MNIS – Physical models for micro and nanosystems

Exercise 2: Self-heating in a suspended wire

WHAT ARE WE GOING TO LEARN:

-modeling in 3D

-multiphysics modeling

-(very basic notions of) heat transfer (more to come later)

In this model, we will cover a simple but often encountered example of self-heating in wires. If you take a wire and apply a voltage drop across it, current will flow. For sufficiently high current densities, this will lead to Joule heating and result in the change of the temperature in the wire. This modifies the resistivity (which is normally temperature dependent), which modifies the current, which modifies the temperature. This feedback between temperature and current makes this problem extremely difficult to solve analytically.

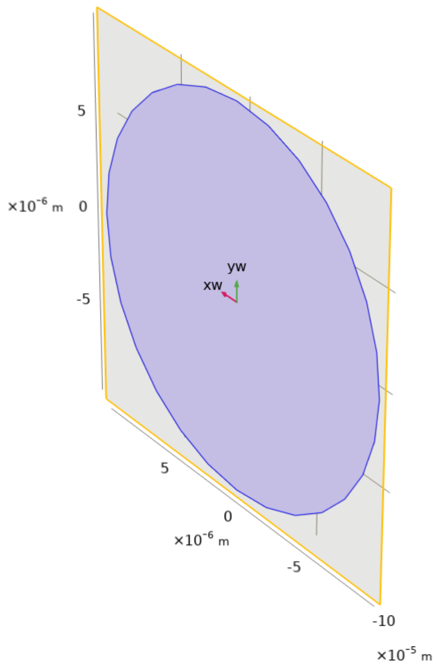
1. BUILDING THE MODEL

We will first start Comsol and define the type of geometry as **3D** and add two physics packages: **AC/DC ► electric currents (ec)** and **heat transfer ► heat transfer in solids (ht).** Go to the next section, choose **stationary study**. Choose **done.**

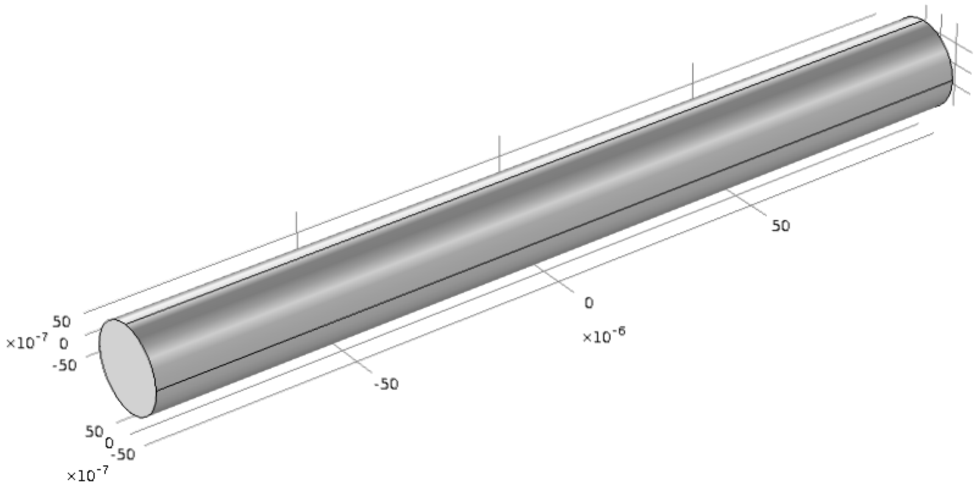
We should now build the model. It will consist of a simple cylinder with a parameter-driven geometry. First let us define the following parameters: radius=10e-6[m], length=200e-6[m], voltage=1[V] and T0 (this is the initial temperature at the start and the end of the wire) = 293[K]. A simple way to build the model would be to just go to geometry and add a cylinder but we are going to do it in a more complicated but flexible way that should be familiar and more natural if you have used Solidworks, Inventor or a similar modeling tool. We are going to first define a plane (this is just a helper for drawing), draw a circle in the plane and then extrude it into the shape of a cylinder. While this approach may seem like an overkill for this model, it is a good way for building more complicated structures for future projects.

First add a **work plane** in the **geometry1** and select **yz-plane** under **work plane ► plane.** Enter **–length/2** for **x-coordinate.** This will ensure that once we build the wire, it's midpoint will be at coordinates (0,0,0) which will make it easier to find. Now go to the plane geometry section under the **work plane** and add a circle there with radius **radius.** Now click on **work plane** in the **model builder** and **build selected.**

After doing a **zoom extends** you should end up with something looking like this:



You should now right-click on **work plane1** and select **extrude.** Enter **length** under **distances from work plane** and make sure **reverse direction** is not checked. **Build all, zoom extends** and we should now have our cylindrical wire.



We can now proceed to the next steps. First, we have to define material properties.

2. DEFINING MATERIAL PROPERTIES

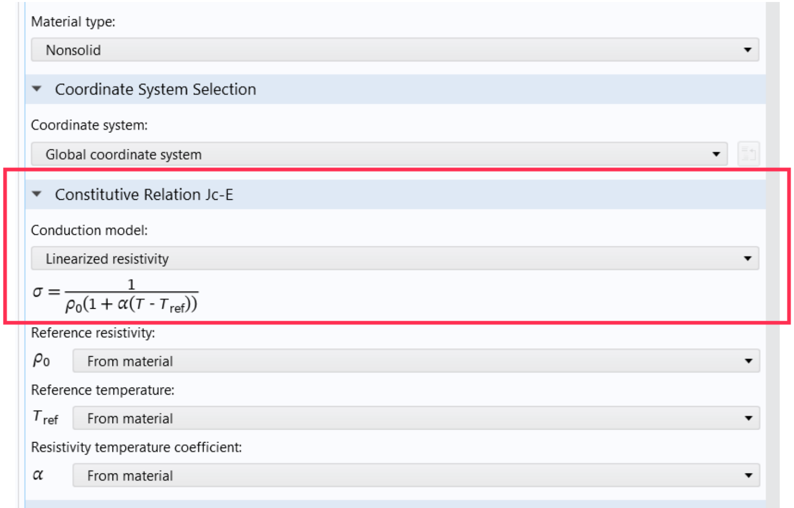
The wire will be made of copper – load the corresponding material definition into the model and assign it to the wire.

We can now move to the next step and define the boundary conditions.

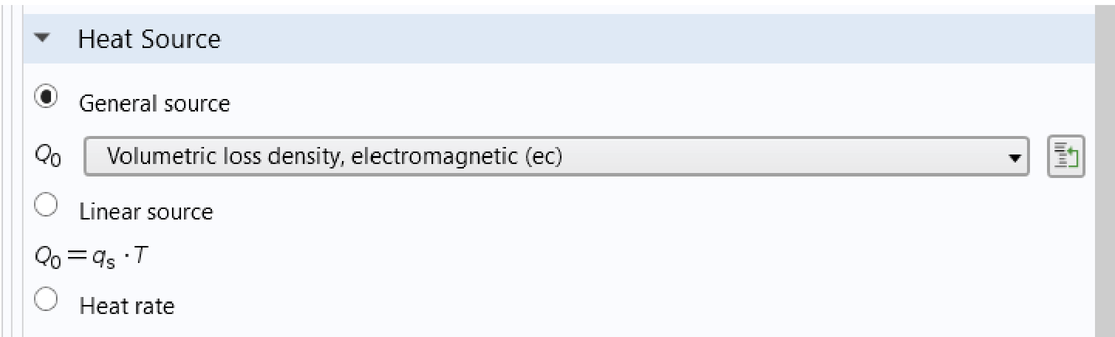
3. DEFINING BOUNDARY CONDITIONS

Here, we have to define two sets of boundary conditions, one for the electrical part of the problem, another one for heat transfer. To keep things simple, we will apply voltage to one end of the wire and assume that the wire is suspended but in good thermal contact at the ends. This means that the temperature at the ends will be T0=293K aka room temperature.

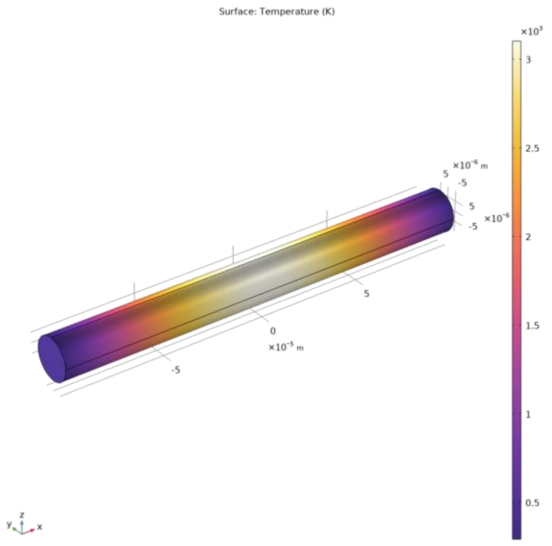
Under the **electric currents** section add **ground** and assign it to the left end of the wire. Assign **electric potential** with value V_0=**voltage** to the other end of the wire**.** Still under **electric currents**, click on **current conservation1** and select **linearized resistivity** for **electrical conductivity** under **Constitutive Relation Jc-E** in the options panel. This will ensure that Comsol will use the temperature-dependent electrical resistivity for its calculations and not a constant number (that is the default). This is what it should look like:

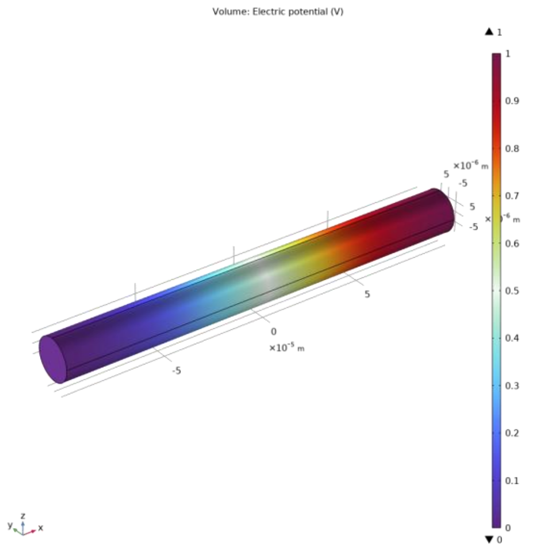


We can now turn to the heat transfer section of boundary conditions. Add a **temperature** boundary condition and set **temperature ► T_0** to **T0.** Assign this boundary condition to both ends of the wire. We now also need to specify that it is the electrical current which heats the wire. We do this by adding a domain condition **heat source** and selecting **volumetric loss density** under **heat source ► general source.** Do not forget to add the wire under the domain.



We can now start compute and if everything is OK we should get this kind of temperature and voltage distributions:

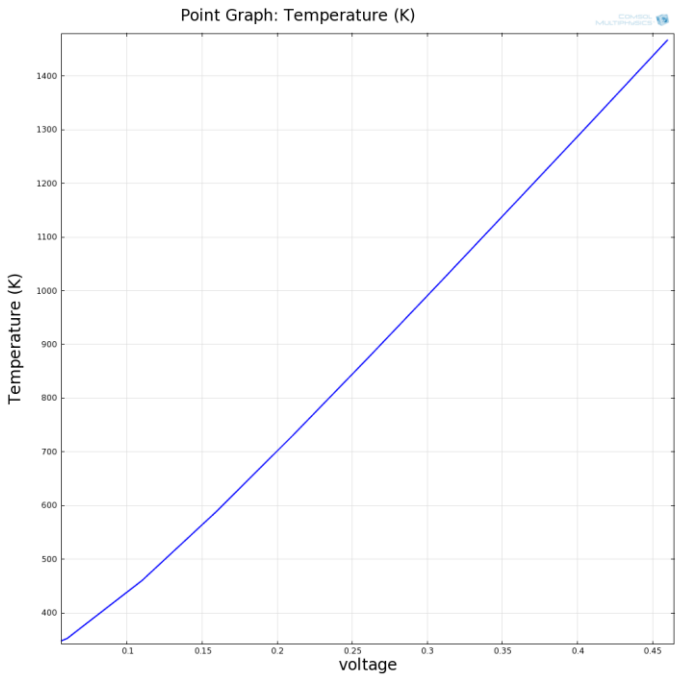




The temperature in the middle is around 3000K, which is much higher than the melting point of copper (1083.4 °C or 1356 K). Let us find out at which voltage the midpoint of wire would reach the melting point.

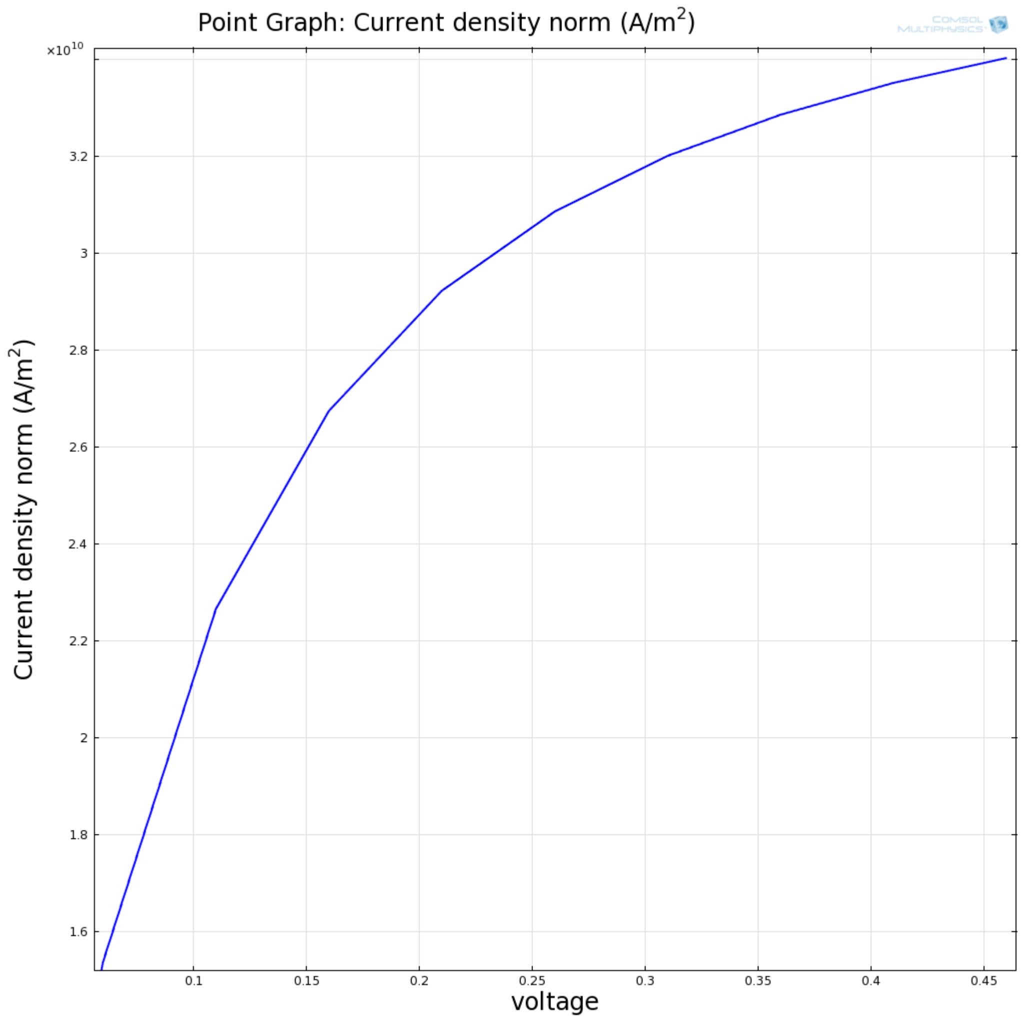
4. MODELING SELF-HEATING

To do this, we need to monitor the temperature in the middle of the wire and perform a parametric voltage sweep. Let us first add a point in the middle of the wire where we would monitor the temperature. Right click on **results ► data sets** and select **cut point 3D** with coordinates (0,0,0). Also add a new **1D plot group** with a **point graph** using the cut point as the data set. Enter **T** under **expression**. Let us rename it **temperature vs voltage.** Now add a **parametric sweep** under **study 1** with voltage as the parameter and values between 0.01 and 0.5 with a step of 0.025 and solve the model. This is what the temperature vs. voltage graph should look like:



We reach the breakdown temperature of 1356K for cca 0.425V. Let us also plot the current density (variable ec.normJ) vs voltage at this point (in another plot group).

It displays saturation at high voltages which is a characteristic signature of self-heating in this kind of geometry. This is the direct consequence of the fact that the wire resistance increases with temperature.



To check this, we can go back to the **current conservation** condition and change the α parameter to **0**. This effectively turns the temperature-dependent part of the resistivity to 0. If you repeat the calculations, you will now get a linear current-voltage dependence, as expected from the Ohm's law.