Partial Differential Equations I

The traditional categorization of PDEs is hyperbolic, parabolic, and elliptic, as described in your text.

$\partial^2 u/\partial t^2 = v^2\, \partial^2 u/\partial x^2$ hyperbolic wave equation (speed v)

$\partial u/\partial t = \partial/\partial x\, (D\, \partial u/\partial x)$ parabolic diffusion equation (diffusion coefficient D)

$\partial^2 u/\partial x^2 + \partial^2 u/\partial y^2 = \rho(x,y)$ elliptic Poisson equation (source ρ

The first two equations are examples of *initial value* (a.k.a. *Cauchy*) problems. If you feed your code the initial conditions for u, then the equations will predict future values of u for all later times. On the other hand, the third equation exemplifies a *boundary value* (or *static solution*) problem. Feed your code the values of u along the outer regions of a grid, then the equation provides you the values of u elsewhere in the grid.

The challenge for initial value and boundary value problems is to find a way to integrate forward in time, or to integrate spatially inwards, with an acceptable accuracy level.

Moreover, it is far more important to classify a problem as either an initial value problem or a boundary value problem (rather than classifying as hyperbolic, parabolic, elliptic), since many problems are mixtures of the three classifications.

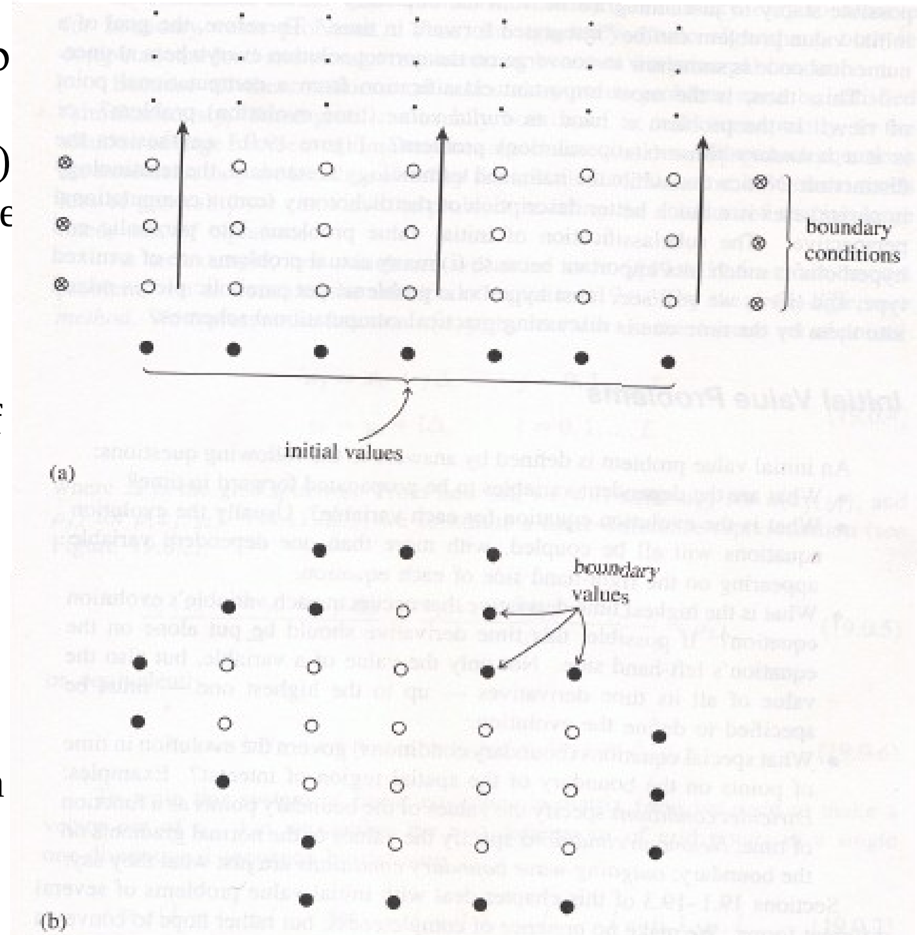


Figure 19.0.1. Initial value problem (a) and boundary value problem (b) are contrasted. In (a) initial values are given on one "time slice," and it is desired to advance the solution in time, computing successive rows of open dots in the direction shown by the arrows. Boundary conditions at the left and right edges of each row (⊗) must also be supplied, but only one row at a time. Only one, or a few, previous rows need be maintained in memory. In (b), boundary values are specified around the edge of a grid, and an iterative process is employed to find the values of all the internal points (open circles). All grid points must be maintained in memory.

from *Numerical Recipes*

Hold on a second! Take a look at Figures 6.2 and 6.4 in your text. Isn't the boundary value problem just a special case of an initial value problem? Notice that the initial value problem setup has three of four boundaries defined, while the boundary value problem simply has all four boundaries defined. So what's the big deal on differentiating between the two techniques? *Discuss with your neighbor*.

Initial Value Problems

Can the boundary conditions for an initial value problem be derivatives (and not just boundary values)?

If you claim that derivatives could be used, what kind of derivatives would they be: with respect to time or space?

Why are boundary conditions even necessary for an initial value problem? If you inspect the top figure on the previous page of these notes, or look at Figure 6.1 in the text, you'll see that the "initial values" along the bottom axes would naturally provide the "boundary values" at time t=0. Hence, wouldn't solving the problem forward in time naturally provide the boundaries for times $t > 0$? Can you think of a physical scenario in which defining the boundary values for times $t > 0$ would be necessary?

How would you envision "marching your solution forward in time" for an initial value problem? i.e., What sort of computational technique would you employ?

For an initial value problem, the main concern is in the **stability** of the solution. Because we are propagating forward in time, small inaccuracies at the beginning will be compounded as we compute forward in time. On the other hand, a boundary value problem is more concerned with the **efficiency** of the code. Since this is a static problem, we're not propagating forward in time. Instead, we are solving a large number of simultaneous algebraic equations – we can solve them by the matrix methods we've already learned, for example.

Boundary Value Problems

The finite difference method takes advantage of the grid setup for boundary value problems. We set up the grid as

$$x_j = x_0 + j\Delta \qquad j=0,1,\ldots,J$$

$$y_l = y_0 + l\Delta \qquad l=0,1,\ldots,L$$

For the elliptic problem,

$$(u_{j+1,l} - 2u_{j,l} + u_{j-1,l})/\Delta^2 + (u_{j,l+1} - 2u_{j,l} + u_{j,l-1})/\Delta^2 = \rho_{j,l}$$

or

$$u_{j+1,l} - 2u_{j,l} + u_{j-1,l} + u_{j,l+1} - 2u_{j,l} + u_{j,l-1} = \Delta^2 \rho_{j,l}$$

or

$$u_{i+L+1} + u_{i-(L+1)} + u_{i+1} + u_{i-1} - 4u_i = \Delta^2 \rho_i \qquad i \equiv j(L+1)+l$$

This equation holds for the interior. For the boundary regions, i.e.,

$j=0$ or $i=0,\ldots,L$

$j=J$ or $i=J(L+1),\ldots,J(L+1)+L$

$l=0$ or $i=0, L+1,\ldots,J(L+1)$

$l=L$ or $i=L, L+1+L,\ldots,J(L+1)+L$

the values of u or its derivative are given. In other words, this is essentially a matrix problem of the form $\boldsymbol{Au}=\boldsymbol{b}$.

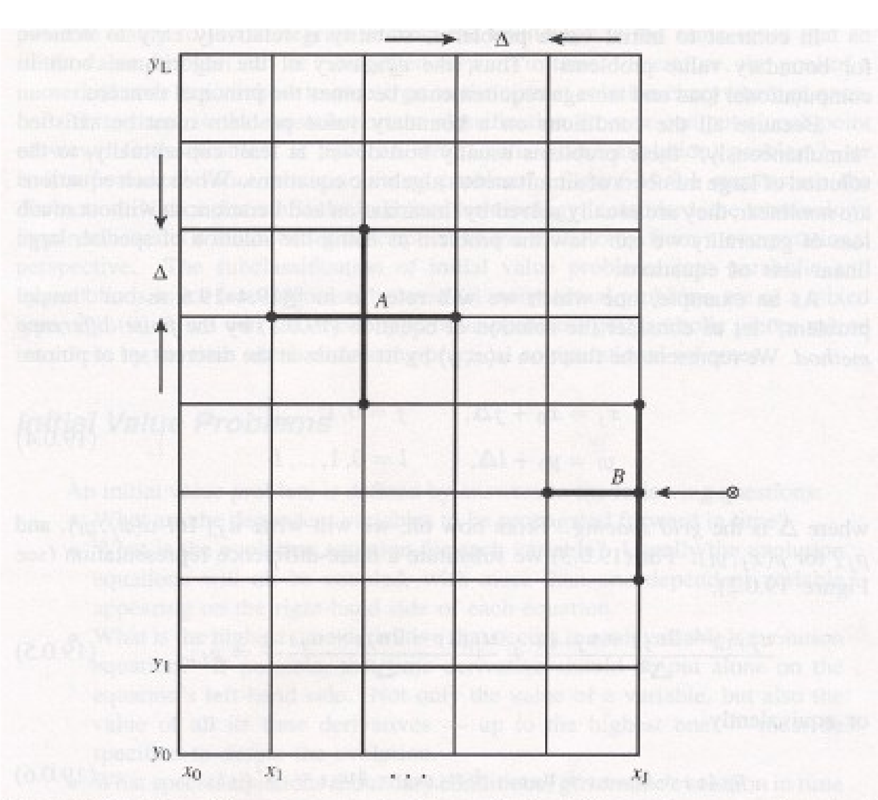
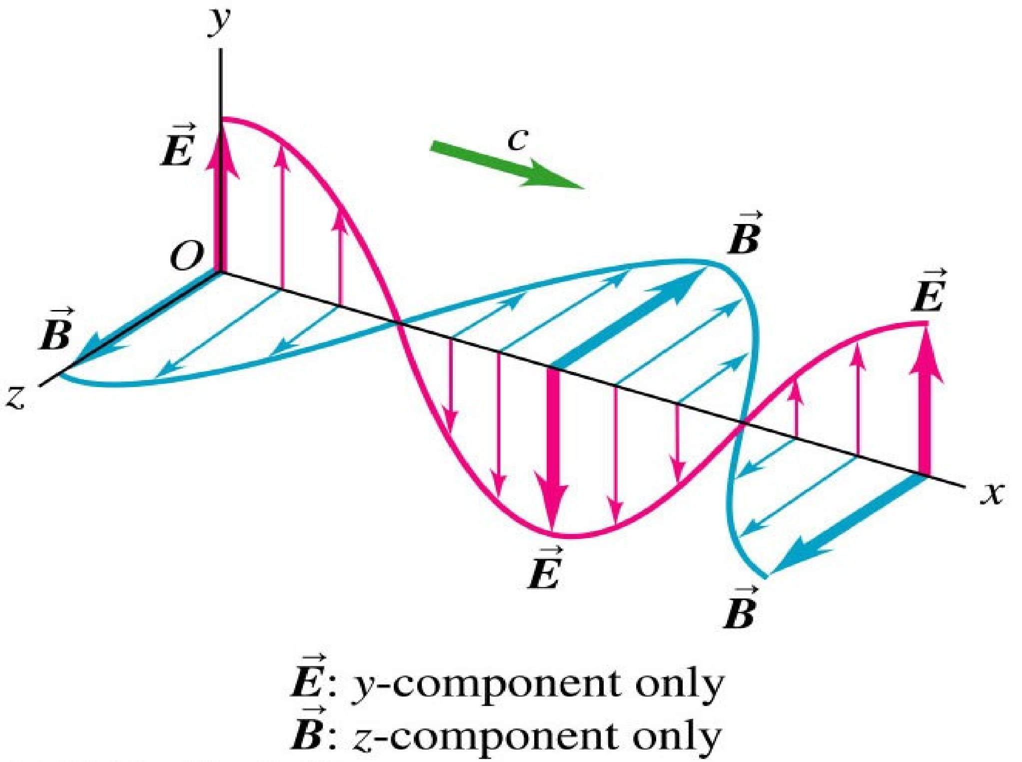


Figure 19.0.2. Finite-difference representation of a second-order elliptic equation on a two-dimensional grid. The second derivatives at the point A are evaluated using the points to which A is shown connected. The second derivatives at point B are evaluated using the connected points and also using "right-hand side" boundary information, shown schematically as ⊗.



A traveling wave

A microwave oven