex.
rbx = simu.getNumPyDblArray("RBX")

# get the index related to the index key "LABEL_1"
index = multi_index.get_index("LABEL_1")

# create an array of only the values of rbx corresponding to the
# index.
rbx_label1 = rbx[index-1]

# create an array new_rbx, with non zero entries only at locations
# indexed in multi_index.
rbx_new = np.zeros(len(rbx))
for index_key in multi_index.index_keys:
    index = multi_index.get_index(index_key)
    rbx_new[index-1] = rbx[index-1]*rbx[index-1]
```

## Notes

The node and element numbers returned from the database and stored in a `multi_index` object are 1 based. Python follows the 0-based array convention. As such, you must subtract 1 from an index vector to get the corresponding Python array references. Index vectors should only be used to access database vectors with the same association (i.e. ELEMENT or NODAL).

## Database Extraction

---

### Purpose

The `OperaDatabaseExtraction` package contains functionality to extract a data set from a given simulation and to subsequently store a (possibly maximal) subset of this data in an easily referable manner. The primary purpose of this is to allow user computations to be performed during the solving stage, although it may equally be used in post-processing.

For example, it may be used to extract "`RBX`", "`RBY`" and "`RBZ`" data, and store only the subset of this data relevant to ferrite regions of a `ROTOR/STATOR` model throughout a transient simulation. Such data may then be used once the transient period is complete.

Typically, this module will be used in conjunction with a `multi_index` object from the `OperaDatabaseIndex` package (["Database Index" on page 793](#)).

### Terminology

In this package, the following terminology is used.

<code>component</code>	Name of a vector variable stored in a simulation, e.g. " <code>RBX</code> "
<code>components</code>	List of components, eg. <code>["RBX", "RBY", "RBZ"]</code>
<code>simulation</code>	Python interface to a simulation as returned by <code>operafea.currentSimulation()</code>
<code>multi_index</code>	Object containing a set of index vectors accessible via a set of <code>indexKeys</code> ( <a href="#">"Database Index" on page 793</a> )
<code>OperaComponentExtraction</code>	Class containing functionality to extract a set of component vectors and store subsets of such vectors given a <code>multi_index</code> object for a single simulation. These are created and utilized by <code>data</code> objects.
<code>data</code>	Object containing functionality to extract and store data over a set of sample times.

### Available classes

`OperaComponentStorage`

An example of a sub-class of the `OperaComponentExtraction` type which provides functionality to extract and store data from a set of components for a set of `index_keys` at multiple sample times.

## Typical usage: Opera-3d/Solver

A data storage object may be created during the initialization phase of a solving stage. Typically, the `data` should be updated via the `actionOnTimeStep` function, which should be called inside a function attached to the '`on_store_time_output`' solver hook.

The function `actionOnTimeStep` requires an initialised `multi_index` object to have been constructed. As such, the following example also generates a `multi_index` object.

In a Python module (`storeModSolver.py`):

```
import operafea
from OperaDatabaseIndex import indexclasses as odic
from OperaDatabaseExtraction import extractionclasses as odexc

sample_id=0
multi_index=odi.OperaMultiIndex()
data_storage=odex.OperaComponentStorage()

# updated by some other function
require_sample = False

def on_store_time_output():
    global sample_id
    simu=operafea.currentSimulation()
    if require_sample:
        sample_id+= 1
        ttime = operafea.getSysVar("TTIME")
        sim_id = simu.getId()
        if sample_id == 1:
            multi_index.generate_indices(simu)

    data_storage.action_on_time_step(simu, multi_index,
                                    sample_id = sample_id, sample_time= ttime)
```

In a command file for use in an Opera-3d/Solver:

```
$IF %COMPARE(&Timestep_Stage&,INIT)==0
    / initialize multi index
    $PYTHON COMMAND=import operafea
    $PYTHON COMMAND=import storeModSolver as sm
    / set custom index keys if required.
    $PYTHON COMMAND=sm.multi_index.set_index_keys(["LABEL_1",
    "LABEL_2"])
```

```

$PYTHON COMMAND=sm.data_storage.set_index_keys(["LABEL_1",
    "LABEL_2"])
/ set custom component list
$PYTHON COMMAND=sm.data_storage.set_components(["RBX",
    "RBY","RBZ"])
/ attach functions to appropriate python hooks.
$PYTHON COMMAND=operafea.registerHookCallback(
    'on_store_time_output',sm.on_store_time_output())
$END IF

```

## Typical usage: Opera-3d/Post

Given an existing `multi_index` object, a `data` storage object may be generated as required. The following example assumes a database is already loaded, and a `multi_index` already present with the required index keys.

In a command file for use in Opera-3d/Post, or from the console:

```

$PYTHON COMMAND=
    from OperaDatabaseIndex import extractionclasses as odex
$PYTHON COMMAND=
    data_storage=odex.OperaComponentStorage(
        components=["RBX","RBY","RBZ"],
        index_keys=multi_index.index_keys)

/ loop through simulations storing required data
$DO #CASE 1 10
    SIMULATION CASE=#CASE
    / construct the interface to the current simulation
    $PYTHON COMMAND=simu=operafea.currentSimulation()
    / store the required data.
    $PYTHON COMMAND=data_storage.actionOnTimeSTep(
        simu, multi_index, sample_time= operafea.getSysVar
        ("TTIME"))
$END DO

```

Once data is stored it may be easily accessed.

```

$PYTHON COMMAND='rbx_10_label1 = data_storage.get_component(
simu.getId(), "LABEL_1", "RBX")'

```

## Database Update

### Purpose

The `OperaDatabaseUpdate` package provides a structure for adding `NumPy` double and complex arrays into an Opera database in a manner consistent with the internal naming scheme.

### Terminology

<code>db_description</code>	Descriptive string for an array, for example " <code>HEAT</code> "
<code>db_type</code>	Descriptive string for the type of array to upload to the database, for example " <code>REAL</code> " , " <code>COMPLEX</code> " , " <code>IMAGINARY</code> "
<code>db_association_type</code>	Descriptive string for the association of the array to upload, typically " <code>NODAL</code> " or " <code>ELEMENT</code> "
<code>data</code>	<code>NumPy</code> array to be uploaded to the database
<code>add_to_database</code>	Method performing all necessary adjustment of a <code>dbaseVector</code> in order to upload it to an Opera database

An `OperaDatabaseUpdate` instance takes a `db_description`, a `db_type`, and a `data` `NumPy` array, and adds the real and/or imaginary components of the `data` to a simulation with the association `db_association_type` and the names, `[R, I] [ASSOC] [DESC]`, depending on whether real and/or imaginary components of the `dbaseVector` are required, and whether the association is to be included in the name.

For example, a real valued "`HEAT`" vector corresponding to `NODAL` values would be added as `RNODALHEAT`.

The `OperaDatabaseUpdate` class also automatically accounts for cases where the supplied `data` is stored in a (potentially) non-contiguous manner, as is the default for `NumPy` arrays. A units expression for the dimensions of the new vector may also be included.

### Available classes

#### `OperaDatabaseUpdate`

A class providing functionality for adding 1d `NumPy` arrays to a simulation in a manner consistent with Opera. This class is contained within the `OperaDatabaseUpdate.databaseupdateclasses` module.

## Typical usage: Opera-3d/Solver

Any valid simulation may have data added to it using a `OperaDatabaseUpdate` instance. Valid simulations will be available during all time-step stages after "`INIT`", however the update would typically occur during the "`END`" or "`OUTPUT`" stages, corresponding to the Python hooks "`on_timestep_end`" and "`on_store_time_output`". For more information on time-step control and Python hooks in the 3d Solvers see the Opera 3d Reference Manual.

In a Python module (`updatePyMod.py`):

```
import operafea
import OperaDatabaseUpdate as oduc
...
database_update=oduc.OperaDatabaseUpdate(
    db_description="HEAT",
    db_type="REAL",
    db_association_type="NODAL")

def on_store_time_output():
    simu = operafea.currentSimulation()
    my_data = np.zeros( nodes )
    my_data[...] = ...

    database_update.add_to_database(my_data,
        simu,
        db_units_expression="POWEU/LENGU**3")

operafea.registerHookCallback(
    "on_store_time_output", on_store_time_output)
```

In the initialisation command file (`solver.com`):

```
$PYTHON COMMAND=import updatePyMod as upm
```

## Notes

The `data` supplied to the function `add_to_database` must have the correct length given the association. For example, if the association is "`ELEMENT`", the length of the array must equal the number of elements in the current simulation's mesh.

## Steady-State Detection

---

### Purpose

The `OperaSteadyStateDetection` package provides functionality to determine when a transient simulation has reached steady-state. That is when the behaviour of the system becomes approximately time periodic. In the top level package, a default detection method acting upon an assigned transient time is provided (i.e. steady-state is reached at a fixed time).

More advanced functionality is provided in the sequential analysis (`sequentialanalysis`) sub-package, which provides two detection methods based on the sequential analysis of different characteristics of the simulation data. In all cases, the methods provide a function to return a testable flag to indicate when steady-state has been detected.

In addition to the steady-state detection mechanisms, functionality is provided for the automated calculation and activation of a set of sample times. Typically this will be enacted upon the detection of steady-state.

The module is intended to be used primarily in the Solver, although it may equally be used in the Post-Processor. The default behaviour of the classes can be altered by setting the named parameters listed below. For an example of this "[Typical usage: Opera-3d/Solver](#)" on page 803.

### Methods: Variance Test

In the first method, also called the "variance test" method, sequential analysis through parameter estimation is performed on collected simulation data. Once the required criteria for steady-state detection has been satisfied, further sampling is disabled. The stopping criterion is based on the minimisation of a cost function that measures the data heteroscedasticity, the tendency of the standard deviation of the observed quantity to vary:

$$t_{\text{steady state}} = \arg \min_{0 \leq t \leq \tau} G(t) \quad , \quad (15.1)$$

where  $G(t) := \text{Var}_{0,t} + \text{Var}_{t,\tau}$ ,  $\text{Var}_{t,t}$  is the variance of the observed quantity computed from a sample collected between times  $t$  and  $t'$ , and  $\tau$  represents either the final time or a reasonable guess sufficient to observe statistically significant results.

### Methods: Mean Test

The second sequential analysis detection method, referred to as the "mean test", attempts to detect the transient time at which the mean of the data settles down to zero. It may be used, for example, to analyse the DC offset in a transient electromagnetic simulation. The detection uses a version of the CUSUM algorithm, in which the following hypotheses are tested:

$$H_0 : \theta = \theta_0, \quad (15.2)$$

$$H_1 : \theta = \theta_1, \quad (15.3)$$

where  $\theta$  represents a set of parameters appropriate for the data probability distribution,  $H_0$  is the hypothesis corresponding to transient state and  $H_1$  is the hypothesis that steady-state has been achieved. Parameter values  $\theta_0$  and  $\theta_1$  correspond to transient and steady-states, respectively. The log-likelihood function for the sample is computed and compared to a threshold value to decide which hypothesis to accept.

In transient electromagnetic simulations, the data (e.g. current, voltage) is periodic and the assumption is that the sample follows an arcsine distribution in which the amplitude and the offset are the control parameters. The nominal value of these parameters define the steady-state and need to be specified along with the threshold for the log-likelihood function.

The algorithm, referred to as CUSUM2, continues until the sample parameters reach a steady-state. A slightly different algorithm, referred to as CUSUM, from the same family but using a normal distribution and different decision technique is added with the intention that it be used when the simulation output is dominated by noise and one is interested in detecting changes of the sample mean.

## Parameters

<code>nb_training_points</code>	Number of data points needed for initial "training". Default value: 10 for the variance test, 30 for CUSUM (not applicable for CUSUM2).
<code>nb_points</code>	Number of data points collected before updating. Default value: 10 for the variance test, -1 for CUSUM (not applicable for CUSUM2).
<code>convergence_threshold</code>	Criterion threshold. Default value: 0.87 for the variance test, 5e+7 for CUSUM (5.2 for CUSUM2).
<code>signal_name</code>	String identifying the variable to be used for the signal. Default value: "RO_TORQUEZ" for the variance test, "RACE_I" for CUSUM and CUSUM2.
<code>nominal_mean</code>	Nominal value of the mean (used only in CUSUM). Default value:0.
<code>sigma</code>	Standard deviation (used only in CUSUM). Default value: -1.
<code>signal_period</code>	Period of the sample (used only in CUSUM2).
<code>signal_ac_amplitude</code>	Amplitude of the data corresponding to steady-state (used only in CUSUM2).

<code>signal_dc_offset</code>	Offset of the data corresponding to steady-state (used only in CUSUM2).
<code>nb_bins</code>	Parameter controlling the approximation to the probability distribution (used only in CUSUM2). Default value: 80.
<code>steady_state_time</code>	Time at which steady-state is reached.

## Available classes

- **`SteadyStateDetection`:**  
Base steady-state detection class. This is primarily used to provide interfaces to common functionality to more advanced steady state detection methods, although it may also be used to enforce steady-state detection at pre-determined times or simulation case numbers.
- **`SequentialAnalysis`:**  
Class that is derived from the `SteadyStateDetection` and is used as a base class for sequential analysis. Classes derived from this class must implement algorithms to detect the occurrence of steady-state.
- **`SequentialAnalysis.VarianceEstimation`:**  
Class implementing sequential analysis through parameter estimation to detect a change in the variance of the data, under the assumption that the mean does not change. This class is suitable for the detection of steady-state in variables related to rotating machines, such as torque and speed.
- **`SequentialAnalysis.CumulativeSum`:**  
Class implementing sequential analysis through hypothesis testing to detect change in the sample mean.
- **`SequentialAnalysis.CumulativeSumWithDCOffset`:**  
Class implementing sequential analysis through hypothesis testing. This class is intended to be used for detecting steady-state in transient electromagnetic simulations when the data is periodic (e.g. detecting steady-state for the DC offset of the electric current).
- **`OperaSampleInfo`:**  
Provides functionality to determine required sample times and to subsequently ensure that such times are output times. A sample info class is often used in conjunction with a steady-state detection class.

## Typical usage: Opera-3d/Solver

A `steady_state_detector` should typically be acted upon during functions associated with the `on_timestep_end` function hook. Consequently, `sample_info` objects should perform their sample generation during functions associated with the `on_timestep_end` hook, and should subsequently be included in functions hooked to "`on_store_time_output`" to guide the future sampling process.

In a Python module, `ssdMod.py`,

```

import operafea
from OperaSteadyStateDetection import sampleinfoclasses as sic
from OperaSteadyStateDetection.sequentialanalysis \
    import seqanalysisclasses as sassdc
...
global steady_state_detector
global sample_info
...
steady_state_detector=sassdc.VarianceEstimation(
    signal_name="RO_TORQUEZ",
    nb_points=10,
    convergence_threshold=0.87,
    user_ss_time= 2.0)
sample_info=sic.OperaSampleInfo(
    cycle_length=0.2,
    samples_per_cycle=20,
    cycles_to_sample=2)

def on_time_step_end():
    global steady_state_detector
    global sample_info
    ...
    simu = operafea.currentSimulation()
    ttime = operafea.getSysVar("TTIME")
    calculate_sample_flag=False
    if not steady_state_detector.at_steady_state:
        steady_state_detector.action_on_timestep_end(simu,ttime)
        calculate_sample_flag=steady_state_detector.at_steady_
        state

    sample_status= sample_info.action_on_timestep_end(
        ttime,simu,calculateSampleFlag)
    ...

def on_store_time_output():
    global sample_info
    ...
    ttime=operafea.getSysVar("TTIME")
    sample_status=sample_info.get_sample_status(ttime)
    if sample_status in [2,4,5,7,8]:
        # current time is a required sample time.
        ...
        if sample_status in [5,7,8]:
            # at least one complete cycle has occurred.
            ...

```

In a comi initialization file, **solver.comi**,

```
$PYTHON COMMAND=import ssdMod as ssdm
$PYTHON COMMAND=operafea.registerHookCallback(
    "on_timestep_end", ssdm.on_time_step_end)
$PYTHON COMMAND=operafea.registerHookCallback(
    "on_store_time_output", ssdm.on_store_time_output)
```

## Debugging

The verbosity option can be set on the steady-state detection class to print out diagnostic information into the res file on each time-step:

```
steady_state_detector.verbosity=True
```

## Loss Calculation

---

### Purpose

The `OperaLossCalculation` package provides functionality to calculate time averaged losses when provided with historic data and to subsequently upload such losses to an Opera database.

Losses are only calculated for specified subsections of the complete model, identified through a `multi_index` object (["Database Index" on page 793](#)), and are computed using data stored in a `data store` (["Database Extraction" on page 796](#)).

The top level package contains classes for the computation of time averaged losses based upon averaging Joule Heat Density, and **J.E** over a time period. The `ironloss` sub-package provides three further methods for calculating losses based on Steinmetz iron loss formulae using Fourier analysis of the B field. In addition, support classes are provided for defining parameter sets for specific volumes and materials, and for accounting for rotation in rotating machine models.

A driver module is also supplied, providing an example of a complete loss calculation implementation. This allows for multiple different loss calculation methods to be used simultaneously, and uses functionality from the Opera steady-state detection package (["Steady-State Detection" on page 801](#)) to determine sampling information once a model has reached steady-state. The functionality of the corresponding classes to this module may be easily replaced according to requirements.

### Terminology

In this package the following terminology is used:

<code>index_key</code>	String corresponding to a user label in a database, identifying a subsection of the complete model, e.g. " <code>ROTORIRON</code> "
<code>index</code>	Vector containing the node or element numbers of all nodes or elements in a subsection of the complete model, as identified by an <code>index</code> .
<code>component</code>	String corresponding to a database variable required for the loss calculation, e.g. " <code>RBX</code> " , " <code>RJHD</code> " .
<code>components</code>	Collection of all components required to calculate the loss, e.g. <code>["RBX", "RBY", "RBZ"]</code> .
<code>simu</code>	A simulation object as returned by <code>operafea.currentSimulation()</code>

<code>sample_id</code>	Unique identifier for a required sample.
<code>multi_index</code>	An object containing a collection of index arrays referable by a set of index keys.
<code>data</code>	An object containing historic data of a set of components at a set of indices extracted from simulations at previous transient times, referable by a <code>(sample_id, index_key, component)</code> triplet.
<code>loss_parameters</code>	An object allowing the storage of material specific parameters required for the loss calculation.
<code>spatial_compensation</code>	An object allowing for rotation to be accounted for (if required).
<code>loss_method</code>	An object with functionality to calculate vectors of losses for index vectors supplied in a <code>multi_index</code> instance.
<code>database_update</code>	An object with functionality to add a complete loss vector to a database given a <code>loss_method</code> with computed losses and a corresponding <code>multi_index</code> object.
<code>steady_state_detector</code>	An object with functionality to detect the onset and completion of a steady-state cycle.

## Available classes: time averaged losses

The `losscalculationclasses` module contains the following classes of interest, containing functionality for calculating time averaged losses:

- `OperaJHDLossCalculator` - time averaged Joule heat Density calculation.
- `OperaJdotELossCalculator` - time averaged J.E calculation.
- `OperaTimeAverageLossCalculator` - time averaged summation of generic database vectors (with the same association).

## Available classes: iron losses

The `ironloss.ironlosscalculationclasses` module contains functionality for computing iron losses using Steinmetz formulae.

- `OperaIronLossSteinmetz2Terms` - 2 term Steinmetz iron loss calculation.
- `OperaIronLossSteinmetzQuadratic` - quadratic Steinmetz iron loss calculation.
- `OperaIronLossSteinmetzBertotti` - 3 term Steinmetz-Bertotti iron loss calculation.

## Available classes: loss parameters and movement

The `parameterclasses` module provides functionality for a standardized parameter class, allowing, for example, material data to be passed into a loss calculation instance.

- `OperaLossParameterSet` - default parameter template class.
- `ironloss.ironlossparameterclasses` - contains implementations of parameter classes with all parameters required by the iron loss calculation classes.
  - `OperaIronLossSteinmetz2TermParameters` - parameters for a 2 term Steinmetz iron loss calculation.
  - `OperaIronLossQuadraticParameters` - parameters for a quadratic Steinmetz iron loss calculation.
  - `OperaIronLossSteinmetzBertottiParameters` - parameters for a 3 term Steinmetz-Bertotti iron loss calculation.
- `OperaRotorMovement` - accounts for constant rotation in a Rotor Stator model using Steinmetz Bertotti iron loss calculation classes.

## Available classes: multiple losses

### `multiplelossdrivermodule`

Implementation of a calculation mechanism allowing for multiple simultaneous loss calculation methods, steady-state detection and automatic sample time generation.

## Typical usage: Opera-3d/Solver

The `OperaLossCalculation.multiplelossdrivermodule` can be used to rapidly construct a complete loss calculation mechanism.

In a Python script file, `lossdriver.py`

```
import operafea
from OperaLossCalculation import multiplelossdrivermodule as ldrm
from OperaLossCalculation import losscalculationclases as lcc
from OperaLossCalculation import spatialcompensationclases as sc
from OperaLossCalculation.ironloss \
    import ironlosscalculationclases as ilcc
from OperaLossCalculation.ironloss \
    import ironlossparameterclases as ilpc
from OperaSteadyStateDetection.sequentialanalysis \
    import seqanalysisclasses as sassdc

#Set labels for loss classes.
loss_labels= ["IRON","ALUMINIUM","CONDUCTOR",...]
ldrm.loss_manager.set_loss_labels(loss_labels)

#For each loss label, define:
```

```
# a loss method
# (derived from the class OperaLossCalculator
# contained in OperaLossCalculation.losscalculationclasses)
# a loss parameters object (if required by the loss_method)
# (derived from the class OperaLossParameterSet
# contained in the module OperaLossCalculation.parameterclasses)
# a spatial compensation method (if required by the model)
# (derived from OperaSpatialCompensation)

# First loss class, corresponding to "IRON"
iron_loss_method=ilcc.OperaIronLossSteinmetz2Terms(["IRON_LABEL_1",
    "IRON_LABEL_2", "IRON_LABEL_3"])
iron_loss_parameters=ilpc.OperaIronLossSteinmetz2TermParameters()

# set default parameters
iron_loss_parameters.add_parameters(set_number=0,
    parameters={"kh" : kh_0, "ke" : ke_0,
        "frequency": operafea.getUsrVar("#FREQ")})

# set parameters specific to a subset of index keys.
iron_loss_parameters.index_key_to_set_relationship["IRON_LABEL_1"]
= 1
iron_loss_parameters.add_parameters(set_number=1,{"kh":kh_0,...})

iron_loss_rotation = sc.OperaRotorMovement()

# update the loss manager
ldrm.loss_manager.set_loss_method("IRON", iron_loss_method)
ldrm.loss_manager.set_loss_parameters("IRON",iron_loss_parameters)
ldrm.loss_manager.set_loss_spatial_compensation(
    "IRON", iron_loss_rotation)

# Second loss class, corresponding to "ALUMINIMUM"
al_loss_method=lcc.OperaJdotELossCalculator(["AL_LABEL_1","AL_
LABEL_2"])
ldrm.loss_manager.set_loss_method("ALUMINIUM", al_loss_method)

# Third loss class, corresponding to "CONDUCTOR"
conductor_loss_method= lcc.OperaJHDLossCalculator(["COND_LABEL_1",
"COND_LABEL_2"])
ldrm.loss_manager.set_loss_method("CONDUCTOR", conductor_loss_
method)

# Either set a pre-defined steady-state time,
# or replace steadyStateDetector
ldrm.steady_state_instance.steady_state_time = ...
# or
```

```

ldrm.steady_state_instance= \
    sassdc.VarianceEstimation(...)

...
# Set information for the sample over a steady-state cycle
ldrm.sample_info.set_cycle_length(...)
ldrm.sample_info.set_samples_per_cycle(...)
ldrm.sample_info.set_cycles_to_sample(...)
...

# Attach to appropriate python hooks
operafea.registerHookCallback('on_store_time_output',
    ldrm.onStoreTimeOutput)
operafea.registerHookCallback('on_timestep_end',
    ldrm.onTimestepEnd)

```

In a command file;

```

$IF %COMPARE(&TIMESTEP_STAGE&,INIT)==0
    $PYTHON file=lossdriver.py
$END IF

```

## Typical usage: Opera-3d/Post

The files *loss\_calculations\_in\_POST.comi* and *initialize\_python\_classes.py* can be used to evaluate iron losses and copper losses on a solved database in the Post-Processor. These files can be found in the **Examples Folder (Opera Manager: File -> Open examples folder-> 3D/Loss\_evaluation\_Python)**.

The python file *initialize\_python\_classes.py* is an adapted version of the *lossdriver.py* as described above. The comi file *loss\_calculations\_in\_POST.comi* provides an interface for:

- selecting type of loss (iron/copper) to be evaluated;
- selecting different parts (using material labels) of the model over which losses are to be calculated;
- entering loss coefficients and other parameters to evaluate losses;
- registering hooks and calling appropriate loss classes; and
- displaying results (total losses).

Before running the comi file, users need to make sure that:

- both the comi and python files are located in the same folder;
- a solved database is loaded in the Post-Processor; and
- a set of equally spaced output times within a steady-state cycle are available in the solved database.

The following is an excerpt from the comi file that performs the loss evaluation using loss classes defined in Python

```
$PYTHON FILE=initialize_python_classes.py
```

```
$PYTHON command=import numpy as np
SIMULATION CASE=#start_case
$PYTHON command=ldrm.loss_manager._actionOnStoreTimeOutputCount = 0
$PYTHON command=ldrm.sample_info.reset()
$PYTHON command=ldrm.sample_info.set_cycle_length(operafea.getUsrVar
("#cycle_length") )
$PYTHON command=ldrm.sample_info.set_samples_per_cycle(operafea.-.
getUsrVar("#samples"))
$PYTHON command=sample_time=ldrm.sample_info.calculate_sample_times
(operafea.getSysVar("TTime"))
$PYTHON COMMAND=ldrm.sample_info.set_sample_times(sampleTime)
$STRING TSTEP_FINAL NO
$DO #CASE #start_case CASES
  SIMULATION CASE=#CASE
  $PYTHON command=ldrm.onStoreTimeOutput()
  //
  $IF %COMPARE(&TSTEP_FINAL&,YES)==0
    $EXIT
  $END IF
$END DO
```

# Index

---

## A

accelerator 382  
anisotropy 160, 302, 570, 574

---

## B

baseplane 477  
BEAMPASS 376  
bend 139  
BH curve 37, 55, 570, 574, 576, 583, 596  
Biot-Savart conductors  
    example 155  
blend 129, 383  
block 79  
body 76-77  
boundary conditions 175, 206, 273, 315, 360, 383, 499, 557, 694, 726

---

## C

CAD files 172

CARMEN 29, 167, 684  
CATIA 172-173  
cavity 382  
cell 76-77  
chamfer 129  
charge density 160  
Circuit Editor 319, 652-653  
circuits 304, 651, 684  
conductors 76, 506  
    meshed 157  
Console 38, 65  
construction lines 478-481  
context menus 102  
coupled analysis 29, 329, 565, 758  
current flow 28  
cutaway 104  
cutting sheet 554  
cylinder 81

---

## D

data files  
    analysis database 61

command scripts 715  
 modeller 61  
 Pre-Processor 534  
 database 60  
 DEMAG 576, 581, 651  
 demagnetization 578, 581  
 display 40, 502, 520

---

**E**

Eddy current NDT 603  
 edge 75  
 eigenvalue 389  
 Electric insulator boundary condition 604  
 electrostatics 28-29, 160, 175, 753  
 ELEKTRA 164, 476, 536, 576, 596, 758  
 element size 42, 545  
 element type 41, 166  
 emitters 343, 361  
 entity 76  
 external circuits  
     see circuits 304  
 extrusions 489

---

**F**

face 75  
 facets 484

---

filamentary conductors 156  
 force 69, 565, 726

---

**G**

grain-oriented material 571

---

**H**

heat 28-30, 160  
 hide 94  
 hysteresis 572, 596

---

**I**

IGES 172-173  
 inductance 301, 317, 641, 657  
 intersection 104  
 Iron Loss 717

---

**L**

layering 183, 273  
 Legendre polynomials 37, 645  
 local coordinate system 76, 143  
 logging 693  
 lossy dielectrics 29, 344, 726

---

**M**

magnetization 28, 42, 58, 581, 596, 646  
Magnetization 28  
Magnetostatic, Electrostatic and Current Flow 28  
magnetostatics 28, 753  
material properties 491, 530  
mesh  
    control 147  
    errors in Modeller 550  
    surface mesh 541  
    volume mesh 542  
meshed conductors 157  
model body 52, 164, 541, 695  
model dimension 171  
model symmetry 50, 66, 160, 205, 285, 308, 317, 697  
morph 140  
Mosaic meshing 364  
Motional Analysis 28  
mouse buttons 40

---

**O**

offset 127  
Opera-2d 172, 696  
Opera Analysis block 713

Opera Manager 37  
orientation 160

---

**P**

pan display 40  
periodicity 557, 684  
permanent magnet 160  
pick 94  
points 477, 480  
cartesian coordinates 482  
construction line intersections 480  
potentials 491-492, 495, 529  
Pre-Processor data files 517  
Pro-E 172

---

**Q**

Q-factor 398  
quench 211, 301  
QUENCH 28, 30, 301, 576, 762

---

**R**

regularization 104  
replay 169  
representative filaments 156

resonant cavity 382  
 rotate display 40  
 rotating machines 28, 557, 684

**S**

SAT 172-173  
 SCALA 30, 357, 754  
 secondary emitters 376  
 Segmented magnets 603  
 sheet face 76, 78  
 shell 127  
 simulations 529  
 Simulink 713  
 sketching 79  
 SOPRANO 382, 755  
 Space Charge 28-29  
 sphere 83  
 Static and Transient Thermal 29, 273  
 Steady State and Modal HF 29  
 Steady State EM 28  
 STEP 172-173  
 stretch 138  
 subdivisions 487, 490  
 subtraction 104  
 sweeping 117

**T**

table files 567, 576, 581, 733, 753  
 TEMPO 30, 576, 758  
 time tables 207  
 torus 84  
 TOSCA 30, 175, 476, 565, 570, 581, 641  
 transformations 113  
 trim 104  
 twist 136

**U**

union 104  
 units 36, 57, 61, 166, 200, 273, 579

**V**

velocity 160  
 vertex 75  
 volume data 160

**W**

wire edge 75, 89  
 working coordinate system 77, 144-145

---

**Z**

zoom display 40