```
# add A and B
A + B
'Output':
array([[53, 61, 46],
       [37, 53, 72],
       [63, 61, 68]])
# subtract A from B
B - A
'Output':
array([[ 23,   3,  -2],
       [ 27,   7,  20],
       [  3,  33, -20]])
# divide A with B
A / B
'Output':
array([[ 0.39473684,  0.90625   ,  1.09090909],
       [ 0.15625   ,  0.76666667,  0.56521739],
       [ 0.90909091,  0.29787234,  1.83333333]])
```

Scalar Operation

A matrix can be acted upon by a scalar (i.e., a single numeric entity) in the same way element-wise fashion. This time the scalar operates upon each element of the matrix or vector. See Figure 10-3.

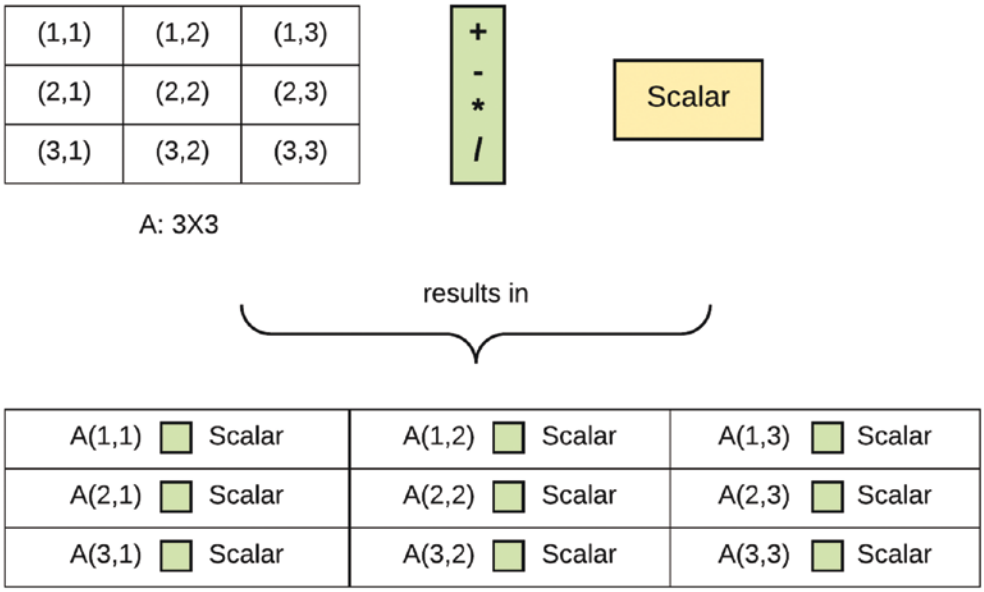

Figure 10-3. *Scalar operations*

Let's look at some examples.

```
# Hadamard multiplication of A and a scalar, 0.5
A * 0.5
'Output':
array([[  7.5,  14.5,  12. ],
       [  2.5,  11.5,  13. ],
       [ 15. ,   7. ,  22. ]])
# add A and a scalar, 0.5
A + 0.5
'Output':
array([[ 15.5,  29.5,  24.5],
       [  5.5,  23.5,  26.5],
       [ 30.5,  14.5,  44.5]])
# subtract a scalar 0.5 from B
B - 0.5
'Output':
array([[ 37.5,  31.5,  21.5],
       [ 31.5,  29.5,  45.5],
       [ 32.5,  46.5,  23.5]])
```

```
# divide A and a scalar, 0.5
A / 0.5
'Output':
array([[ 30.,  58.,  48.],
       [ 10.,  46.,  52.],
       [ 60.,  28.,  88.]])
```

Matrix Transposition

Transposition is a vital matrix operation that reverses the rows and columns of a matrix by flipping the row and column indices. The transpose of a matrix is denoted as A^T. Observe that the diagonal elements remain unchanged. See Figure 10-4.

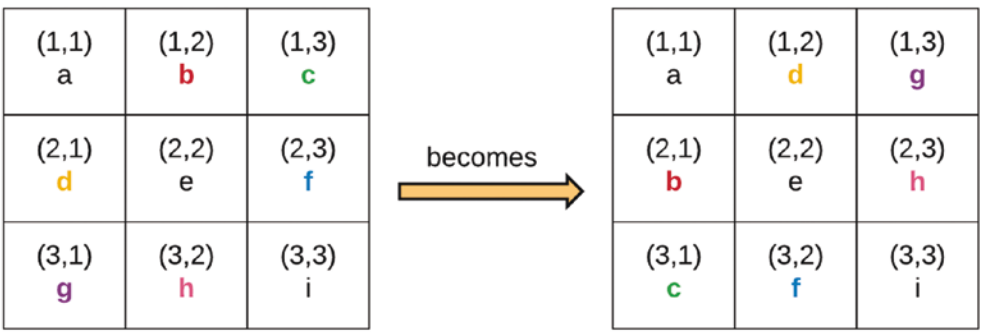

Figure 10-4. *Matrix transpose*

Let's see an example.

```
A = np.array([[15, 29, 24],
              [ 5, 23, 26],
              [30, 14, 44]])
# transpose A
A.T   # or A.transpose()
'Output':
array([[15,  5, 30],
       [29, 23, 14],
       [24, 26, 44]])
```