MATLAB Kursu

HAZIRLAYAN: BURAK ÖZPOYRAZ - burakozpoyraz@gmail.com

Table of Contents

DERS - 2 1
1) MATRİSLER 1
2) PROBLEM / KÜLTÜR HESABI 4

DERS - 2

1) MATRİSLER

Matrisler iki boyutlu vektörlerdir. Ancak MATLAB'de 2'den fazla boyutta da matrisler oluşturulabilir. Bir matris aşağıda verildiği gibi tanımlanabilir.

```
A1 = [1 2;
      3 4];
B = [-3 -7 4; 5 8 12];
C = [0 -4 9 7 16; -6 17 15 24 22; 2 -13 11 14 -21];

C_size = size(C);
C12 = C(1, 2);
C(2, 3) = 100;
C41 = C(4, 1);
C_sub = C(2 : 3, [1 3 5]);
C(4, 3) = -27; % What do you expect as the result?
```

İki matrisin birbirine eşit olup olmadığını kontrol etmek için normal eşitlik kontrol etme operatörü kullanılabilir. İki matrisin elemanları ayrı ayrı kontrol edilerek sonuçta yine bir matris elde edilir.

```
D = [1 2 3 4 5; 6 7 8 9 10];
E = [1 2 3 4 5; 7 8 9 10 11];
matrix_equality = D == E
```

1) Aşağıda verilen M ve N matrislerinin birbirine eşit olup olmadığını kontol ederek sonuçta eşitlik durumunda "1", diğer durumda ise "0" elde ediniz. İpucu olarak aşağıdaki adımlar verilmiştir:

1. Bir matristeki toplam eleman sayısını elde ediniz.
2. Matrislerin eşitliklerini kontrol ettikten sonra kaç tane eşit eleman olduğunu sayınız.

3. Eşit eleman sayısı toplam eleman sayısına eşit ise sonuç 1, değilse 0 olacaktır.

MATLAB'de tanımlı `sum` fonksiyonunun girdi olarak bir vektör aldığında çıktıya o vektörün elemanlarının toplamını döndürdüğünü öğrenmiştik. Ancak `sum` fonksiyonu girdi olarak bir matris aldığında çıktıya her bir sütunun toplamından oluşan bir satır vektör döndürür:

$$\text{sum}\left(A = \begin{pmatrix} 1 & 3 & 4 \\ 5 & 7 & 9 \end{pmatrix}\right) = (6 \quad 10 \quad 13)$$

Bir matristeki tüm elemanları toplayabilmek için `sum` fonksiyonu aşağıdaki gibi kullanılmalıdır:

```
sum_val = sum(A, "all");
```

Bu bilgiden yola çıkarak verilen problemi yukarıdaki adımları izleyerek aşağıda çözünüz.

```
M = [1 3 7; 5 6 9];
N = [1 3 7; 5 6 9];

num_elements = prod(size(M));
num_equal_elements = sum(M == N, "all");
result = num_elements == num_equal_elements
```

Tamamen sıfırlardan veya birlerden oluşan matrisler ile birim matris aşağıda verildiği gibi oluşturulabilir.

```
Z23 = zeros(2, 3);
O37 = ones(3, 7);
I3 = eye(3);
I34 = eye(3, 4)
```

Matrislerin transpozu daha önce vektörler için anlatıldığı şekilde elde edilebilir.

```
F = [2 4 5; 6 8 10];
F_transpose = transpose(F);
```

Matrislerin aritmetik işlemleri aşağıda verildiği gibi yapılabilir.

```
M1 = [1 2 3; 4 5 6];
M2 = [-3 -7 8; -5 3 -11];
matrix_addition = M1 + M2;
matrix_subtraction = M1 - M2;
matrix_scalar_multiplication = M1 .* M2;
M1 = 3 * M1;

M3 = [-2 15 6 17; -1 0 4 3];
M4 = [5 7 13 16 34; -3 12 14 -16 -5; 1 -18 -26 -33 28; 9 -18 -13 36 21];
matrix_multiplication = M3 * M4;

M5 = [2 4 6; 8 10 12];
```

```
V1 = [3; 5; 7];
mat_vec_multiplication = M5 * V1;

inequality_check = (M5 > 9);

M6 = [3 5; 7 9];
M6_inv = inv(M6);
M6_inv2 = M6^(-1);

M6_square = M6.^2;
```

Rastgele (random) elemanlardan oluşan vektörler ve matrisler oluşturmak için bazı hazır fonksiyonlar kullanılabilir.

```
uni_vec = rand(1, 7); % Uniformly distributed vector in the range [0 1]
uni_mat = rand(3, 5);
uni_int_vec = randi([1, 4], [1, 5]); % Uniformly distributed integers
% in the vector [1 4]
uni_int_mat = randi([1, 4], [5, 4]);
gauss_vec = randn(1, 1e5); % Gaussian distributed vector with zero mean and
% unit variance
uni_comp_vec = randn(1, 1e5) + 1i * randn(1, 1e5); % Complex Gaussian
% distributed vector with zero mean and variance 2
```

2) Aşağıda verilen [1 5] aralığında dengeli bir dağılım gösteren vektörde 3 elemanının sıklığını hesaplayınız.

```
vec = randi([1, 5], [1, 10]);
freq = length(find(vec == 3)) / length(vec);
```

3) Bir madeni parayı 3 defa attığınızda en az bir yazı gelme olasılığını hesaplayan kodu yazıp teorik değerle karşılaştırınız.

```
num_experiment = 1e5;
E = randi([0, 1], [3, num_experiment]);
P_experiment = sum(sum(E) > 0) / num_experiment;
```

```
P_theoretical = 7 / 8;
```

2) PROBLEM / KÜLTÜR HESABI

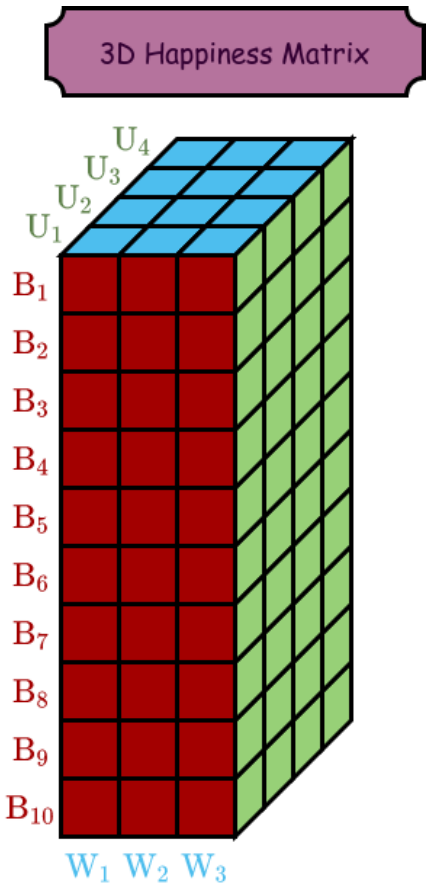
Kitap kurdu olan bir kişi elindeki kitapların hepsini bitirdikten sonra yeni bir kitap almaya karar vermiştir. Bunun için yeni alabileceği 10 adet kitap belirlemiştir. Ancak hesaplı bir kişi olduğu için internetten 3 farklı kitap satışı yapan sitede aynı kitapların fiyatlarını araştırmış ve sonucunda bir tablo yapmıştır. Tabloyu aşağıdaki gibi elde etmiştir:

KİTAP		WEBSİTE		
		W_1	W_2	W_3
	K_1	10	8	13.5
	K_2	21	33	16.5
	K_3	17	18	19.5
	K_4	6	7.5	10
	K_5	65	70	59
	K_6	45.5	43	40
	K_7	125	117	132
	K_8	14	17.5	18
	K_9	15	18	18
	K_{10}	11	10	9

Sadece fiyatlara göre alışveriş yapmak istemeyen, kullanıcıların memnun kalıp kalmadığını da araştıran kitap kurdumuz, kullanıcıların yukarıdaki kitap alışverişlerini yaptıklarında ne kadar memnun kaldıklarına dair bir tablo oluşturmak istemektedir.

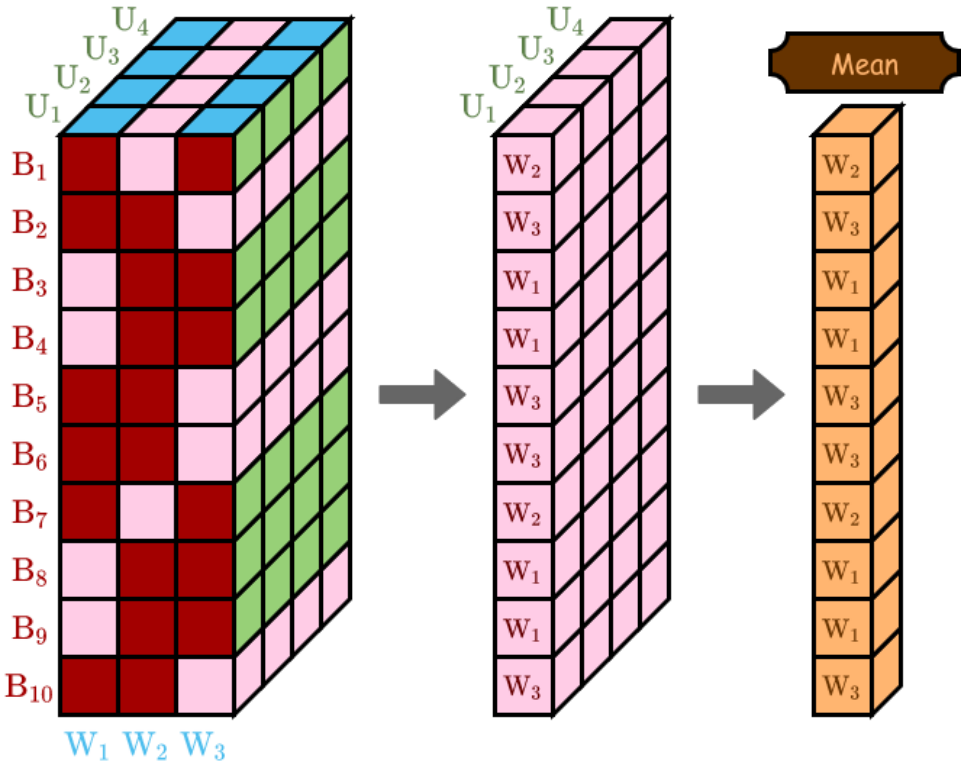
1) Kitap kurdumuzun memnuniyet tablosunu biz oluşturacağız. Puanlamanın 1'den 5'e kadar olduğu düşünülürse, 4 adet kullanıcının her bir websiteden aldığı her kitap için verdiği puanı içeren bir matrisi, rastgele olarak oluşturunuz.

Hint: Burada 3 adet değişken (kullanıcılar, kitaplar ve siteler) olduğu için 3 boyutlu bir matris oluşturulmalıdır.




```
% Happiness Matrix////////////////////////////////////
happiness_matrix = randi([1, 5], [10, 3, 4]);
% ////////////////////////////////////////////////////
```

2) Kitap kurdumuz, bu iki tabloyu (memnuniyet ve fiyat tabloları) kullanarak bazı istatistikler elde edip, hangi siteden hangi kitabı sipariş vereceğine karar verecektir. Bunun için kitapların en düşük fiyattan satıldığı sitelerden oluşan bir vektör elde etmek ve sonrasında bu kitaplardan ortalama en yüksek memnuniyet puanına sahip olanı satın almak istemektedir.




```
% Price Matrix/////////////////////////////////////////
price_matrix = [10 8 13.5; 21 33 16.5; 17 18 19.5; 6 7.5 10; 65 70 59;
                45.5 43 40; 125 117 132; 14 17.5 18; 15 18 19.5; 11 10 9];
% /////////////////////////////////////////////////////

% Books With Minimum Price//////////////////////////
% We find the indices of the websites that each book
% is sold at minimum price.
[~, min_price_ind_vec] = min(transpose(price_matrix));
% /////////////////////////////////////////////////////

% Happiness Matrix of Minimum Price Websites/////////
% We create the 2D happiness matrix (matrix at the
% middle with rose color) composed of happiness values
% from the minimum prices websites. (Hint: reshape,
% repmat, and sub2ind functions)
hp_size = size(happiness_matrix);

book_ind_vec = 1 : 10;
dim1 = reshape(repmat(book_ind_vec, [4, 1]), [1, 40]);

dim2 = reshape(repmat(min_price_ind_vec, [4, 1]), [1, 40]);

user_ind_vec = 1 : 4;
dim3 = repmat(user_ind_vec, [1, 10]);

linear_ind_vec = sub2ind(hp_size, dim1, dim2, dim3);
```

```
min_price_hp_array = happiness_matrix(linear_ind_vec);
min_price_hp_matrix = transpose(reshape(min_price_hp_array, [4, 10]));
% ////////////////////////////////////////////////////

% Mean Vector//////////////////////////////////////////
mean_vec = mean(transpose(min_price_hp_matrix));
% ////////////////////////////////////////////////////

% Book Selection From Mean Vector//////////////////////
[max_score, first_max_ind] = max(mean_vec);
max_ind_vec = find(mean_vec == max_score);

pm_size = size(price_matrix);
dim1 = max_ind_vec;
dim2 = min_price_ind_vec(max_ind_vec);
linear_ind_vec = sub2ind(pm_size, dim1, dim2);
[~, min_price_book_choice] = min(price_matrix(linear_ind_vec));
book_choice = max_ind_vec(min_price_book_choice);
website_choice = min_price_ind_vec(book_choice);
% ////////////////////////////////////////////////////
```