Abstract
This paper presents a first modeling guide for the modeling and simulation of up to full 3D dynamic Heat, Air & Moisture (HAM) transport of building constructions using COMSOL with Matlab. The modeling scripts are provided at the appendix. Furthermore, all modeling files and results are published at the following webpage: http://sts.bwk.tue.nl/hamlab/
The aim of the paper is to provide a relative easy access to HAM modeling and simulation for people who have only minor experience with MatLab and COMSOL (release 3.2). Another aim is to increase the user group of COMSOL so that possibly more and more interesting models and results become available for the building physics society. Comments of users will be used to improve this document in future versions.

0. INTRODUCTION
The driving force for this work is the International Energy Agency Annex 41 project. The aim of the Annex 41 project is to study the effects of whole building HAM response on comfort, energy consumption and enclosure durability. The sample construction in this paper is related with the Lightweight construction of the Subtask 1, Common Exercise 1 'BESTEST REVISED'. The goal of this exercise is an extension of the BESTEST Case 600 with the focus on moisture conditions in the building.
This paper provides a modeling guide for the modeling and simulation of up to full 3D dynamic Heat, Air & Moisture (HAM) transport of a roof/wall construction using the material properties and boundary values of the above described Annex 41 exercise. A short description of the material properties and boundary values is presented below.
Material properties and boundary values

The material specifications are presented in table I.

Table I. Material specifications

<table>
<thead>
<tr>
<th></th>
<th>Dry λ (W/mK)</th>
<th>Thickness (m)</th>
<th>U (W/m²K)</th>
<th>R (m²K/W)</th>
<th>Dry Density (kg/m³)</th>
<th>Dry C_(p) (J/kgK)</th>
<th>Perm (kg/msPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exterior wall</strong> (inside to outside)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wood panels</td>
<td>0.160</td>
<td>0.012</td>
<td>13.333</td>
<td>0.075</td>
<td>395^</td>
<td>1880</td>
<td>1e-9</td>
</tr>
<tr>
<td>Cellulose ins.</td>
<td>0.040</td>
<td>0.066</td>
<td>0.606</td>
<td>1.650</td>
<td>55.0^</td>
<td>1880^</td>
<td>5.5e-5</td>
</tr>
<tr>
<td>Wood siding</td>
<td>0.140</td>
<td>0.009</td>
<td>15.556</td>
<td>0.064</td>
<td>530</td>
<td>900</td>
<td>1e-9</td>
</tr>
<tr>
<td>Ext. surf. coeff</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total air-air</td>
<td>0.514</td>
<td></td>
<td></td>
<td>1.944</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total surf.-surf.</td>
<td>0.559</td>
<td></td>
<td></td>
<td>1.789</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Roof</strong> (inside to outside)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Int. surf coeff</td>
<td>8.29</td>
<td></td>
<td>0.121</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wood panels</td>
<td>0.160</td>
<td>0.010</td>
<td>16.000</td>
<td>0.063</td>
<td>395^</td>
<td>1880</td>
<td>1e-9</td>
</tr>
<tr>
<td>Cellulose ins.</td>
<td>0.040</td>
<td>0.1118</td>
<td>0.358</td>
<td>2.794</td>
<td>55.0</td>
<td>1880^</td>
<td>5.5e-5</td>
</tr>
<tr>
<td>Roof deck</td>
<td>0.140</td>
<td>0.019</td>
<td>7.368</td>
<td>0.136</td>
<td>530</td>
<td>1880^</td>
<td>1e-9</td>
</tr>
<tr>
<td>Ext. surf. coeff</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total air-air</td>
<td>0.318</td>
<td></td>
<td></td>
<td>3.147</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total surf.-surf.</td>
<td>0.334</td>
<td></td>
<td></td>
<td>2.992</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The boundary values are provided in figure 1. The corresponding mfile for plotting figure 1 can be found in Appendix 2.

![Figure 1. Boundary values. Left: One month period. Right: 48-hours period.](image-url)
Mathematical model description
The background of the multiphysics modeling is provided in Appendix 1 (Chapter 4 & 7 of the COMSOL 3.2 Modeling Guide).

Methodology
The methodology was to start simple with a 1D Heat & Moisture (HM) model and extend this model step-by-step to a full 3D dynamic Heat, Air & Moisture (HAM) model. Six cases, consisting of combinations of 1D, 2D & 3D geometries and airtight (HM) & air permeable (HAM) are presented.
1. ONE DIMENSIONAL
The roof of table I is selected.

1.1 Geometry & Boundary Conditions
The sub domains and boundaries are presented in figure 2.

Table II provides the labels of the sub domains and boundaries

<table>
<thead>
<tr>
<th>sub domain number</th>
<th>boundary number</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>wood panels</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>cellulose insulation</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>roof deck</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Surface to inside air</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>internal</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>internal</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Surface to outside air</td>
</tr>
</tbody>
</table>
1.2 Airtight (HM1D)

The mfile to model and simulate a 1D airtight roof is provided in Appendix 3. The aim is to integrate Heat and Moisture by multiphysics applications of Heat Transfer (defined by cell-structure appla) and Diffusion (defined by cell-structure applb). This complete mfile is explained step-by-step:

<table>
<thead>
<tr>
<th>Step</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Header</td>
</tr>
<tr>
<td>2</td>
<td>Clear memory</td>
</tr>
<tr>
<td>3</td>
<td>Load data, used for the indoor and external temperatures and vapor pressures as functions of time. See figure 1 and Appendix 2</td>
</tr>
<tr>
<td>4</td>
<td>Define 1D geometry using solid1. Start inside at x=0, followed by the thicknesses of wood, cellulose and roof deck, which ends at outside. Plot the geometry using geomplot. Figure 2 appears with the sub domain- and boundaries numbers. The reader should notice that numbers are assigned automatically.</td>
</tr>
<tr>
<td>5</td>
<td>Set the application mode of appla to Heat Transfer (variable T). The PDE and boundary specifications are presented in Appendix 1</td>
</tr>
<tr>
<td>6</td>
<td>Initialize at 10°C on whole domain</td>
</tr>
<tr>
<td>7</td>
<td>Define the PDE coefficients sets 1 (wood), 2(cellulose) and 3(roofdeck)</td>
</tr>
<tr>
<td>8</td>
<td>Assign set 1 to sub domain 1; set 2 to sub domain 2 and set 3 to sub domain 3.</td>
</tr>
<tr>
<td>9</td>
<td>Define 3 types of boundaries: internal ( q = h_i*(T_i-T) ), continuous and external ( q = h_e*(T_e-T) )</td>
</tr>
<tr>
<td>10</td>
<td>Assign type 1 to boundary number 1; type 2 to boundary numbers 2,3 and type 3 to boundary number 4</td>
</tr>
<tr>
<td>11</td>
<td>Set the application mode of applb to Diffusion (variable c equals vapor pressure). The PDE and boundary specifications are presented in Appendix 1</td>
</tr>
<tr>
<td>12</td>
<td>Initialize at 1000 Pa on whole domain</td>
</tr>
<tr>
<td>13</td>
<td>Define the PDE coefficient D wood (1), cellulose (2) and roof deck (3)</td>
</tr>
<tr>
<td>14</td>
<td>Assign set 1 to sub domain 1; set 2 to sub domain 2 and set 3 to sub domain 3.</td>
</tr>
<tr>
<td>15</td>
<td>Define 3 types of boundaries: internal ( N = \beta p_i*(P_i-P) ), continuous and external ( N = \beta p_e*(P_e-P) )</td>
</tr>
<tr>
<td>16</td>
<td>Assign type 1 to boundary number 1; type 2 to boundary numbers 2,3 and type 3 to boundary number 4</td>
</tr>
<tr>
<td>17</td>
<td>Commands that provides a solution</td>
</tr>
<tr>
<td>18</td>
<td>Calculate T en Pv at boundaries for each time step</td>
</tr>
<tr>
<td>19</td>
<td>Plot results (see figure 3)</td>
</tr>
<tr>
<td>20</td>
<td>save T en Pv at boundaries for analysis</td>
</tr>
</tbody>
</table>
The results are presented in the following figure:

*Figure 3. The solution of HM1D and boundary conditions for a two day period. Upper: Temperatures, Lower: Vapor pressures.*
1.3 Air permeable (HAM1D)

The complete mfile to model and simulate a 1D air permeable roof is provided in Appendix 4. The aim is to extend the heat and moisture model HM1D with an airflow through the construction from outside to inside. In addition to the previous model, a velocity through the construction has to be provided. If a steady air pressure difference between inside and outside of 2.5 Pa is assumed, a velocity of 0.1 µm/s can be estimated from the permeability's provided in Table I. The main differences between this mfile and the reference mfile (see HM1D) is explained step-by-step:

<table>
<thead>
<tr>
<th>Step</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>The Convection and Conduction Application mode is now selected</td>
</tr>
<tr>
<td>2.</td>
<td>The velocity is provided.</td>
</tr>
<tr>
<td>3.</td>
<td>The Convection and Diffusion Application mode is now selected</td>
</tr>
<tr>
<td>4.</td>
<td>Again, the velocity is provided</td>
</tr>
<tr>
<td>5.</td>
<td>Output is similar but with different variable names</td>
</tr>
</tbody>
</table>

Due to the low velocity the simulation results are almost identical with HM1D (as expected). In figure 4 to solution of HAM1D is compared with HM1D.

![Figure 4](image-url)

Figure 4 The surface conditions of HAM1D (o) versus HM1D (+) for a two day period. Upper: Temperatures, Lower: Vapor pressures.

**Verification**

Two relative verification results are presented below. First, the result of setting the velocity to zero in the HAM1D model should equal the HM1D model:
Figure 5. Verification results between models, for a two day period. Upper: Temperatures, Lower: Vapor pressures.

Second, the steady state of the HAM1D model (with air velocity) has been compared with the exact solution:

Figure 6. Verification of HAM1D with exact solution
2. TWO DIMENSIONAL

2.1 Geometry & Boundary Conditions

Figure 7. The sub domain numbers

Figure 8. The boundary numbers
2.2 Airtight (HM2D)

The complete mfile to model and simulate a 2D airtight roof-wall corner is provided in Appendix 5. The aim is to extend the 1D heat and moisture model HM1D to a 2D model. The main differences between this mfile and the reference mfile (see HM1D) is explained step-by-step:

<table>
<thead>
<tr>
<th>Step</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A 2D geometry is build using the <em>poly2</em> command. The results are plotted in figures 7 and 8.</td>
</tr>
<tr>
<td>2</td>
<td>The PDE equations settings are assigned to the 9 subdomains</td>
</tr>
<tr>
<td>3</td>
<td>An extra boundary type (zero flux) is added</td>
</tr>
<tr>
<td>4</td>
<td>Due to the large amount of boundaries a separate variable for bnd.ind is used (indvar). The boundary settings are assigned with this variable</td>
</tr>
<tr>
<td>5</td>
<td>The PDE equations settings are assigned to the 9 subdomains</td>
</tr>
<tr>
<td>6</td>
<td>The boundary settings are assigned</td>
</tr>
<tr>
<td>7</td>
<td>The output is similar but with different variable names</td>
</tr>
<tr>
<td>8</td>
<td>Create animations</td>
</tr>
<tr>
<td>9</td>
<td>Create surface plots</td>
</tr>
</tbody>
</table>

The results are presented below:

*Figure 9. The different materials represented by their heat conduction coefficients*
Figure 10. The temperature distribution after 48 hours

Figure 11. The vapor pressure distribution after 48 hours
2.3 Air permeable (HAM2D)
The complete mfile to model and simulate a 2D air permeable roof-wall corner is provided in Appendix 6. The aim is to extend the combine the 1D heat, air and moisture model HAM1D and the 2D heat and moisture model HM2D into a 2D model. The *main differences* between this mfile and the reference mfiles (see HAM1D & HM2D) is explained step-by-step:

<table>
<thead>
<tr>
<th>Step</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The velocity is calculated using a third PDE accounting for the air permeability.</td>
</tr>
<tr>
<td>2</td>
<td>The velocity is calculated using a third PDE accounting for the air permeability.</td>
</tr>
<tr>
<td>3</td>
<td>The third application calculates the mass transport through the construction given the air permeability's. Due to numerical problems using the original air permeability's (order 1e-9) the values are scaled with a factor 1e5.</td>
</tr>
</tbody>
</table>

The results are provided below:

![Figure 12. The temperature distribution after 48 hours](image-url)
Figure 13. The vapor pressure distribution after 48 hours

Figure 14. The atmospheric pressure distribution
Figure 15. The air velocity distribution
3. THREE DIMENSIONAL
Due to limited computer resources the size of the domain is limited from 0.5 to 0.25.

3.1 Geometry & Boundary Conditions

Figure 16. The sub domain numbers

Figure 17. The boundary numbers
3.2 Airtight (HM3D)

The complete mfile to model and simulate a 3D airtight roof-wall corner is provided in Appendix 7. The aim is to extend the 2D heat and moisture model HM2D to a 3D model. The main differences between this mfile and the reference mfile (see HM2D) is explained step-by-step:

<table>
<thead>
<tr>
<th>Step</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Due to limited computer resources the size of the domain is limited by replacing 0.5 with 0.25</td>
</tr>
<tr>
<td>2</td>
<td>A 3D geometry is building by extruding the 2D geometry (extrude) and by adding three more blocks (block3)</td>
</tr>
<tr>
<td>3</td>
<td>The boundary settings variable is set using figure</td>
</tr>
<tr>
<td>4</td>
<td>The PDE equations settings are assigned to the 12 sub domains</td>
</tr>
<tr>
<td>5</td>
<td>The PDE equations settings are assigned to the 12 sub domains</td>
</tr>
<tr>
<td>6</td>
<td>The line of interest for output is set.</td>
</tr>
</tbody>
</table>

The results are presented below:

![Figure 18. The different materials represented by there heat conduction coefficients](image)
Figure 19. The temperature distribution after 48 hours

Figure 20. The vapor pressure distribution after 48 hours.
3.3 Air permeable (HAM3D)

The complete mfile to model and simulate a 3D airtight roof-wall corner is provided in Appendix 8. The aim is to extend the HAM2D and HM3D to a 3D HAM model. The mfile is constructed by using the geometry of HM3D and the PDE model of HAM2D. The results are displayed below:

**Figure 21. The temperature distribution after 48 hours**

**Figure 22. The vapor pressure distribution after 48 hours**
Figure 23. The atmospheric pressure distribution

Figure 24. The air velocity distribution
## 4. SUMMARY OF FILES

The following files are included in the *HAMConstruction.zip* file downloadable from HAMLab website [http://sts.bwk.tue.nl/hamlab/](http://sts.bwk.tue.nl/hamlab/)

<table>
<thead>
<tr>
<th>filename</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td></td>
</tr>
<tr>
<td>HM1DEX1.m</td>
<td>1D Heat &amp; Moisture model of the roof</td>
</tr>
<tr>
<td>HAM1DEX1.m</td>
<td>1D Heat, Air &amp; Moisture model of the roof</td>
</tr>
<tr>
<td>HM2DEX1.m</td>
<td>2D Heat &amp; Moisture model of the roof/wall</td>
</tr>
<tr>
<td>HAM2DEX1.m</td>
<td>2D Heat, Air &amp; Moisture model of the roof/wall</td>
</tr>
<tr>
<td>HM3DEX1.m</td>
<td>3D Heat &amp; Moisture model of the roof/wall</td>
</tr>
<tr>
<td>HAM3DEX1.m</td>
<td>3D Heat, Air &amp; Moisture model of the roof/wall</td>
</tr>
<tr>
<td></td>
<td>with corresponding movies (* mpg) and profiles (<em>.mat)</em></td>
</tr>
<tr>
<td>OTHER</td>
<td></td>
</tr>
<tr>
<td>HAM1DEX0.m</td>
<td>verification model</td>
</tr>
<tr>
<td>HAM1DEX2.m</td>
<td>verification model</td>
</tr>
<tr>
<td>clplot.m</td>
<td>plot indoor and external climates</td>
</tr>
<tr>
<td>hampl1.m</td>
<td>plot verification results</td>
</tr>
<tr>
<td>pi600fun.m</td>
<td>indoor vapor pressure function</td>
</tr>
<tr>
<td>pe600fun.m</td>
<td>external vapor pressure function</td>
</tr>
<tr>
<td>ti600fun.m</td>
<td>indoor temperature function</td>
</tr>
<tr>
<td>tesl_roof600fun.m</td>
<td>external temperature function</td>
</tr>
<tr>
<td>psatf.m</td>
<td>saturation vapor pressure function</td>
</tr>
<tr>
<td>Case600_OpenData.mat</td>
<td>data uses for indoor and external climates</td>
</tr>
<tr>
<td>TslData.mat</td>
<td>data used for indoor and external climates</td>
</tr>
</tbody>
</table>
The Convection and Conduction Application Mode

In addition to heat transfer by conduction, the Convection and Conduction application mode includes heat transfer by convection. In the convection term for the equation that defines this application mode you can specify the velocity vector as an analytical expression. Alternatively, you can connect it directly to the solution of the equations of motion, for example, through a multiphysics coupling to the Incompressible Navier-Stokes application mode.

Variables and Space Dimensions

The Convection and Conduction application mode is available in 1D, 2D, 3D, Axial symmetry 1D, and Axial symmetry 2D.

Note: The optional Chemical Engineering Module also contains this Convection and Conduction application mode. In addition to the above-mentioned space dimensions, it features pseudo-2D and pseudo-3D geometry options in which it uses time as a second or third spatial dimension. This feature is useful in situations where convection in the direction of the flow is large. This often applies to reactors and equipment for unit operations.

PDE Formulation

In this application mode you can choose from two formulations:

\[ \delta_{ts} \rho C_{p} \frac{\partial T}{\partial t} + \nabla \cdot (-k \nabla T) = Q - \rho C_{p} u \cdot \nabla T \]  
(nonconservative)

\[ \delta_{ts} \rho C_{p} \frac{\partial T}{\partial t} + \nabla \cdot (-k \nabla T + \rho C_{p} u T) = Q \]  
(conservative)

In COMSOL Multiphysics the nonconservative formulation is the default for advection and diffusion types of equations. It assumes an incompressible fluid, which means that \( \nabla \cdot \mathbf{u} = 0 \). This ensures that no unphysical source term arises from a flow.
field where the incompressibility constraint, \( \nabla \cdot \mathbf{u} = 0 \), is not absolutely fulfilled. The nonconservative formulation puts the convective term on the right-hand side of the equation, which implies that setting the temperature gradient to zero directly expresses the convective boundary condition. This procedure avoids the use of interpolation and gives higher accuracy. For convective boundary conditions in heat balances, where the flow is incompressible and given by Darcy’s law, the use of the nonconservative mode gives the most simple and accurate model definition.

You can toggle between the nonconservative and conservative forms in the Application Mode Properties dialog box by selecting them in the Equation form list.

**Subdomain Settings**

The following table contains the quantities in the equations:

<table>
<thead>
<tr>
<th>COEFFICIENT</th>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \delta_{ts} )</td>
<td>Dts_T</td>
<td>Time-scaling coefficient</td>
</tr>
<tr>
<td>( \rho )</td>
<td>rho_T</td>
<td>Density</td>
</tr>
<tr>
<td>( C_p )</td>
<td>C_T</td>
<td>Heat capacity</td>
</tr>
<tr>
<td>( k )</td>
<td>k_T</td>
<td>Thermal conductivity</td>
</tr>
<tr>
<td>( k_{ij} )</td>
<td>k_{x_ix_j}</td>
<td>Thermal conductivity tensor, ( x_i )( x_j ) component</td>
</tr>
<tr>
<td>( Q )</td>
<td>Q_T</td>
<td>Heat source</td>
</tr>
<tr>
<td>( u, v, w )</td>
<td>u_T, v_T, w_T</td>
<td>Velocity in the ( x_1 ), ( x_2 ), and ( x_3 )-direction</td>
</tr>
</tbody>
</table>

For equations in 2D or 3D, pay special attention to the isotropic thermal conductivity, \( k \). If you select this coefficient, the application mode expands it to the diagonal of the thermal conductivity tensor, that is, \( k_{ij} \) equals \( k \).

**ARTIFICIAL DIFFUSION**

The Convection and Conduction application mode supports artificial diffusion using the following methods:

- Isotropic diffusion
- Streamline diffusion
- Crosswind diffusion

To specify and activate artificial diffusion:

1. Open the Subdomain Settings dialog box.
2 Click the **Physics** tab.

3 With at least one subdomain selected, click the **Artificial Diffusion** button.

See “Stabilization Techniques” on page 522 in the *COMSOL Multiphysics User’s Guide* for more information about artificial diffusion.

### Boundary Conditions

The available boundary conditions are:

<table>
<thead>
<tr>
<th>Boundary Condition</th>
<th>Description</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T = T_0 )</td>
<td>Temperature</td>
<td>( T )</td>
</tr>
<tr>
<td>(-\mathbf{n} \cdot (-k\nabla T + \rho C_p \mathbf{u} T) = q_0)</td>
<td>Heat flux</td>
<td>( n \cdot (-k\nabla T + \rho C_p \mathbf{u} T) = 0)</td>
</tr>
<tr>
<td>( \mathbf{n} \cdot (-k\nabla T) = 0)</td>
<td>Insulation/symmetry</td>
<td>( \mathbf{n} \cdot (-k\nabla T + \rho C_p \mathbf{u} T) = 0)</td>
</tr>
<tr>
<td>( \mathbf{n} \cdot (-k\nabla T + \rho C_p \mathbf{u} T) = 0)</td>
<td>Axial symmetry</td>
<td></td>
</tr>
</tbody>
</table>

The only difference between this mode and the Conduction application mode is that the heat flux contains the added convection term. In cases where convection across a boundary is much greater than diffusion, use the convective flux boundary condition. It sets the diffusive flux at the boundary to zero, but it allows convective flux to exit the domain.

**Note:** When working in an axisymmetry application mode, use the axial symmetry boundary condition only on the symmetry axis.

### Application Mode Variables

The Convection and Conduction application mode uses the following expressions and coefficients in boundary conditions, equations, and for postprocessing purposes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T )</td>
<td>S/B</td>
<td>Temperature</td>
<td>( T )</td>
</tr>
<tr>
<td>grad_T</td>
<td>S/N</td>
<td>Temperature gradient</td>
<td>(</td>
</tr>
<tr>
<td>NAME</td>
<td>TYPE</td>
<td>DESCRIPTION</td>
<td>EXPRESSION</td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>---------------------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>dflux_T</td>
<td>S</td>
<td>Conductive flux</td>
<td>([k \nabla T])</td>
</tr>
<tr>
<td>cflux_T</td>
<td>S</td>
<td>Convective flux</td>
<td>([\rho C_p T u])</td>
</tr>
<tr>
<td>tflux_T</td>
<td>S</td>
<td>Total heat flux</td>
<td>([-k \nabla T + \rho C_p T u])</td>
</tr>
<tr>
<td>ndflux_T</td>
<td>B</td>
<td>Normal conductive</td>
<td>(n \cdot (-k \nabla T))</td>
</tr>
<tr>
<td>ncflux_T</td>
<td>B</td>
<td>Normal convective</td>
<td>(\rho C_p T n \cdot u)</td>
</tr>
<tr>
<td>ntflux_T</td>
<td>B</td>
<td>Normal total heat</td>
<td>(n \cdot (-k \nabla T + \rho C_p T u))</td>
</tr>
<tr>
<td>dflux_Tx_i</td>
<td>V</td>
<td>Conductive flux,</td>
<td>(\sum_j -k \frac{\partial T}{\partial x_j})</td>
</tr>
<tr>
<td>cflux_Tx_i</td>
<td>V</td>
<td>Convective flux,</td>
<td>(\rho C_p T u_i)</td>
</tr>
<tr>
<td>tflux_Tx_i</td>
<td>V</td>
<td>Total heat flux,</td>
<td>(\sum_j -k_{ij} \frac{\partial T}{\partial x_j} + \rho C_p T u_i)</td>
</tr>
<tr>
<td>cellPe_T</td>
<td>S</td>
<td>Cell Peclet number</td>
<td>(\frac{\rho C_p u h}{k})</td>
</tr>
<tr>
<td>Dts_T</td>
<td>S</td>
<td>Time-scale factor</td>
<td>(\delta_t)</td>
</tr>
<tr>
<td>rho_T</td>
<td>S</td>
<td>Density</td>
<td>(\rho)</td>
</tr>
<tr>
<td>C_T</td>
<td>S</td>
<td>Heat capacity</td>
<td>(C_p)</td>
</tr>
<tr>
<td>k_T, kx_ky_T</td>
<td>S</td>
<td>Thermal conductivity</td>
<td>(k, k_{ij})</td>
</tr>
<tr>
<td>Q_T</td>
<td>S</td>
<td>Heat source</td>
<td>(Q)</td>
</tr>
<tr>
<td>u_T, v_T, w_T</td>
<td>S</td>
<td>Velocity of c, x_i component</td>
<td>(u_i)</td>
</tr>
<tr>
<td>Dm_T</td>
<td>S</td>
<td>Mean diffusion</td>
<td>(\sum_{i,j} k_{ij} \beta_i \beta_j / \beta)</td>
</tr>
<tr>
<td>res_T_cc</td>
<td>S</td>
<td>Equation residual</td>
<td>(\nabla \cdot (-k \nabla T + \rho C_p T u) - Q)</td>
</tr>
<tr>
<td>res_sc_T_cc</td>
<td>S</td>
<td>Shock-capturing residual</td>
<td>(\nabla \cdot (\rho C_p T u) - Q)</td>
</tr>
<tr>
<td>da_T</td>
<td>S</td>
<td>Total time-scale</td>
<td>(\delta_t \rho C_p)</td>
</tr>
<tr>
<td>q</td>
<td>B</td>
<td>Inward heat flux</td>
<td>(q_0)</td>
</tr>
<tr>
<td>T0</td>
<td>B</td>
<td>Prescribed temperature</td>
<td>(T_0)</td>
</tr>
</tbody>
</table>
**Note:** To form the complete application mode variable names, add a suffix consisting of an underscore and the application mode name (default: cc), for example, \( \text{tflux}_T_{\text{cc}} \). (This does not apply to the dependent variable for the temperature.)

The vector variables, indicated by V in the Type column, are not present in 1D versions of the Convection and Conduction application mode.
The Convection and Diffusion Application Mode

This application mode models the most common type of transport in chemical systems: transport by convection and diffusion. You can simulate transport by convection and diffusion in 1D, 2D, and 3D as well as for axisymmetric systems in 1D and 2D.

The dependent variable in the application mode is the concentration of mass, \( c \).

**Note:** The optional Chemical Engineering Module contains an application mode for the convection and diffusion of several species. It also supports modeling using pseudo-2D and pseudo-3D geometries.

**PDE Formulation**

The equations for the nonconservative and conservative formulations for a species, \( c \), are:

\[
\delta_t \frac{\partial c}{\partial t} + \nabla \cdot (-D \nabla c) = R - u \cdot \nabla c \quad \text{nonconservative}
\]

\[
\delta_t \frac{\partial c}{\partial t} + \nabla \cdot (-D \nabla c + c u) = R \quad \text{conservative}
\]

The nonconservative formulation is the default for advection and diffusion types of equations in COMSOL Multiphysics because the software assumes an incompressible fluid. Thus the term \( c \nabla \cdot u \) equals zero and gets dropped from the nonconservative formulation. This ensures that no nonphysical source term arises from a flow field where the incompressibility constraint, \( \nabla \cdot u = 0 \), is not absolutely fulfilled.

For stationary analysis the term with the time derivative gets dropped.
Subdomain Settings

The various coefficients used in the equations are:

<table>
<thead>
<tr>
<th>COEFFICIENT</th>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \delta_{ts} )</td>
<td>Dts_c</td>
<td>Time-scaling coefficient</td>
</tr>
<tr>
<td>( D )</td>
<td>D_c</td>
<td>Diffusion coefficient or tensor</td>
</tr>
<tr>
<td>( D_{ij} )</td>
<td>Dx_xj_c</td>
<td>Diffusion coefficient tensor, ( x_i x_j ) component</td>
</tr>
<tr>
<td>( R )</td>
<td>R_c</td>
<td>Reaction rate</td>
</tr>
<tr>
<td>( u, v, w )</td>
<td>u_c, v_c, w_c</td>
<td>Velocity in the ( x_1 )-, ( x_2 )-, and ( x_3 )-direction</td>
</tr>
</tbody>
</table>

For equations in 2D or 3D, pay special attention to the isotropic diffusion coefficient, \( D \). If you select this coefficient, the application mode expands it to the diagonal of the diffusion coefficient tensor, that is, \( Dx_xi \) equals \( D \).

You specify the equation coefficients in the **Subdomain Settings** dialog box.

**ARTIFICIAL DIFFUSION**

The Convection and Diffusion application mode supports artificial diffusion using the following methods:

- Isotropic diffusion
- Streamline diffusion
- Crosswind diffusion

To specify and activate artificial diffusion:

1. Open the **Subdomain Settings** dialog box.
2. Click the **Physics** tab.
3. With at least one subdomain selected, click the **Artificial Diffusion** button.

See “Stabilization Techniques” on page 522 in the *COMSOL Multiphysics User’s Guide* for more information about artificial diffusion.
Boundary Conditions

The available boundary conditions are

<table>
<thead>
<tr>
<th>BOUNDARY CONDITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c = c_0$</td>
<td>Concentration</td>
</tr>
<tr>
<td>$-\mathbf{n} \cdot (-D\nabla c + \mathbf{c}\mathbf{u}) = N_0$</td>
<td>Flux</td>
</tr>
<tr>
<td>$\mathbf{n} \cdot (-D\nabla c + \mathbf{c}\mathbf{u}) = 0$</td>
<td>Insulation/Symmetry</td>
</tr>
<tr>
<td>$\mathbf{n} \cdot (-D\nabla c) = 0$</td>
<td>Convective flow</td>
</tr>
<tr>
<td>$\mathbf{n} \cdot (-D\nabla c + \mathbf{c}\mathbf{u}) = 0$</td>
<td>Axial symmetry</td>
</tr>
</tbody>
</table>

**Concentration** In the equation for the concentration boundary condition, $c_0$ is a user-specified concentration.

**Diffusive Flux** In the equation for the flux condition, $N_0$ is an arbitrary user-specified flux expression.

**Axial Symmetry** The axial symmetry condition is identical to the insulation/symmetry condition. It is available only for axisymmetric models using cylindrical coordinate systems. Use this boundary condition only on the symmetry axis.

You specify boundary conditions in the **Boundary Settings** dialog box.
**Application Mode Variables**

The Convection and Diffusion application mode uses the following expressions and coefficients in boundary conditions, equations, and for postprocessing purposes:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>DESCRIPTION</th>
<th>EXPRESSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c )</td>
<td>S/B</td>
<td>Concentration</td>
<td>( c )</td>
</tr>
<tr>
<td>( \text{grad}_c, c \partial x_i )</td>
<td>S/V</td>
<td>Concentration gradient</td>
<td>( \nabla c, \frac{\partial c}{\partial x_i} )</td>
</tr>
<tr>
<td>( \text{dflux}_c )</td>
<td>S</td>
<td>Diffusive flux</td>
<td>(</td>
</tr>
<tr>
<td>( \text{cflux}_c )</td>
<td>S</td>
<td>Convective flux</td>
<td>(</td>
</tr>
<tr>
<td>( \text{tflux}_c )</td>
<td>S</td>
<td>Total flux</td>
<td>(</td>
</tr>
<tr>
<td>( \text{nflux}_c )</td>
<td>B</td>
<td>Normal diffusive flux</td>
<td>( n \cdot (-D \nabla c) )</td>
</tr>
<tr>
<td>( \text{ncflux}_c )</td>
<td>B</td>
<td>Normal convective flux</td>
<td>( c n \cdot u )</td>
</tr>
<tr>
<td>( \text{ntflux}_c )</td>
<td>B</td>
<td>Normal total flux</td>
<td>( n \cdot (-D \nabla c + c u) )</td>
</tr>
<tr>
<td>( \text{dflux} _c _x_i )</td>
<td>V</td>
<td>Diffusive flux, ( x _i ) component</td>
<td>( \sum_j -D_{ij} \frac{\partial c}{\partial x_j} )</td>
</tr>
<tr>
<td>( \text{cflux} _c _x_i )</td>
<td>V</td>
<td>Convective flux, ( x _i ) component</td>
<td>( c u_i )</td>
</tr>
<tr>
<td>( \text{tflux} _c _x_i )</td>
<td>V</td>
<td>Total flux, ( x _i ) component</td>
<td>( \sum_j -D_{ij} \frac{\partial c}{\partial x_j} + c u_i )</td>
</tr>
<tr>
<td>( \text{cellPe} _c )</td>
<td>S</td>
<td>Cell Peclet number</td>
<td>( \frac{</td>
</tr>
<tr>
<td>( \text{Dts} _c )</td>
<td>S</td>
<td>Time-scaling coefficient</td>
<td>( \delta_{ts} )</td>
</tr>
<tr>
<td>( \text{udl} _c )</td>
<td>S</td>
<td>Dimensionless velocity</td>
<td>( u_{dl} )</td>
</tr>
<tr>
<td>( \text{D} _c, D \partial x_j _c )</td>
<td>S</td>
<td>Diffusion coefficient</td>
<td>( D, D_{ij} )</td>
</tr>
<tr>
<td>( \text{R} _c )</td>
<td>S</td>
<td>Reaction rate</td>
<td>( R )</td>
</tr>
<tr>
<td>( u _c, v _c, w _c )</td>
<td>S</td>
<td>Velocity of ( c, x _i ) component</td>
<td>( u_i )</td>
</tr>
<tr>
<td>( N _c )</td>
<td>B</td>
<td>Inward flux</td>
<td>( N_0 )</td>
</tr>
<tr>
<td>( c0 _c )</td>
<td>B</td>
<td>Concentration</td>
<td>( c_0 )</td>
</tr>
<tr>
<td>( \text{beta} _c _x_i )</td>
<td>S</td>
<td>Convective field, ( x _i ) component</td>
<td>( u_i )</td>
</tr>
</tbody>
</table>
CHAPTER 4: DIFFUSION

The vector expressions indicated with a $\nabla$ in the Type column are not present in 1D versions of the Convection and Diffusion application mode.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>DESCRIPTION</th>
<th>EXPRESSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dm_c</td>
<td>S</td>
<td>Mean diffusion coefficient</td>
<td>$\sum_{j} D_{ij} \beta_i \beta_j$</td>
</tr>
<tr>
<td>res_c</td>
<td>S</td>
<td>Equation residual</td>
<td>$\nabla \cdot (-D \nabla c + c u) - R$</td>
</tr>
<tr>
<td>res_sc_c</td>
<td>S</td>
<td>Shock capturing residual</td>
<td>$\nabla \cdot (c u) - R$</td>
</tr>
<tr>
<td>da_c</td>
<td>S</td>
<td>Total time-scale factor</td>
<td>$\delta_{ts}$</td>
</tr>
</tbody>
</table>

**Note:** To form the complete application mode variable names, add a suffix consisting of an underscore and the application mode name (default: cd), for example, $dflux_d_{cd}$. (This does not apply to the dependent variable for the concentration.)
Appendix 2: Mfile for plotting boundary values

%CLPLOT CLimate PLOT for used Heat Air & Moisture Examples
%       Denver climate is used (=BESTEST)
%       Indoor climate is simulated with additional loads heat & moisture
%       (see Annex41 Subtask 1 Commonn exercise 1)
%       Boundarie value functions:
%       External solar-air temperature: tesl_roof600fun(t)
%       External vapor pressure: pe600fun(t)
%       Internal air temperature: ti600fun(t)
%       Internal vapor pressure: pi600fun(t)
% JvS 2006/Aug

clear all
%LOAD DATA FROM FILES
global tout Rh Rhe Te Tair t_roof Tesl_roof
load Case600_OpenData.mat
load TslData.mat
%DEFINE TIME
tu=[0:3600:31*24*3600];
tdag=tu/(24*3600);
tu48=[0:3600:48*3600];
tdag48=tu48/(24*3600);

%PLOT MONTH 1
figure(1)
subplot(211)
plot(tdag,ti600fun(tu),tdag,tesl_roof600fun(tu))
legend('Ti','Te solair')
ylabel('T [oC]')
subplot(212)
plot(tdag,pi600fun(tu),tdag,pe600fun(tu))
legend('Pi','Pe')
ylabel('P [Pa]')
xlabel('time [days]')

%PLOT 48 HOURS
figure(2)
subplot(211)
plot(tdag48,ti600fun(tu48),tdag48,tesl_roof600fun(tu48))
legend('Ti','Te solair')
ylabel('T [oC]')
subplot(212)
plot(tdag48,pi600fun(tu48),tdag48,pe600fun(tu48))
legend('Pi','Pe')
ylabel('P [Pa]')
xlabel('time [days]')
Appendix 3: Mfile REFERENCE Model HM1D

%HM1DEX1 Heat & Moisture 1D EXample 1
% JvS 2006/Aug

clear all

%LOAD Data for Indoor and External climates needed for
% ti600fun(t);tesl_roof600fun(t);pi600fun(t);pe600fun(t)
global tout Rh Rhe Te Tair t_roof Tesl_roof
load Case600_OpenData.mat
load TslData.mat

%1D GEOMETRY
s=solid1([0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)]);

% PLOT SUBDOMAINS NUMBERS & BOUNDARY NUMBERS
figure(1)
geomplot(s,'sublabels','on' ,'pointlabels','on');

%HEAT TRANSFER (appla)
%Select Application (default variable T equals Temperature [oC])
appla.mode.class='HeatTransfer';
appla.assignsuffix='_ht';

%Initialize at 10 oC
appla.equi.init=10;

%Define the different PDE coefficients acting on subdomains
appla.equi.k= {'0.16' '0.04' '0.14' };
appla.equi.rho= {'395' '55' '530' };
appla.equi.C= {'1880' '1880' '1880' };

%Assign above PDE coefficients to the different subdomain numbers
appla.equi.ind=[1 2 3];

%Define the different PDE coefficients acting on boundaries
appla.bnd.q={'8.29*(ti600fun(t)-T)'
'29.3*(tesl_roof600fun(t)-T)'};
appla.bnd.type= {'q' 'cont' 'q' };

%Assign above PDE coefficients to the different boundary numbers
appla.bnd.ind= [1 2 3];

%MOISTURE TRANSFER (applb)
%Select Application (default variable c equals Vapor Pressure Pv [Pa])
applb.mode.class='FlDiffusion';
applb.assignsuffix='_di';

%Initialize at 1000 Pa
applb.equi.init=1000;

%Define the different PDE coefficients acting on subdomains
applb.equi.D= {'psatf(T)*1.8e-10/(101*2.1)' 'psatf(T)*1.8e-10/(14*1.4)' 'psatf(T)*1.8e-10/(120*95)'};

%Assign above PDE coefficients to the different subdomain numbers
applb.equi.ind=[1 2 3];
%Define the different PDE coefficients acting on boundaries
applb.bnd.N= {'2e-8*(pi600fun(t)-c)' [0] '8e-8*(pe600fun(t)-c)'}; 
%%
applb.bnd.type={'N' 'cont' 'N'};

%Assign above PDE coefficients to the different boundary numbers
applb.bnd.ind= [1 2 2 3];

%Commands to mesh, integrate applications and solve (0-48 hours)
fem.geom=s;
fem.appl={appla applb};
fem.mesh=meshinit(fem);
%fem.mesh=meshrefine(fem); %uncomment for higher mesh resolution
fem=multiphysics(fem);
fem.xmesh=meshextend(fem);
fem.sol=femtime(fem,'tlist',[0:3600:48*3600]);

%OUTPUT (timeseries of T & P at specific positions)
%time stepping
tu=fem.sol.tlist;
ntu=length(tu);
tdag1D=tu/(24*3600);

%positions (boundaries and material interfaces)
xp1D=[0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)];

%T & P at xp1D for all solution numbers i.e. for all time steps
Txp1D=postinterp(fem,'T',xp1D,'Solnum',[1:ntu]);
Px1D=postinterp(fem,'c',xp1D,'Solnum',[1:ntu]);

%PLOT Results
figure(2)
subplot(211)
pplot(tdag1D,Txp1D,tdag1D,ti600fun(tu),tdag1D,tesl_roof600fun(tu))
subplot(212)
pplot(tdag1D,Px1D,tdag1D,pi600fun(tu),tdag1D,pe600fun(tu))

%SAVE to MatLab mat file
save HM1DEX1Data tdag1D xp1D Txp1D Px1D
Appendix 4: Mfile HAM1D model

%HAM1DEX1 Heat, AIR & Moisture 1D EXample 1, Extension of HM1DEX1
% Comments %% are placed where this file differs with HM1DEX1
% To understand this file, first study HM1DEX1.m
%
% JvS 2006/Aug

clear all

global tout Rh Rhe Te Tair t_roof Tesl_roof
load Case600_OpenData.mat
load TslData.mat

s=solid1([0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)]);
figure(1)
geomplot(s,'sublabels','on' ,'pointlabels','on');

%%The Convection and Conduction Application mode is now selected:
appla.mode.class='FlConvCond'; %%% appla.assignsuffix='_cc'; %%%
appla.equ.init=10;
appla.equ.k= {'0.16' '0.04' '0.14' };
appla.equ.rho= {'395' '55' '530' };
appla.equ.C= {'1880' '1880' '1880' };

%%Define PDE coefficient velocity
appla.equ.u={'-1e-7'}; %%%
appla.equ.ind=[1 2 3];
appla.bnd.q= {'8.29*(ti600fun(t)-T)' [0] '29.3*(tesl_roof600fun(t)-T)'}; %%%
appla.bnd.type= {'q' 'cont' 'q'};
appla.bnd.ind= [1 2 2 3];

%%The Convection and Diffusion Application mode is now selected:
applb.mode.class='FlConvDiff'; %%%
applb.assignsuffix='_cd'; %%%
applb.equ.init=1000;
applb.equ.D= {'psatf(T)*1.8e-10/(101*2.1)' 'psatf(T)*1.8e-10/(14*1.4)' 'psatf(T)*1.8e-10/(120*95)'};

%%Define PDE coefficient velocity
applb.equ.u={'-1e-7'}; %%%
applb.equ.ind=[1 2 3];
applb.bnd.N= {'psatf(T)/101*2e-8*(pi600fun(t)-c)' [0] 'psatf(T)/120*8e-8*(pe600fun(t)-c)'}; %%%
applb.bnd.type= {'N' 'cont' 'N'};
applb.bnd.ind= [1 2 2 3];

fem.geom=s;
fem.appl={appla applb};
fem.mesh=meshinit(fem);
%fem.mesh=meshrefine(fem); %Uncomment for more accurate mesh
fem=multiphysics(fem);
fem.xmesh=meshextend(fem);
fem.sol=femtime(fem,'tlist',[0:3600:48*3600]);

%OUTPUT similar but with different variable names
          tu=fem.sol.tlist;
          ntu=length(tu);
          Atdag1D=atu/(24*3600);

          Axp1D=[0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)];
          ATxp1D=postinterp(fem,'T',Axp1D,'Solnum',[1:ntu]);
          APxp1D=postinterp(fem,'c',Axp1D,'Solnum',[1:ntu]);

          figure(2)
          subplot(211)
          plot(Atdag1D,ATxp1D,Atdag1D,ti600fun(tu),Atdag1D,tesl_roof600fun(tu))
          subplot(212)
          plot(Atdag1D,APxp1D,Atdag1D,pi600fun(tu),Atdag1D,pe600fun(tu))

          save HAM1DEX1Data Atdag1D Axp1D ATxp1D APxp1D
Appendix 5: Mfile HM2D model

%HM2DEX1 Heat & Moisture 2D EXample 1  Extension of HM1DEX1
% Comments %% are placed where this file differs with HM1DEX1
% To understand this file, first study HM1DEX1.m

clear all
global tout Rh Rhe Te Tair t_roof Tesl_roof
load Case600_OpenData.mat
load TslData.mat

% 2D GEOMETRY
xp=[0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)];
yp=[0 0.012 (0.012+0.066) (0.012+0.066 + 0.009)];
roofwood=poly2([0 0 0.5 0.5],[0.5 (0.5+xp(2)) (0.5+xp(2)) 0.5]);
roofcel =poly2([0 0 0.5 0.5],[0.5+xp(2)) (0.5+xp(2)) 0.5+xp(3))
roofdeck=poly2([0 0 0.5 0.5],[0.5+xp(3)) (0.5+xp(4)) 0.5+xp(4));
wallwood=poly2([0 0 yp(2) yp(2)] ,[0 0.5+xp(3) 0.5+xp(3) 0]);
wallcel =poly2([yp(2) yp(2) yp(3) yp(3)] ,[0 0.5+xp(2) 0.5+xp(2) 0]);
wallsid =poly2([yp(3) yp(3) yp(4) yp(4)] ,[0 0.5 0.5 0]);
s=roofwood+roofcel+roofdeck+wallwood+wallcel+wallsid;

figure(1)
geomplot(s,'sublabels','on' );
axis([-0.1 0.7 -0.1 0.7])
figure(2)
geomplot(s,'edgelabels','on' );
axis([-0.1 0.7 -0.1 0.7])

appla.mode.class='HeatTransfer';
appla.assignsuffix='_ht';
appla.equ.init=20;
appla.equ.k= {'0.16' '0.04' '0.14' '0.16'};
appla.equ.rho= {'395' '55' '530' '395'};
appla.equ.C= {'1880' '1880' '1880' '1880'};

%There are 9 sub domains
appla.equ.ind=[3 3 3 3 2 2 2 1 1];

%An extra boundary type (zero flux) is added
appla.bnd.q={'8.29*(ti600fun(t)-T)' [0] '29.3*(tesl_roof600fun(t)-T)'
[0]};
appla.bnd.type= {'q' 'cont' 'q' 'q0'};

%Due to the large amount of boundaries variable for bnd.ind : indvar
%First set all boundaries to type 2 (='cont')
sub=flgeomnmr(s);
negd=flgeomnes(s);
indvar=2*ones(1,negd);
%Second re-assign the other types (1,3,4)
indvar([22 23])=1;
indvar([1 3 5 7 9 ] )=3;
indvar([2 11 18 24 25 26])=4;
appla.bnd.ind= indvar;
applb.mode.class='FLDiffusion';
applb.assignsuffix='_di';
applb.equ.init=1000;
applb.equ.D= {'psat(T)*1.8e-10/(101*2.1)' 'psat(T)*1.8e-
10/(14*1.4)' 'psat(T)*1.8e-10/(120*95)' };%There are 9 sub domains
applb.equ.ind=[3 3 3 3 2 2 2 1 1];
applb.bnd.N= {'psat(T)/101*2e-8*(pi600fun(t)-c)'  [0]
'psat(T)/120*8e-8*(pe600fun(t)-c)' [0]}; %
applb.bnd.type={'N'    'cont'    'N' ,    'N0'};
%There negd boundaries
applb.bnd.ind= index;

fem.geom=s;
fem.mesh=meshinit(fem);
fem.appl={appla applb};
fem=multiphysics(fem);
fem.xmesh=meshextend(fem);
fem.sol=femtime(fem,'tlist',[0:3600:48*3600]);

%OUTPUT similar but with different variable names
tu=fem.sol.tlist;
ntu=length(tu);
tdag=tu/(24*3600);

p2Da=[0.09 0.09 0.09 0.09 ]
     (xp+0.5)];
p2Db=[0.5 0.5 0.5 0.5
     (xp+0.5)];
tdag2D=tu/(24*3600);
Txp2Da=postinterp(fem,'T',p2Da,'Solnum',[1:ntu]);
Pxp2Da=postinterp(fem,'c',p2Da,'Solnum',[1:ntu]);
Txp2Db=postinterp(fem,'T',p2Db,'Solnum',[1:ntu]);
Pxp2Db=postinterp(fem,'c',p2Db,'Solnum',[1:ntu]);
save HM2DEX1Data tdag2D p2Da Txp2Da Pxp2Da p2Db Txp2Db Pxp2Db

%CREATE ANIMATIONS
figure(3)
Mfilm=postmovie(fem,'tridata','T','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HM2DEX1_T_2D')
clear mapM Mfilm

figure(4)
Mfilm=postmovie(fem,'tridata','c','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HM2DEX1_P_2D')

%PLOT SOLUTIONS
figure(5)
postplot(fem,'tridata','T','Solnum',49)

figure(6)
postplot(fem,'tridata','c','Solnum',49)
Appendix 6: Mfile HAM2D model

%HAM2DEX1 Heat, AIR & Moisture 2D EXample 1 Extension of HM2DEX1 & HAM1DEX1
% JvS 2006/Aug

clear all

global tout Rh Rhe Te Tair t_roof Tesl_roof
load Case600_OpenData.mat
load Ts1Data.mat

% 2D GEOMETRY
xp=[0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)];
yp=[0 0.012 (0.012+0.066) (0.012+0.066 + 0.009)];

roofwood=poly2([0 0 0.5 0.5], [0.5 (0.5+xp(2)) (0.5+xp(2)) 0.5]);
roofcel =poly2([0 0 0.5 0.5], [(0.5+xp(2)) (0.5+xp(3)) (0.5+xp(3)) (0.5+xp(2))]);
roofdeck=poly2([0 0 0.5 0.5], [(0.5+xp(3)) (0.5+xp(4)) (0.5+xp(4)) (0.5+xp(3))]);
wallwood=poly2([0 0 yp(2) yp(2)], [0 0.5+xp(3) 0.5+xp(3) 0]);
wallcel =poly2([yp(2) yp(2) yp(3) yp(3)], [0 0.5+xp(2) 0.5+xp(2) 0]);
wallsid =poly2([yp(3) yp(3) yp(4) yp(4)], [0 0.5 0.5 0]);

s=roofwood+roofcel+roofdeck+wallwood+wallcel+wallsid;

figure(1)
geomplot(s,'sublabels','on');
axis([-0.1 0.7 -0.1 0.7])

figure(2)
geomplot(s,'edgelabels','on');
axis([-0.1 0.7 -0.1 0.7])

%%The Convection and Conduction Application mode is now selected:
appla.mode.class='FlConvCond'; %
appla.assignsuffix='_cc'; %
appla.equ.init=10;
appla.equ.k= {'0.16' '0.04' '0.14'};
appla.equ.rho= {'395' '55' '530'};
appla.equ.C= {'1880' '1880' '1880'};

%%Define PDE coefficient velocity
appla.equ.u='1e-5*fluxx_ht';
appla.equ.v='1e-5*fluxy_ht';

%%
%There are 9 sub domains
appla.equ.ind=[3 3 3 3 2 2 1 1 1];

%An extra boundary type (zero flux) is added
appla.bnd.q= [8.29*(ti600fun(t)-T) [0]
              '29.3*(tesl_roof600fun(t)-T)' [0]]; %
appla.bnd.type= {'q' 'cont' 'q' 'q0'};

1
%Due to the large amount of boundaries do following:
%Seperate for bnd.ind : indvar
nsub=flgeomnmr(s);
negd=flgeomnes(s);
indvar=2*ones(1,negd);
%Second re-assign the other types (1,3,4)
indvar([22 23])=1;
indvar([1 3 5 7 9])=3;
indvar([2 11 18 24 25 26])=4;
appla.bnd.ind= indvar;

%%The Convection and Diffusion Application mode is now selected:
applb.mode.class='FlConvDiff'; %
applb.assignsuffix='_cd'; %
applb.equ.init=1000;
applb.equ.D= {'psatf(T)*1.8e-10/(101*2.1)'
            'psatf(T)*1.8e-10/(14*1.4)'
            'psatf(T)*1.8e-10/(120*95)'};

%%Define PDE coefficient velocity
applb.equ.u=('1e-5*fluxx_ht');
applb.equ.v=('1e-5*fluxy_ht');

%There are 9 sub domains
applb.equ.ind=[3 3 3 3 3 2 2 2 2 1 1];
applb.bnd.N= {'psatf(T)/101*2e-8*(pi600fun(t)-c)' [0]
            'psatf(T)/120*8e-8*(pe600fun(t)-c)' [0]}; %
applb.bnd.type={'N' 'cont' 'N' 'N0'};

%There negd boundaries
applb.bnd.ind= indvar;

%P (atmospheric pressure difference modeling with potential flow)
applc.mode.class='HeatTransfer';
applc.dim=('p');
applc.assignsuffix='_ht';
applc.equ.init=0;
applc.equ.rho=1;
applc.equ.C=1;
applc.equ.k= {'1e-4' '5.5' '1e-4'};
applc.equ.ind=[3 3 3 3 2 2 2 2 1 1];
applc.bnd.T0= {'0' [0] '2.5' [0]};
applc.bnd.type=('T' 'cont' 'T' 'q0');
applc.bnd.ind= indvar;

fem.geom=s;
fem.mesh=meshinit(fem);
fem.appl={appla applb applc};
fem=multiphysics(fem);
fem.xmesh=meshextend(fem);
fem.sol=femtime(fem,'tlist',[0:3600:48*3600]);

%OUTPUT similar but with different variable names
tu=fem.sol.tlist;
tu=ntu=

A1p2Da=
0.09 0.09 0.09 0.09
(xp+0.5)
];

A1p2Db=[
0.5 0.5 0.5 0.5
(xp+0.5)
];

A1tdag2D=tu/(24*3600);

A1T xp2Da=postinterp(fem,'T',A1p2Da,'Solnum',[1:ntu]);
A1P xp2Da=postinterp(fem,'c',A1p2Da,'Solnum',[1:ntu]);
A1T xp2Db=postinterp(fem,'T',A1p2Db,'Solnum',[1:ntu]);
A1P xp2Db=postinterp(fem,'c',A1p2Db,'Solnum',[1:ntu]);

save HAM2DEX1Data A1tdag2D A1p2Da A1T xp2Da A1P xp2Da A1p2Db A1T xp2Db
A1P xp2Db

figure(3)
Mfilm=postmovie(fem,'tridata','T','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HAM2DEX1_T_2D')
clear mapM Mfilm

figure(4)
Mfilm=postmovie(fem,'tridata','c','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HAM2DEX1_P_2D')

figure(5)
Mfilm=postmovie(fem,'tridata','p','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HAM2DEX1_Patm_2D')

figure(6)
postplot(fem,'tridata','T','Solnum',ntu)

figure(7)
postplot(fem,'tridata','c','Solnum',ntu)
title('Pv')

figure(8)
postplot(fem,'tridata','p','Solnum',ntu)
title('Patm')

figure(9)
postplot(fem,'arrowdata', {'fluxx_ht', 'fluxy_ht'})

figure(10)
postplot(fem,'tridata','k_cc')

figure(11)
postplot(fem,'tridata','D_c_cd')

figure(12)
postplot(fem,'tridata','k_ht')
Appendix 7: Mfile HM3D model

%HM3DEX1SMALL Heat & Moisture 3D EXample 1 Extension of HM2DEX1
% Comments %% are placed where this file differs with HM2DEX1
% To understand this file, first study HM2DEX1.m
% A smaller domain is used 0.25 instead of 0.5
%
%JvS 2006/Aug

clear all

load Case600_OpenData.mat
load TslData.mat
xp=[0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)];
yp=[0 0.012 (0.012+0.066) (0.012+0.066 + 0.009)];
roofwood=poly2([0 0 0.25 0.25] ,[0.25 (0.25+xp(2)) (0.25+xp(2))]);
roofcel  =poly2([0 0 0.25 0.25] ,[(0.25+xp(2)) (0.25+xp(3)) (0.25+xp(2))]);
roofdeck=poly2([0 0 0.25 0.25] ,[(0.25+xp(3)) (0.25+xp(2)) (0.25+xp(4)) (0.25+xp(3))]);
wallwood=poly2([0 0 yp(2) yp(2)] ,[0 0.25+xp(3) 0.25+xp(3) 0.25+xp(3) 0]);
wallcel  =poly2([yp(2) yp(2) yp(3) yp(3)] ,[0 0.25+xp(2) 0.25+xp(2) 0.25+xp(2) 0]);
wallsid =poly2([yp(3) yp(3) yp(4) yp(4)] ,[0 0.25 0.25 0]);
s=roofwood+roofcel+roofdeck+wallwood+wallcel+wallsid;

%3D GEOMETRY : Extrude 2D + addtional 3 blocks
blk1=extrude(s,'distance',0.25);
blk2=block3(0.25-yp(4),0.25,0.066,'pos',[yp(4),0,0.25-0.066]);
blk3=block3(0.25,0.25+xp(4),0.009,'pos',[0,0,0.25]);
blk4=block3(0.25-yp(4),0.25,0.012,'pos',[yp(4),0,0.25-0.066-0.012]);
blk=blk1+blk2+blk3+blk4;

figure(1)
geomplot(blk,'sublabels','on','transparency',0);
xlabel('x')
ylabel('y')
zlabel('z')

% 1-2-3
figure(2)
geomplot(blk,'facelabels','on','transparency',0);

indvar=zeros(1,nbnd);
indvar([44 47 52])=1;
indvar([1 4 7 8 12 16 20 21])=3;
indvar([2 3 5 10 14 18 23 24 28 32 36 37 41 46 49 55 56 57 58 59 60])=4;
indvar([6 9 11 13 15 17 19 22 25 26 27 29 30 31 33 34 35 38 39 40 42 43 45 48 50 51 53 54])=2;

appla.mode.class='HeatTransfer';
appla.assignsuffix='_ht';
appla.equ.init=20;
appla.equ.k= {'0.16' '0.04' '0.14' };
appla.equ.rho= {'395' '55' '530' };  
appla.equ.C= {'1880' '1880' '1880' };

%There are 12 sub domains use comsol GUI assign sub domains
appla.equ.ind=[3 3 3 3 3 2 2 2 1 1 1 2];

appla.bnd.q= ['8.29*(ti600fun(t)-T)' [0]  
'29.3*(tesl_roof600fun(t)-T)' [0]];  
appla.bnd.type= {'q' 'cont' 'q' 'q0'};
appla.bnd.ind= indvar;

applb.mode.class='FlDiffusion';
applb.assignsuffix='_di';
applb.equ.init=1000;
applb.equ.D= {'psatf(T)*1.8e-10/(101*2.1)'  'psatf(T)*1.8e-10/(14*1.4)'  'psatf(T)*1.8e-10/(120*95)' };

%There are 12 sub domains
applb.equ.ind=[3 3 3 3 3 2 2 2 1 1 1 2];

applb.bnd.N=  ['psatf(T)/101*2e-8*(pi600fun(t)-c)' [0]  
'psatf(T)/120*8e-8*(pe600fun(t)-c)' [0]];  
applb.bnd.type= {'N' 'N' 'N'};
applb.bnd.ind= indvar;

fem.geom=blk;
fem.mesh=meshinit(fem);
fem.appl={appla applb};
fem=multiphysics(fem);
fem.xmesh=meshextend(fem);
fem.sol=femtime(fem,'tlist',[0:3600:48*3600]);

%%OUTPUT%%%
tu=fem.sol.tlist;
ntu=length(tu);
tdag=tu/(24*3600);
p3D=[
0 0.010 (0.010+0.1118) (0.010+0.1118+0.019);
0.25 0.25 0.25 0.25;
0.25 0.25 0.25 0.25
];

tdag3D=tu/(24*3600);  
Txp3D=postinterp(fem,'T',p3D,'Solnum',[1:ntu]); 
Pxp3D=postinterp(fem,'c',p3D,'Solnum',[1:ntu]);

save HM3DEX1Data tda3D p3D Txp3D Pxp3D

figure(3)  
Mfilm=postmovie(fem,'tridata','T','Repeat',1);  
mapM=colormap;  
mpgwrite(Mfilm,mapM,'HM3DEX1_T_3D')

clear mapM Mfilm

figure(4)  
Mfilm=postmovie(fem,'tridata','c','Repeat',1);  
mapM=colormap;  
mpgwrite(Mfilm,mapM,'HM3DEX1_P_3D')

figure(5)  
postplot(fem,'tridata','T','Solnum',ntu)

figure(6)  
postplot(fem,'tridata','c','Solnum',ntu)
Appendix 8: Mfile HAM3D model

%HAM3DEX1     Heat & Air Moisture 3D EXample 1 Extension of both
HM2DEX1 & HAM2DEX1
%             Comments %% are placed where this file differs with
HM2DEX1
%             To understand this file, first study HM2DEX1.m &
HAM3DEX1.m
%             A smaller domain is used 0.25 instead of 0.5 to prevent
%             memory overflow due to limited computer memory
%
%JvS 2006/Aug

clear all

global tout Rh Rhe Te Tair t_roof Tesl_roof
load Case600_OpenData.mat
load TslData.mat

xp=[0 0.010 (0.010+0.1118) (0.010+0.1118+0.019)];
yp=[0 0.012 (0.012+0.066) (0.012+0.066 + 0.009)];

roofwood=poly2([0 0 0.25 0.25],
(0.25+xp(2)) (0.25));
roofcel =poly2([0 0 0.25 0.25],
(0.25+xp(2)) (0.25+xp(3))
(0.25+xp(3)) (0.25+xp(2))
);
roofdeck=poly2([0 0 0.25 0.25],
(0.25+xp(3)) (0.25+xp(4))
(0.25+xp(4)) (0.25+xp(3))
);
wallwood=poly2([0 0 yp(2) yp(2)],
[0 0.25+xp(3) 0.25+xp(3) 0]);
wallcel =poly2([yp(2) yp(2) yp(3) yp(3)],
[0 0.25+xp(2) 0.25+xp(2) 0]);
wallsid =poly2([yp(3) yp(3) yp(4) yp(4)],
[0 0.25 0.25 0]);

s=roofwood+roofcel+roofdeck+wallwood+wallcel+wallsid;

%3D GEOMETRY : Extrude 2D + addtional 3 blocks
blk1=extrude(s,'distance',0.25);
blk2=block3(0.25-yp(4),0.25,0.066,'pos',[yp(4),0,0.25-0.066]);
blk3=block3(0.25,0.25+xp(4),0.009,'pos',[0,0,0.25]);
blk4=block3(0.25-yp(4),0.25,0.012,'pos',[yp(4),0,0.25-0.066-0.012]);

blk=blk1+blk2+blk3+blk4;

nsub=flgeomnmr(blk);
nbnd=flgeomnbs(blk);

figure(1)
geomplot(blk,'sublabels','on','transparency',0);
xlabel('x')
ylabel('y')
zlabel('z')

figure(2)
geomplot(blk,'facelabels','on','transparency',0);

indvar=zeros(1,nbnd);
indvar([44 47 52])=1;
indvar([1 4 7 8 12 16 20 21])=3;
indvar([2 3 5 10 14 18 23 24 28 32 36 37 41 46 49 55 56 57 58 59 60])=4;
indvar([6 9 11 13 15 17 19 22 25 26 27 29 30 31 33 34 35 38 39 40 42 43 45 48 50 51 53 54])=2;

%% The Convection and Conduction Application mode is now selected:
appla.mode.class='FlConvCond';
appla.assignsuffix='_cc';
appla.equ.init=10;
appla.equ.k= {'0.16' '0.04' '0.14'};
appla.equ.rho= {'395' '55' '530'};
appla.equ.C= {'1880' '1880' '1880'};
%There are 12 sub domains use comsol GUI assign sub domains
appla.equ.ind=[3 3 3 3 3 2 2 2 1 1 1 2];

%% Define PDE coefficient velocity
appla.equ.u={'1e-5*fluxx_ht'};
appla.equ.v={'1e-5*fluxy_ht'};

%%
appla.bnd.q= {
    '8.29*(ti600fun(t)-T)' [0]
    '29.3*(tesl_roof600fun(t)-T)' [0]
} ;
appla.bnd.type= {'q' 'cont' 'q' 'q0'};
appla.bnd.ind= indvar;

%% The Convection and Diffusion Application mode is now selected:
applb.mode.class='FlConvDiff';
applb.assignsuffix='_cd';
applb.equ.init=1000;
applb.equ.D= {
    'psatf(T)*1.8e-10/(101*2.1)' 'psatf(T)*1.8e-10/(14*1.4)' 'psatf(T)*1.8e-10/(120*95)'
};
%There are 12 sub domains
applb.equ.ind=[3 3 3 3 3 2 2 2 1 1 1 2];

applb.bnd.N= {
    'psatf(T)/101*2e-8*(pi600fun(t)-c)' [0]
    'psatf(T)/120*8e-8*(pe600fun(t)-c)' [0]
} ;
applb.bnd.type= {'N' 'cont' 'N' 'N0'};
applb.bnd.ind= indvar;

%% Define PDE coefficient velocity
applb.equ.u={'1e-5*fluxx_ht'};
applb.equ.v={'1e-5*fluxy_ht'};
%%

%P (atmospheric pressure difference modeling with potential flow)
applc.mode.class='HeatTransfer';
applc.dim={'p'};
applc.assignsuffix='_ht';
applc.equ.init=0;
applc.equ.rho=1;
applc.equ.C=1;
applc.equ.k = [1e-4 '5.5' '1e-4 '];
applc.equ.ind=[3 3 3 3 2 2 2 1 1 1 2];
applc.bnd.T0= [0 [0] '2.5' [0]];
applc.bnd.type= ['T' 'cont' 'T' 'q0' ];
applc.bnd.ind= indvar;

fem.geom=blk;
fem.mesh=meshinit(fem);
fem.appl={appla applb applc};
fem=multiphysics(fem);
fem.xmesh=meshextend(fem);
fem.sol=femtime(fem,'tlist',[0:3600:48*3600]);

%%%%OUTPUT%%%%

tu=fem.sol.tlist;
ntu=length(tu);
Ap3D=[
0 0.010 (0.010+0.1118) (0.010+0.1118+0.019);
0.25 0.25 0.25 0.25
0.25 0.25 0.25 0.25
];
Atdag3D=tu/(24*3600);
ATxp3D=postinterp(fem,'T',Ap3D,'Solnum',[1:ntu]);
APxp3D=postinterp(fem,'c',Ap3D,'Solnum',[1:ntu]);
save HAM3DEX1Data Atdag3D Ap3D ATxp3D APxp3D

figure(3)
Mfilm=postmovie(fem,'tridata','T','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HAM3DEX1_T_3D')
clear mapM Mfilm

figure(4)
Mfilm=postmovie(fem,'tridata','c','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HAM3DEX1_P_3D')
clear mapM Mfilm

figure(5)
Mfilm=postmovie(fem,'tridata','p','Repeat',1);
mapM=colormap;
mpgwrite(Mfilm,mapM,'HAM3DEX1_Patm_3D')

figure(6)
postplot(fem,'tridata','T','Solnum',ntu)

figure(7)
postplot(fem,'tridata','c','Solnum',ntu)
title('Pv')

figure(8)
postplot(fem,'tridata','p','Solnum',ntu)
title('Patm')

figure(9)
postplot(fem,'arrowdata',{'fluxx_ht','fluxy_ht'})

figure(10)
postplot(fem,'tridata','k_cc')

figure(11)
postplot(fem,'tridata','D_c_cd')

figure(12)
postplot(fem,'tridata','k ht')