MODELLING-Representing space and time in a numerical model

Simulation parameters

When setting up any simulation, whether computer-based or to be computed manually, it is important to identify the parameters involved. This means knowing something about the physics of the problem. Then we must identify what 'space' the simulation model is running in. For example, are we modelling in time or space or both? Also, how many spatial dimensions do we require?

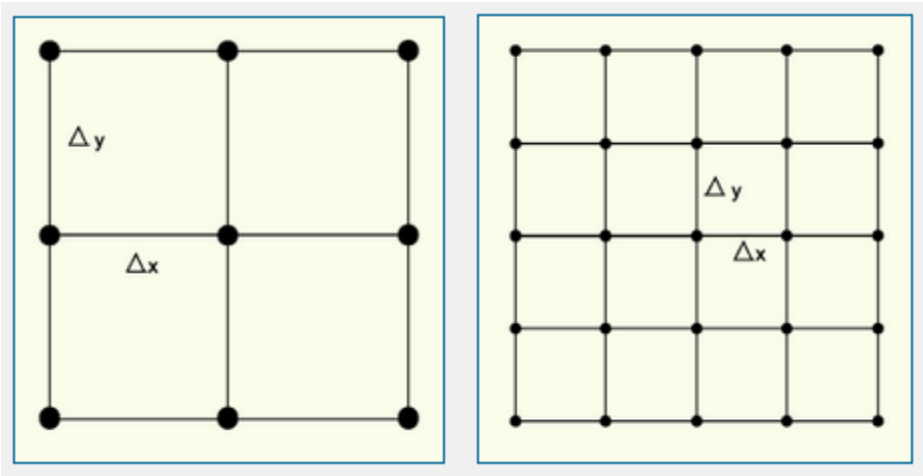
Even when we implement a model using a computer-based simulation tool, such as SPICE or FLOTHERM, we are also forced to represent the physical world using discrete steps for each of our parameters of interest. The more steps there are, the greater the number of calculation points in our model, and the slower our simulation will run – there is a price to pay for any increase in modelling detail. This is similar in many respects to the sampling period in an A-D/D-A converter, where a smaller sample period (higher sampling frequency) results in a more accurate translation of the analogue waveform.

Representing space

To represent space in a simulation model we use a grid of points, often referred to as a 'mesh'. Each point in the mesh represents a point of calculation, where we calculate the temperature or electric field strength, for example, at each iteration. The distance between each point in a three-dimensional mesh is governed by spatial steps termed Δx, Δy and Δz, and these govern the spatial resolution (the symbol Δ is used to represent the dscrete change in the parameter). Figure 1(a) shows an example of part of a two-dimensional mesh. Small values of Δx and Δy provide a higher spatial resolution than larger values, allowing us to represent more spatial detail in our model (Figure 1(b)). However, increasing the spatial resolution increases the number of calculations required at each iteration, and results in a slower-running simulation.

Figure 1(a): Part of a 2-D mesh (left)

Figure 1(b): Higher resolution 2-D mesh (right)

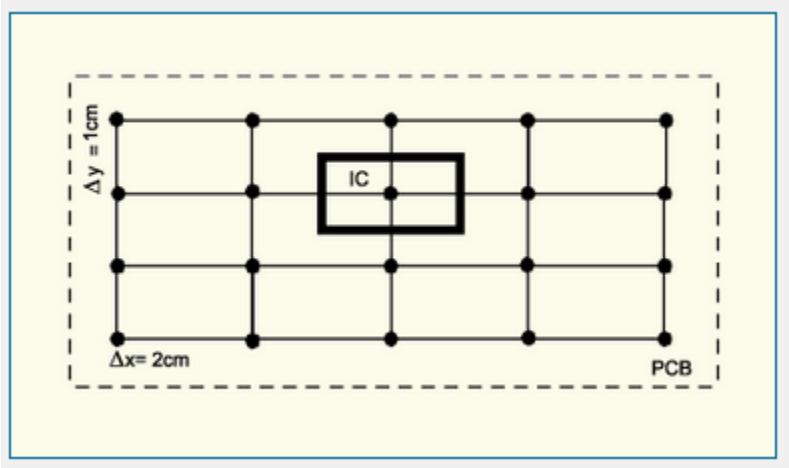


Part of a 2-D mesh Higher resolution 2-D mesh

The values of Δx, Δy and Δz required for a simulation are determined by the amount of spatial detail required. For example, if we were modelling an integrated circuit of length 2cm, width 1cm and depth 2mm, mounted on PCB surface, then the maximum appropriate spatial values in our simulation would be Δx = 2cm, Δy = 1cm, Δz = 2mm

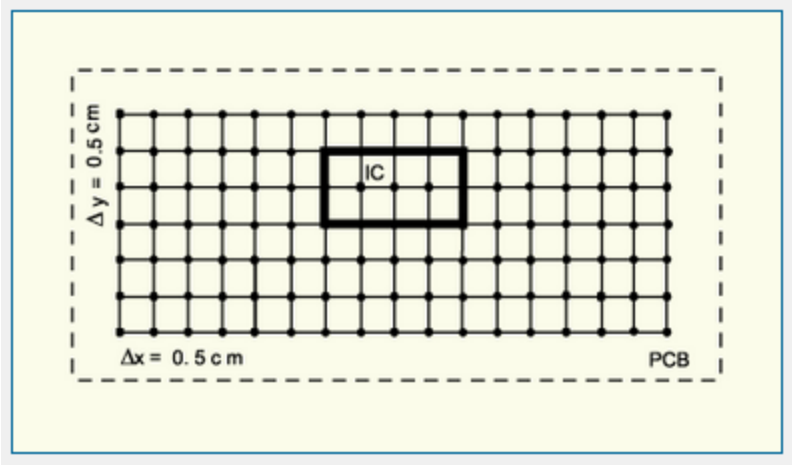
Any larger than this and it would not be possible to represent the IC within the model at all. However, if we wanted to observe how the temperature varied across the IC, we would need to reduce the values of Δx and Δy further. It is also possible to use a variable mesh, where the values of Δx, Δy and Δz are allowed to vary throughout the model space. This allows areas where there is more spatial detail to be modelled more precisely and enables the use of large values of Δx, Δy and Δz where spatial detail is not required. This approach optimises run time whilst still giving enough detail, but does make the simulation more complex to set up. Figures 2(a), 2(b) and 2(c) demonstrate three different approaches in two dimensions.

Figure 2(a): 2D view of a mesh



2D view of a mesh describing a small section of PCB and a 2cm x 1cm IC. Δx = 2cm and Δy = 1cm. Notice there is only one node describing the central temperature of the IC.

Figure 2(b): Same model of PCB and IC

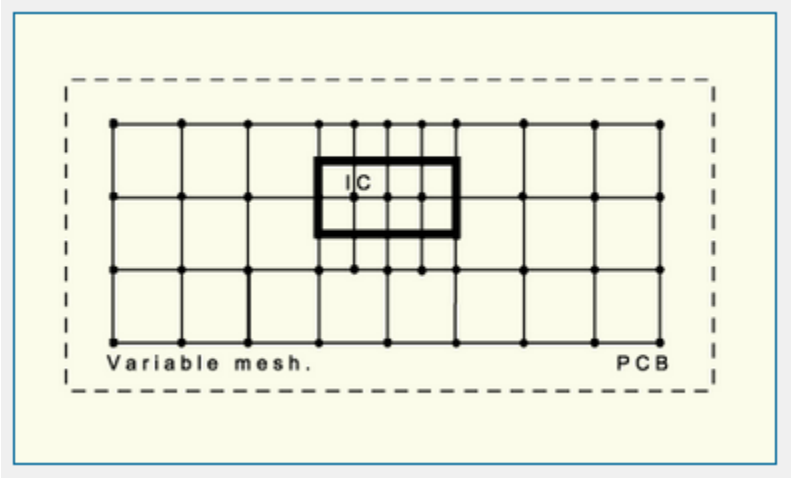


Same model of PCB and IC

Figure 2(b): Same model of PCB and IC but with a higher spatial resolution with Δx =Δy = 0.5cm.

We now can begin to model how the temperature varies across the IC at the expense of more calculation points everywhere and a slower running simulation.

Figure 2(c): Same simulation incorporating a variable mesh



Same simulation incorporating a variable mesh.

Figure 2(c): Same simulation incorporating a variable mesh. Over the area of the IC, where we need more detail about how the temperature varies we use $\Delta x = \Delta y = 0.5$cm. Elsewhere, for the rest of the PCB, where we do not need as much information, we use a lower spatial resolution, with $\Delta x = \Delta y = 1$cm.

Fitting the mesh

Which is the correct spatial resolution for a model depends upon the amount of detail required in the simulation output and how closely the model must represent the real device or system being modelled. For example, if you are modelling a component whose dimensions are 10mm x 5mm x 0.1mm, and you want to know how the temperature varies across it, then clearly your values of Dx and Dy must be lower than 10mm and 5mm respectively. But reducing the value of Dx and Dy more than is necessary, or over-engineering, will result in more computation and a slower-running simulation. It is therefore important to clearly define the amount of spatial information you require from your simulation. In some cases, a variable mesh may be applicable, involving values of Dx, Dy and Dz that vary throughout the model, so that detail can be modelled where necessary and computation saved where it is not.

When choosing values for Dx, Dy and Dz, it is also important to take into account any curved or angled surfaces, and consider how accurately they must be represented. Figure 3a shows how the angled edge of a PCB might be entered into a piece of 2-D modelling software and Figure 3b shows how the numerical model may actually see it.

Figure 3a: Model outline (left)

Figure 3b: Model interpretation (right)

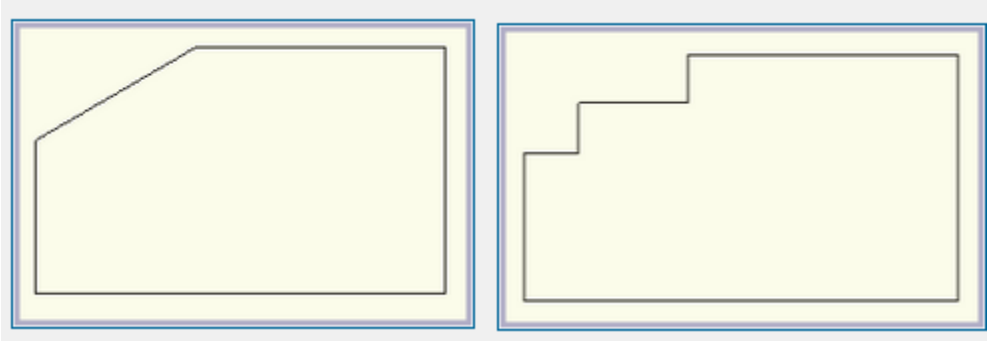
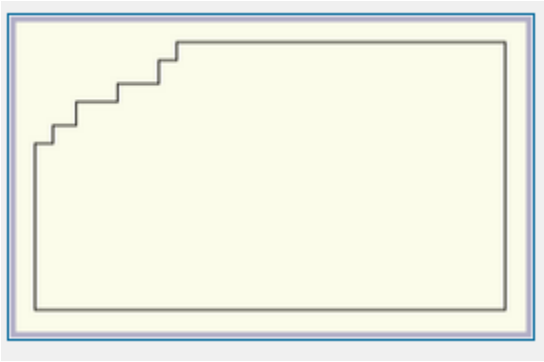


Figure 3a: Model outlineFigure 3b: Model interpretation

Because of the discrete nature of the any model, angled surfaces are always represented by some form of 'staircase' approximation. The higher the spatial resolution, the better the representation of the angled surface in the model. Figure 4 shows the same surface using a finer modelling grid.

Figure 4: Model interpretation with higher spatial resolution



tion

Simplifying the modelling task

Apart from using the largest possible mesh size that will give meaningful results, there are two main strategies for simplifying a simulation model in order to reduce the amount of computation needed. These are to model only in as many dimensions as the problem requires, and to make use of any symmetry in the physical model.

1-D, 2-D and 3-D models

The number of spatial dimensions in a model can be set according to the physical geometry of the system being modelled. Although all real systems are three dimensional, sometimes they need only be represented 1-D or 2-D with a simulation model. A model of heat flow down a thermally insulated thin copper wire could be a represented by a 1-D model, as most of the heat only flows along the length of the wire. A 3-D model is only required when there is a significant movement of energy in each of the three directions. For example, if only lateral forces in a PCB were of interest then a 2-D model would be sufficient.

Symmetry in models

In systems that are symmetric, it is sometimes possible to model one half of a system and use a reflective boundary condition to make the simulation see the effect of the other half without actually having to model it. Figure 5a shows a plan view of a PCB containing two ICs.

Figure 5a: A PCB with symmetry (left);

Figure 5b: One half of PCB entered into modelling software (right)

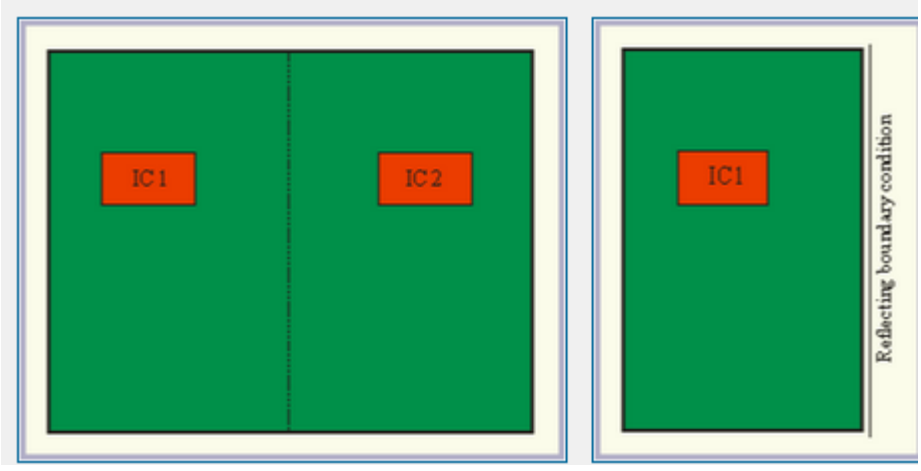


Figure 5a: A PCB with symmetryFigure 5b: One half of PCB Entered into modelling software.

Provided that everything is symmetrical, and that IC2 has the same parameters as IC1, then we need only enter one half of the model into the simulation package as in figure 5b. A reflecting boundary condition is then entered at the right hand side of figure 5b, to 'fool' the simulation package into seeing the other half, even though it is not actually there. In a 2-D model, as shown above, this technique halves the computation time involved. If there were two axes of symmetry we could quarter the simulation run time. In 3-D symmetrical systems, we could have 3 axes of symmetry, allowing us to reduce the simulation run time by a factor of 1/8th by using three reflecting boundaries.

Representing time

Another important aspect when designing a model concerns whether we are interested in the steady or final state of a system, such as its final temperature, or whether we are also interested in how it arrived this final state, for example, how rapidly a component has heated up. Deciding which of these we need greatly affects the type of model we use and the number and value of the parameters that the model requires.

Where transient information is not required, this is termed a steady-state simulation. For example, in a mechanical model of a board assembly, we may only be interested in whether the board cracks or not as it progresses through a manufacturing process, and not in how the pressure builds up beforehand to cause the damage. Steady-state models are often used to obtain a yes or no answer to a particular question, as they usually run more quickly than transient models, because the numerical time step, Dt, can often be made larger than in a transient model.

Where information about how a design progresses through to its end state is required, we use a transient model. For example, in our mechanical model of a board assembly, we could monitor how the stresses built up before the board cracked in the manufacturing process. With this information, it may be possible to re-design the PCB in some way to avoid the build-up in stress. Alternatively, it may be possible to modify the manufacturing process in some way to avoid the build-up in stress. Either way, a transient model allows us to make more detailed judgements about how to improve a design. However, because the simulation has to give us more information, a transient model uses far more computing power than a steady-state model, and usually takes substantially longer to execute.

The value of Δt chosen for a given simulation will depend on the detail of the transient information that is required and on how quickly the simulation input and outputs vary. For example, if we were modelling a device where the temperature varied very slowly, then the value of Δt could be larger than if we were modelling a device whose temperature varied quickly. Choosing an appropriate value for Δt, therefore, does partly depend upon some prior knowledge of how the system might behave and whether you are expecting fast transients or a very slow variation in output. Where there is no knowledge of this, there is an element of trial and error involved to obtain a value of Δt that will give reliable and correct results. Whatever value you choose however, it must at least concur with any stability or accuracy criteria inherent to the chosen modelling method. It is also important to realise that when any material properties are changed, Δt may have to be changed, especially if the change in material properties tends to increase the rate of any transients within the simulation.

Author: Martin Tarr

Source: http://www.ami.ac.uk/courses/topics/0200_rstnm/index.html