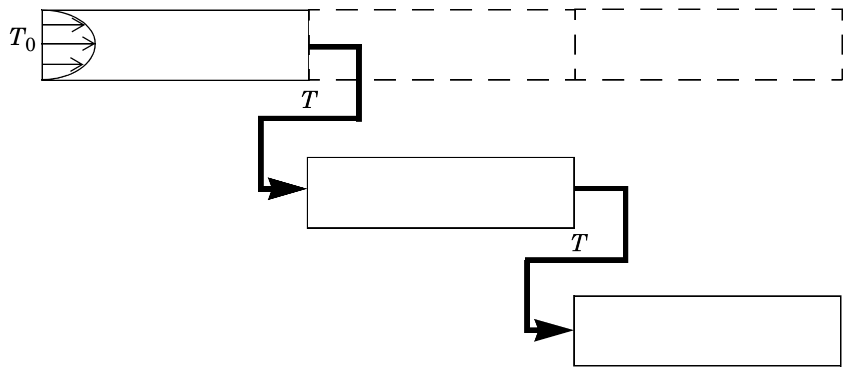
Created in COMSOL Multiphysics 6.1

Convective Heat Transfer with Pseudo-Periodicity

Introduction

This example simulates convective heat transfer in a channel filled with water. It also demonstrates a technique to reduce memory requirement, by solving the model repeatedly on a pseudo-periodic section of the channel. Each solution corresponds to a different section, and before each solution step the temperature of the outlet boundary from the previous solution is mapped to the inlet boundary.



Model Definition

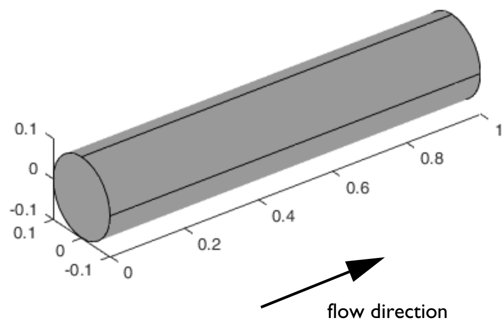
The geometry represents the inside of a 1 meter long section of a pipe. The length of the entire pipe is 6 meters.

The channel is filled with water that flows at a velocity of 1 mm/sec. To take the fluid flow into account in the heat transfer equation use a laminar velocity profile in the convective term. Such a profile can be represented as a function of the radial position according to

$$U = U_{\max}\left(1 - \left(\frac{r}{r_0}\right)^2\right)$$

The water in the channel is heated through the walls. Model this heating by applying a 100 W/m^2 heat flux to these boundaries.

For the first solution set a constant temperature of 283 K at the inlet boundary. For subsequent solutions, apply the outlet temperature distribution from the previous solution to the inlet boundary.



Implement and compare two methods for temperature evaluation at the outlet boundary:

1. In the first method you specify a point grid on the boundary. The temperature is then evaluated by interpolation on the specified grid.
2. In the second method you obtain the temperature at the mesh node points.

Results and Discussion

Figure 1 shows the temperature distribution in the model for each solution, corresponding to the 6 sections of the pipe. These can be placed together to view the temperature distribution in the entire length of the pipe. As the water flows through the pipe it reaches a maximum temperature of 294 K.

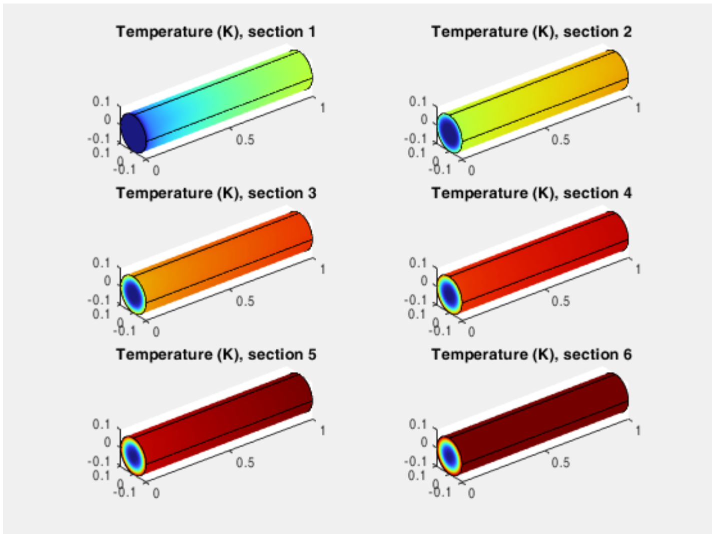


Figure 1: Temperature distribution along the pipe, the inlet is of section 1 at the left boundary.

Figure 2 shows the temperature distribution at the bottom of the pipe (edge number 5). The blue line corresponds to the outlet temperature obtained according to the interpolation method. The red line corresponds to the outlet temperature evaluated on the mesh node points. You can notice a slight difference in the solution; the red line is expected to be more accurate, because evaluation of the temperature in the mesh node points is more accurate than the interpolation on the point grid.

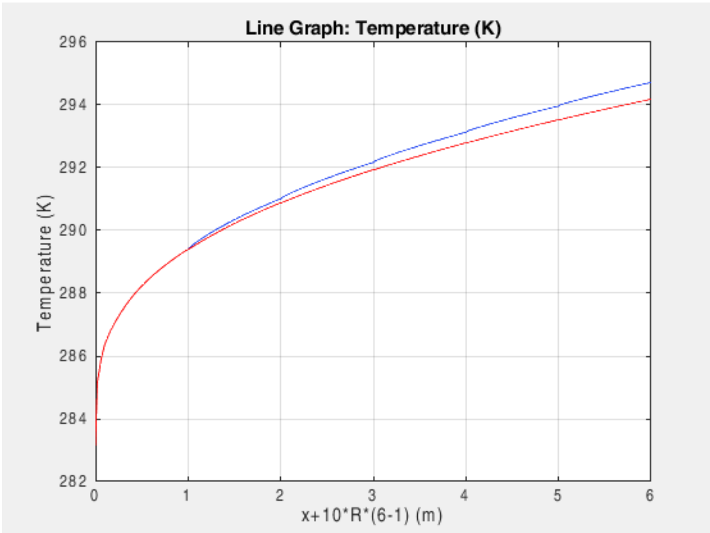


Figure 2: Temperature distribution along the bottom line of the pipe. In blue the solution using interpolated outlet temperature values, in red the solution using outlet temperature evaluated at the mesh node points.

Notes About the COMSOL Implementation

This example demonstrates how to map data from one domain to another, and from one solution to another. This involves writing data to file, then using an interpolation function to read the file and apply the data in the model.

The most efficient approach for this simulation is to start by setting up the model in the graphical user interface of the COMSOL Desktop®. You can then save the model *.mph file, which you can easily load into MATLAB®, where you continue to implement the script for solving the problem.

Wrapper functions used by the script:

- **mphopen** to load the model *.mph file.
- **mpheval** to evaluate the outlet temperature at the mesh node points.
- **mphinterp** to evaluate the outlet temperature at specified points.
- **mphplot** to display plots.

Application Library path: `LiveLink_for_MATLAB/Tutorials/pseudoperiodicity_llmatlab`

Modeling Instructions — MATLAB®

In this section you find a detailed explanation of the commands you need to enter at the MATLAB command line in order to run the simulation.

1 Start **COMSOL with MATLAB**.

You now have two possibilities to continue:

- Enter each command, starting at step 2 below, at the MATLAB command line.
- Paste the full model script, included in the section Model M-File, into a text editor, then save the file with a ".m" extension, and finally run this file in MATLAB.

2 Start by loading the model object containing the base settings of the physics model, to proceed enter the command:

```
model = mphopen('pseudoperiodicity_llmatlab');
```

Note: See the section *Modeling Instructions — COMSOL Desktop* for the modeling instruction of the model .mph file `pseudoperiodicity_llmatlab.mph`.

SOLVING WITH THE INTERPOLATED OUTLET TEMPERATURE

1 Set-up a for-loop that runs for 6 iterations:

```
for i = 1:6
```

2 The first operation to run in loop is to compute the solution:

```
model.study('std1').run;
```

3 After the computation of the first solution, the model object needs some changes in order to take into account the solution of the previous solution. Use the if statement, as indicated below, to apply these changes only once:

```
if i==1
```

4 Now add an interpolation function feature node named `inletTemp` to the model object. This reads the file pseudoperiodic_data.txt, which contains the temperature distribution at the outlet boundary. You generate the file under step 13 below. Type the following commands:

```
int1 = model.func.create('int1', 'Interpolation');
int1.model('comp1');
int1.set('source', 'file');
filename = fullfile(tempdir,'pseudoperiodic_data.txt');
int1.set('filename', filename);
int1.set('nargs', '2');
model.func('int1').setIndex('funcs', 'inletTemp', 0, 0);
```

5 Set the temperature at the inlet boundary to the interpolation function `inletTemp(x, y)`:

```
temp1 = model.physics('ht').feature('temp1');
temp1.set('T0',1,'inletTemp(y,z)');
```

6 To create a 1D plot of the temperature along edge 5 type:

```
pg1 = model.result.create('pg1', 'PlotGroup1D');
lngr1 = pg1.feature.create('lngr1', 'LineGraph');
lngr1.selection.set(5);
lngr1.set('xdata', 'expr');
```

The variable `lngr1` is a shortcut to the line graph of the `pg1` plot group in the model.

7 Now set-up a 3D surface plot of the temperature and change the color range:

```
pg2 = model.result.create('pg2', 3);
pg2.set('titleactive', 'on');
surf1 = pg2.feature.create('surf1', 'Surface');
surf1.set('expr', {'T'});
surf1.set('rangecoloractive', 'on');
surf1.set('rangecolormin', '283.15');
surf1.set('rangecolormax', '294');
```

8 Close the if statement:

```
end
```

9 Define and apply the expression for the x-axis data of the line graph `lngr1`:

```
str = sprintf('x+10*R*(%d-1)',i);
lngr1.set('xdataexpr', str);
```

10 To display plot group `pg2` enter:

```
figure(1)
subplot(3,2,i)
str = sprintf('Temperature (K), section %d',i);
pg2.set('title', str);
mphplot(model,'pg2')
```

11 Create a new MATLAB figure to display the plot group `pg1` in a separate figure:

```
figure(2)
```

```
mphplot(model,'pg1')
hold on
```

12 Next extract the temperature at the outlet boundary. Use the function `mphinterp` that requires that the coordinates for evaluation are defined in a mesh grid format. Enter the following commands:

```
x = 1;
y = -1e-1:1e-2:1e-1;
z = -1e-1:1e-2:1e-1;
[xx,yy,zz] = meshgrid(x,y,z);
coord = [xx(:),yy(:),zz(:)]';
[y0,z0,T] = mphinterp(model,{'y','z','T'},'coord',coord);
```

13 Save the temperature and the coordinate data in a file formatted for the interpolation function node `int1`, see step 4 above:

```
fid = fopen(filename,'wt');
fprintf(fid,'%%y z T\n');
for j = 1:length(T)
   if ~isnan(y0(j))||~isnan(z0(j))
      fprintf(fid,'%f %f %f\n',y0(j),z0(j),T(j));
   end
end
fclose(fid);
```

14 Now refresh the interpolation function with the newly created file:

```
model.component('comp1').func('int1').refresh;
```

15 Close the for-loop, and display a message containing the current iteration number:

```
disp(sprintf('End of iteration No.%d',i));
end
```

In case you have enter the commands above directly at the MATLAB command line, the for-loop is run and two MATLAB figures are displayed. They correspond to Figure 1 and Figure 2, respectively. At this stage the plot corresponding to Figure 2 displays only the curve based on the solution with the interpolated outlet temperature. Follow the remainder of the instructions below to solve the model again, this time evaluating the outlet temperature at the mesh node points.

SOLVING WITH THE OUTLET TEMPERATURE EVALUATED AT THE MESH NODE POINTS

1 First, re-initialize the inlet temperature condition to the constant value `T0`. Also, change the line color to red in the line graph. Type:

```
temp1.set('T0',1,'T0');
lngr1.set('linecolor', 'red');
```

2 Initialize the for-loop for 6 iterations and compute the solution:

```
for i = 1:6
   model.study('std1').run;
```

3 At the first iteration change the inlet temperature condition to `inletTemp(x,y)`:

```
   if i==1
      temp1.set('T0',1,'inletTemp(y,z)');
   end
```

4 Evaluate the temperature on the outlet, boundary 6, at the mesh node points using the `mpheval` function:

```
   data = mpheval(model,'T','edim',2,'selection',6,'refine',2);
   T = data.d1;
   y = data.p(2,:);
   z = data.p(3,:);
```

5 You can now save the data in a file:

```
   fid = fopen(filename,'w');
   fprintf(fid,'%%y z T\n');
   for j = 1:length(T)
      fprintf(fid,'%f %f %f\n',y(j),z(j),T(j));
   end
   fclose(fid);
```

6 Next, plot the plot group `pg1` in the second figure that has been generated in the previous loop (see step 11 on page 7), enter the commands below:

```
   figure(2)
   str = sprintf('x+10*R*(%d-1)',i);
   lngr1.set('xdataexpr', str);
   mphplot(model,'pg1')
```

7 Now refresh the interpolation function with the newly created file:

```
model.component('comp1').func('int1').refresh;
```

8 Finally, to close the for-loop enter:

```
   disp(sprintf('End of iteration No.%d',i));
end
```

MODEL M-FILE

Below you find the full script of the model. You can copy it and paste it into a text editor and save it with the ".m" extension. To run the script in MATLAB make sure that the path to the folder containing the script is set in MATLAB, then type the file name without the ".m" extension at the MATLAB prompt.

```
model = mphopen('pseudoperiodicity_llmatlab');
```

```
for i = 1:6
   model.study('std1').run;
   if i==1
      int1 = model.func.create('int1', 'Interpolation');
      int1.model('comp1');
      int1.set('source', 'file');
      filename = fullfile(tempdir,'pseudoperiodic_data.txt');
      int1.set('filename', filename);
      int1.set('nargs', '2');
      model.func('int1').setIndex('funcs', 'inletTemp', 0, 0);

      temp1 = model.physics('ht').feature('temp1');
      temp1.set('T0',1,'inletTemp(y,z)');

      pg1 = model.result.create('pg1', 'PlotGroup1D');
      lngr1 = pg1.feature.create('lngr1', 'LineGraph');
      lngr1.selection.set(5);
      lngr1.set('xdata', 'expr');

      pg2 = model.result.create('pg2', 3);
      pg2.set('titleactive', 'on');
      surf1 = pg2.feature.create('surf1', 'Surface');
      surf1.set('expr', {'T'});
      surf1.set('rangecoloractive', 'on');
      surf1.set('rangecolormin', '283.15');
      surf1.set('rangecolormax', '294');
   end

   str = sprintf('x+10*R*(%d-1)',i);
   lngr1.set('xdataexpr', str);

   figure(1)
   subplot(3,2,i)
   str = sprintf('Temperature (K) , section %d',i);
   pg2.set('title', str);
   mphplot(model,'pg2')

   figure(2)
   mphplot(model,'pg1')
   hold on

   x = 1;
   y = -1e-1:1e-2:1e-1;
   z = -1e-1:1e-2:1e-1;
   [xx,yy,zz] = meshgrid(x,y,z);
   coord = [xx(:),yy(:),zz(:)]';
   [y0,z0,T] = mphinterp(model,{'y','z','T'},'coord',coord);

   fid = fopen(filename,'wt');
   fprintf(fid,'%%y z T\n');
```

```
    for j = 1:length(T)
        if ~isnan(y0(j))||~isnan(z0(j))
            fprintf(fid,'%f %f %f\n',y0(j),z0(j),T(j));
        end
    end
    fclose(fid);
    model.component('comp1').func('int1').refresh;

    disp(sprintf('End of iteration No.%d',i));
end

temp1.set('T0',1,'T0');
lngr1.set('linecolor', 'red');

for i = 1:6
    model.study('std1').run;

    if i==1
        temp1.set('T0',1,'inletTemp(y,z)');
    end

    data = mpheval(model,'T','edim',2,'selection',6,'refine',2);
    T = data.d1;
    y = data.p(2,:);
    z = data.p(3,:);

    fid = fopen(filename,'w');
    fprintf(fid,'%%y z T\n');
    for j = 1:length(T)
        fprintf(fid,'%f %f %f\n',y(j),z(j),T(j));
    end
    fclose(fid);

    figure(2)
    str = sprintf('x+10*R*(%d-1)',i);
    lngr1.set('xdataexpr', str);
    mphplot(model,'pg1')
    model.component('comp1').func('int1').refresh;

    disp(sprintf('End of iteration No.%d',i));
end
```

Modeling Instructions — COMSOL Desktop

Use the COMSOL Desktop to set-up the heat transfer simulation. You can later load this example into MATLAB, using LiveLink™, to continue the model implementation.

From the **File** menu, choose **New**.

NEW

In the **New** window, click **Model Wizard**.

MODEL WIZARD

1 In the **Model Wizard** window, click **3D**.

2 In the **Select Physics** tree, select **Heat Transfer>Heat Transfer in Fluids (ht)**.

3 Click **Add**.

4 Click **Study**.

5 In the **Select Study** tree, select **General Studies>Stationary**.

6 Click **Done**.

GLOBAL DEFINITIONS

Parameters 1

1 In the **Model Builder** window, under **Global Definitions** click **Parameters 1**.

2 In the **Settings** window for **Parameters**, locate the **Parameters** section.

3 In the table, enter the following settings:

Name	Expression	Value	Description
R	10[cm]	0.1 m	Pipe radius
u0	1[cm/s]	0.01 m/s	Maximum velocity
T0	283.15[K]	283.15 K	Inlet temperature
q0	100[W/m^2]	100 W/m²	Inward heat flux

DEFINITIONS

Variables 1

1 In the **Home** toolbar, click **Variables** and choose **Local Variables**.

2 In the **Settings** window for **Variables**, locate the **Variables** section.

3 In the table, enter the following settings:

Name	Expression	Unit	Description
r	sqrt(y^2+z^2)	m	Radius parameter

GEOMETRY 1

Cylinder 1 (cyl1)

1 In the **Geometry** toolbar, click **Cylinder**.

2 In the **Settings** window for **Cylinder**, locate the **Size and Shape** section.

3 In the **Radius** text field, type `R`.

4 In the **Height** text field, type `10*R`.

5 Locate the **Axis** section. From the **Axis type** list, choose **x-axis**.

Form Union (fin)

In the **Model Builder** window, right-click **Form Union (fin)** and choose **Build Selected**.

MATERIALS

In the **Model Builder** window, under **Component 1 (comp1)** right-click **Materials** and choose **Browse Materials**.

MATERIAL BROWSER

1 In the **Material Browser** window, select **Built-in>Water, liquid** in the tree.

2 Click **Add to Component**.

3 Click **Done**.

HEAT TRANSFER IN FLUIDS (HT)

Fluid 1

1 In the **Model Builder** window, under **Component 1 (comp1)>Heat Transfer in Fluids (ht)** click **Fluid 1**.

2 In the **Settings** window for **Fluid**, locate the **Heat Convection** section.

3 Specify the **u** vector as

(1-r/R)^2*u0	x
0	y
0	z

Temperature 1

1 In the **Physics** toolbar, click **Boundaries** and choose **Temperature**.

2 Select Boundary 1 only.

3 In the **Settings** window for **Temperature**, locate the **Temperature** section.

4 In the T_0 text field, type `T0`.

Outflow 1

1 In the **Physics** toolbar, click **Boundaries** and choose **Outflow**.

2 Select Boundary 6 only.

Heat Flux 1

1 In the **Physics** toolbar, click **Boundaries** and choose **Heat Flux**.

2 Select Boundaries 2–5 only.

3 In the **Settings** window for **Heat Flux**, locate the **Heat Flux** section.

4 In the q_0 text field, type `q0`.

MESH 1

Free Triangular 1

1 In the **Mesh** toolbar, click **Boundary** and choose **Free Triangular**.

2 Select Boundary 1 only.

Distribution 1

1 Right-click **Free Triangular 1** and choose **Distribution**.

2 Select Edges 1 and 4 only.

3 In the **Settings** window for **Distribution**, locate the **Distribution** section.

4 In the **Number of elements** text field, type `15`.

Size

1 In the **Model Builder** window, under **Component 1 (comp1)>Mesh 1** click **Size**.

2 In the **Settings** window for **Size**, locate the **Element Size** section.

3 From the **Predefined** list, choose **Extremely fine**.

Swept 1

1 In the **Mesh** toolbar, click **Swept**.

2 In the **Settings** window for **Swept**, click **Build All**.

SAVE THE MODEL

1 You can now save the model in the COMSOL format, from the **File** menu select **Save**.

2 Browse to a directory which path is set in MATLAB and enter `domain_activation_llmatlab` in the **File name** text field.

3 Click **Save**.