

is useful for writing proofs, it is not usually an important property of a neural network implementation. Instead, many neural network libraries implement a related function called the **cross-correlation**, which is the same as convolution but without flipping the kernel:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i + m, j + n) K(m, n). \quad (9.6)$$

Many machine learning libraries implement cross-correlation but call it convolution. In this text we will follow this convention of calling both operations convolution, and specify whether we mean to flip the kernel or not in contexts where kernel flipping is relevant. In the context of machine learning, the learning algorithm will learn the appropriate values of the kernel in the appropriate place, so an algorithm based on convolution with kernel flipping will learn a kernel that is flipped relative to the kernel learned by an algorithm without the flipping. It is also rare for convolution to be used alone in machine learning; instead convolution is used simultaneously with other functions, and the combination of these functions does not commute regardless of whether the convolution operation flips its kernel or not.

See figure 9.1 for an example of convolution (without kernel flipping) applied to a 2-D tensor.

Discrete convolution can be viewed as multiplication by a matrix. However, the matrix has several entries constrained to be equal to other entries. For example, for univariate discrete convolution, each row of the matrix is constrained to be equal to the row above shifted by one element. This is known as a **Toeplitz matrix**. In two dimensions, a **doubly block circulant matrix** corresponds to convolution. In addition to these constraints that several elements be equal to each other, convolution usually corresponds to a very sparse matrix (a matrix whose entries are mostly equal to zero). This is because the kernel is usually much smaller than the input image. Any neural network algorithm that works with matrix multiplication and does not depend on specific properties of the matrix structure should work with convolution, without requiring any further changes to the neural network. Typical convolutional neural networks do make use of further specializations in order to deal with large inputs efficiently, but these are not strictly necessary from a theoretical perspective.