Vectorize/Non-vectorize Application on Matrix

GulnurUzun

2024-06-20

Vectorized Version

Create the A Matrix

First, the vector **vector_A** is created, and then using this vector, a 3x5 **matrix A** is formed, where the values are arranged row by row.

```
vector_A = c(-3.2, 0, 0.1, 2.4, 7.5,
             0.27, -4.3, -0.03, -0.07, 0,
             -9, 0, 6.2, -1, -1.2)

A = matrix(vector_A, nrow = 3, ncol = 5, byrow = TRUE)
A
```

```
##       [,1] [,2]  [,3]  [,4] [,5]
## [1,] -3.20  0.0  0.10  2.40  7.5
## [2,]  0.27 -4.3 -0.03 -0.07  0.0
## [3,] -9.00  0.0  6.20 -1.00 -1.2
```

Part 1.a

a) Counts the number of positive elements in matrix A.

At this stage, the number of positive values in the matrix is being determined. The process proceeds as follows:

1. The **which()** function is used to determine the indices of the positive values in matrix A.
2. Then, these indices are counted using the **length()** function, and the total number of positive values is assigned to the variable positive.counts.A.

positive.counts.A is 5.

```
positive.counts.A = length(A[which(A > 0)])
positive.counts.A
```

```
## [1] 5
```

Part 1.b

b) Gets the absolute value of each element in A if the value is smaller than -0.5.

At this stage, the aim is to take the absolute values of the elements in matrix A that are less than -0.5. The steps followed sequentially are as follows:

1. **which(A < -0.5):** This expression finds the indices of values in matrix A that are less than -0.5.

2. **A[which(A < -0.5)]:** This expression selects the relevant values from matrix A using the indices of values less than -0.5.
3. **abs(A[which(A < -0.5)]):** This expression takes the absolute values of the elements less than -0.5.

As a result, the variable abs.values will contain the absolute values of the elements in matrix A that are less than -0.5.

abs.values is 3.2, 9, 4.3, 1, 1.2

```
abs.values = abs(A[which(A < -0.5)])
abs.values
```

```
## [1] 3.2 9.0 4.3 1.0 1.2
```

Part 1.c

c) Replaces all zeros in A with -10.

At this stage, all elements in matrix A with a value of 0 are being replaced with -10. This operation changes the content of the matrix by substituting all 0 values with -10.

```
A[which(A == 0)] = -10
A
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]
## [1,] -3.20 -10.0  0.10  2.40   7.5
## [2,]  0.27  -4.3 -0.03 -0.07 -10.0
## [3,] -9.00 -10.0  6.20 -1.00  -1.2
```

Non-Vectorized Version

The same results as those obtained in the vectorized section were obtained here. The only difference is the use of a for loop and if statements.

Create the A Matrix

First, the vector vector_A is created, and then using this vector, a 3x5 matrix A_nonvectorized_matrix is formed, where the values are arranged row by row.

```
vector_A = c(-3.2, 0, 0.1, 2.4, 7.5,
             0.27, -4.3, -0.03, -0.07, 0,
             -9, 0, 6.2, -1, -1.2)

A_nonvectorized_matrix = matrix(vector_A, nrow = 3, ncol = 5, byrow = TRUE)
```

Part 1.a

a) Counts the number of positive elements in matrix A.

At this stage, the aim is to calculate the number of positive values within a matrix. Each element of the matrix is being checked using two loops. If the element is positive, i.e., greater than zero, the value of the variable positive_count_nonvec is incremented by one. As a result, the positive_count_nonvec variable holds the total number of positive values within the matrix. The number of positive values in our matrix is: 5

```
positive.count.nonvec = 0

for (i in 1:nrow(A_nonvectorized_matrix)) {
  for (j in 1:ncol(A_nonvectorized_matrix)) {
    if (A_nonvectorized_matrix[i, j] > 0) {
      positive.count.nonvec = positive.count.nonvec + 1
    }
  }
}
positive.count.nonvec
```

```
## [1] 5
```

Part 1.b

b) Gets the absolute value of each element in A if the value is smaller than -0.5.

At this stage, matrix elements are individually checked to find those meeting certain conditions, and their absolute values are stored in a vector.

Firstly, the dimensions of the matrix are obtained, and then an empty vector is created. Subsequently, a loop is set up for each matrix element, and if the element is **less than -0.5**, its absolute value is added to the vector. Finally, this vector is printed. This process is employed to identify matrix elements that meet specified conditions by iteratively examining them with loops.

```
# Get the dimensions of the matrix
nrows <- nrow(A_nonvectorized_matrix)
ncols <- ncol(A_nonvectorized_matrix)

# Create a temporary vector abs.values.nonvec initialized with zeros
abs.values.nonvec <- numeric(0)

# Check each element of the matrix
for (i in 1:nrows) {
  for (j in 1:ncols) {
    # If the element is less than -0.5, take its absolute value
    if (A_nonvectorized_matrix[i, j] < -0.5) {
      abs.values.nonvec <- c(abs.values.nonvec, abs(A_nonvectorized_matrix[i, j]))
    }
  }
}
print(abs.values.nonvec)
```

```
## [1] 3.2 4.3 9.0 1.0 1.2
```

Part 1.c

c) Replaces all zeros in A with -10.

At this stage, if an element in our matrix **equals zero**, its value will be **changed to -10**. This process is employed to locate zero values within the matrix and replace them with -10. In this manner, elements in the matrix that fulfill a specific condition have been altered.

```
for (i in 1:nrow(A_nonvectorized_matrix)) {
  for (j in 1:ncol(A_nonvectorized_matrix)) {
    if (A_nonvectorized_matrix[i, j] == 0) {
      A_nonvectorized_matrix[i, j] = -10
    }
  }
}
A_nonvectorized_matrix
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]
## [1,] -3.20 -10.0  0.10  2.40   7.5
## [2,]  0.27  -4.3 -0.03 -0.07 -10.0
## [3,] -9.00 -10.0  6.20 -1.00  -1.2
```