

7.3: Higher Dimensions

We can work out the finite-difference equations for higher dimensions in a similar manner. In two dimensions, for example, the wavefunction $\psi(x, y)$ is described with two indices:

$$\psi_{mn} \equiv \psi(x_m, y_n). \quad (7.3.1)$$

The discretization of the derivatives is carried out in the same way, using the mid-point rule for first partial derivatives in each direction, and the three-point rule for the second partial derivative in each direction. Let us suppose that the discretization spacing is equal in both directions:

$$h = x_{m+1} - x_m = y_{n+1} - y_n. \quad (7.3.2)$$

Then, for the second derivative, the Laplacian operator

$$\nabla^2 \psi(x, y) \equiv \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \quad (7.3.3)$$

can be approximated by a **five-point rule**, which involves the value of the function at (m, n) and its four nearest neighbors:

$$\nabla^2 \psi(x_m, y_n) \approx \frac{\psi_{m+1,n} + \psi_{m,n+1} - 4\psi_{mn} + \psi_{m-1,n} + \psi_{m,n-1}}{h^2} + O(h^2). \quad (7.3.4)$$

For instance, the finite-difference equations for the 2D Schrödinger wave equation is

$$-\frac{1}{2h^2} [\psi_{m+1,n} + \psi_{m,n+1} - 4\psi_{mn} + \psi_{m-1,n} + \psi_{m,n-1}] + V_{mn}\psi_{mn} = E\psi_{mn}. \quad (7.3.5)$$

7.3.1 Matrix Reshaping

Higher-dimensional differential equations introduce one annoying complication: in order to convert between the finite-difference equation and the matrix equation, the indices have to be re-organized. For instance, the matrix form of the 2D Schrödinger wave equation should have the form

$$\sum_{\nu} H_{\mu\nu} \psi_{\nu} = E \psi_{\mu}, \quad (7.3.6)$$

where the wavefunctions are organized into a 1D array labeled by a "point index" μ . Each point index corresponds to a *pair* of "grid indices", (m, n) , representing spatial coordinates on a 2D grid. We have to be careful not to mix up the two types of indices.

We will adopt the following conversion scheme between point indices and grid indices:

$$\mu(m, n) = mN + n, \quad \text{where } m \in \{0, \dots, M-1\}, \quad n \in \{0, \dots, N-1\}. \quad (7.3.7)$$

One good thing about this conversion scheme is that Scipy provides a `reshape` function which can convert a 2D array with grid indices (m, n) into a 1D array with the point index μ :

```
>>> a = array([[0, 1, 2], [3, 4, 5], [6, 7, 8]])
>>> a
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
>>> b = reshape(a, (9))      # Reshape a into a 1D array of size 9
>>> b
array([0, 1, 2, 3, 4, 5, 6, 7, 8])
```

The `reshape` function can also convert a 1D back into the 2D array, in the right order:

```
>>> c = reshape(b, (3,3))    # Reshape b into a 2D array of size 3x3
>>> c
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

```
array([[0, 1, 2],  
       [3, 4, 5],  
       [6, 7, 8]])
```

Under point indices, the discretized derivatives take the following forms:

$$\frac{\partial \psi}{\partial x}(\vec{r}_\mu) \approx \frac{1}{2h}(\psi_{\mu+N} - \psi_{\mu-N}) \quad (7.3.8)$$

$$\frac{\partial \psi}{\partial y}(\vec{r}_\mu) \approx \frac{1}{2h}(\psi_{\mu+1} - \psi_{\mu-1}) \quad (7.3.9)$$

$$\nabla^2 \psi(\vec{r}_\mu) \approx \frac{1}{h^2}(\psi_{\mu+N} + \psi_{\mu+1} - 4\psi_\mu + \psi_{\mu-N} + \psi_{\mu-1}). \quad (7.3.10)$$

The role of boundary conditions is left as an exercise. There are now two sets of boundaries, at $m \in \{0, M-1\}$ and $n \in \{0, N-1\}$. By examining the finite-difference equations along each boundary, we can (i) assign the right discretization coordinates and (ii) modify the finite-difference matrix elements to fit the boundary conditions. The details are slightly tedious to work out, but the logic is essentially the same as in the previously-discussed 1D cases.

This page titled [7.3: Higher Dimensions](#) is shared under a [CC BY-SA 4.0](#) license and was authored, remixed, and/or curated by [Y. D. Chong](#) via [source content](#) that was edited to the style and standards of the LibreTexts platform.