MATLAB Kursu

HAZIRLAYAN: BURAK ÖZPOYRAZ - burakozpoyraz@gmail.com

Table of Contents

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
```

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.

Ekstra olarak ***sum*** fonksiyonunun aşağıdaki özelliği verilmiştir:

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

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

size_M = size(M);
num_elements = prod(size_M);
num_equal_elements = sum(sum(M == N));
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
```

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);
```

Bir madeni parayı 3 defa attığınızda en az bir yazı gelme olasılığını hesaplayan kodu yazınız.

```
num_experiment = 1e5;
E = randi([0, 1], [3, num_experiment]);
[row_indices, col_indices] = find(E == 1);
P_experiment = length(unique(col_indices)) / num_experiment;
```

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

2) PROBLEM / KÜLTÜR HESABI

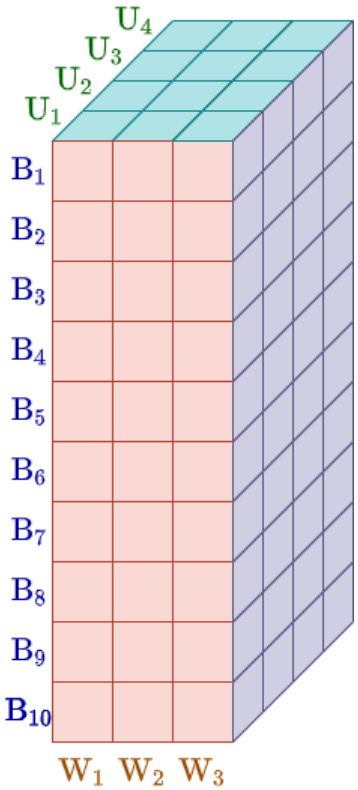
Kitap kurdu olan bir kişi elindeki kitapların hepsini bitirdikten sonra yeni bir kitap almaya karar vermiştir. Bunun için yeni alabileceği 10 adet kitap belirlemiştir. Ancak hesaplı bir kişi olduğu için internetten 3 farklı kitap satışı yapan sitede aynı kitapların fiyatlarını araştırmış ve sonucunda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

		Website		
		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
Kitap	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.

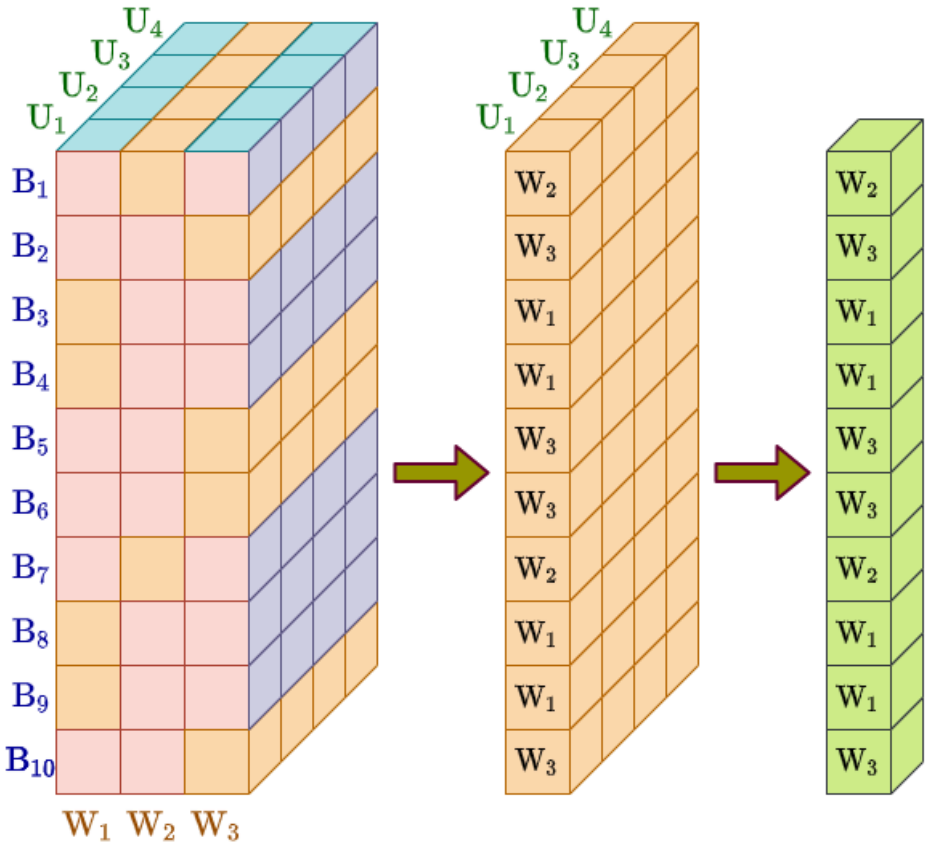
Hint: Burada 3 adet değişken (kullanıcılar, kitaplar ve siteler) olduğu için 3 boyutlu bir matris oluşturulmalıdır.




```
% Happiness Box/////////////////////////////////////
```

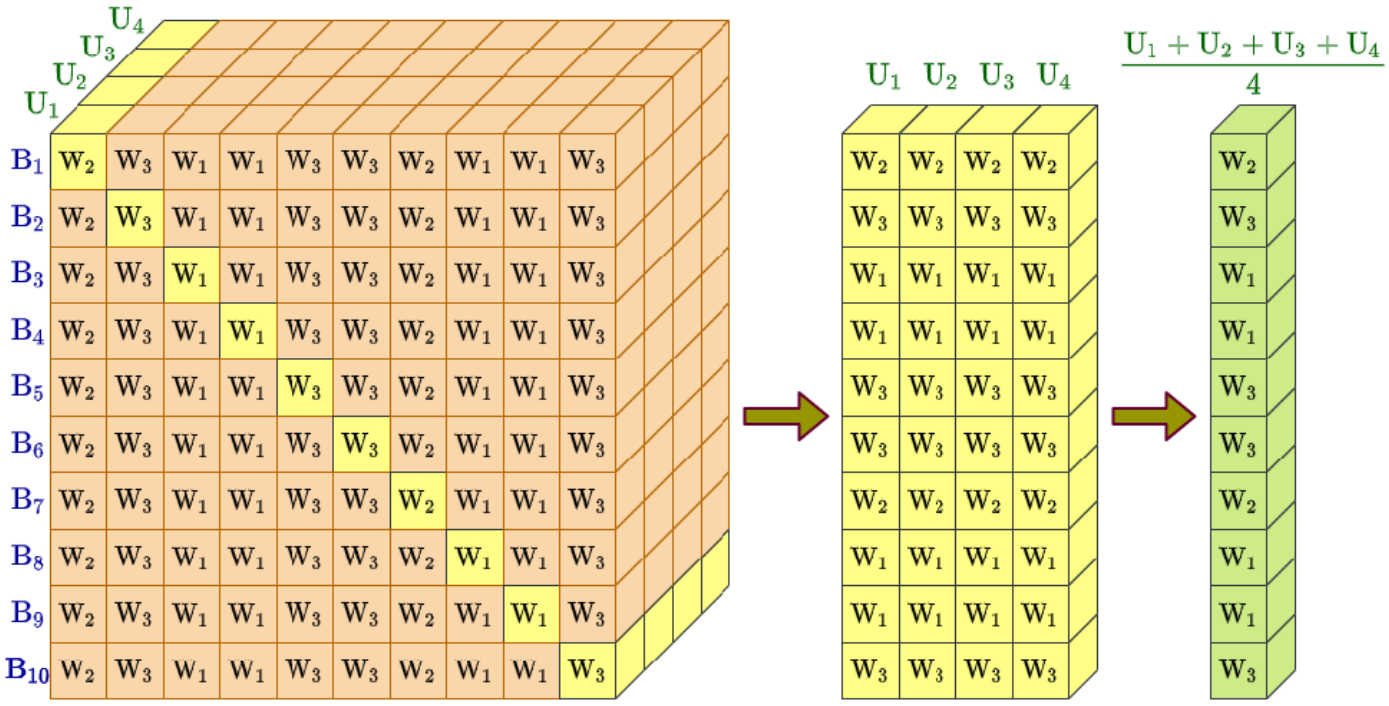
```
happiness_matrix = randi([1, 5], [10, 3, 4]);
% ////////////////////////////////////////////////////
```

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 öncelikle her kitabın en düşük satış fiyatından oluşan bir vektör elde etmek ve sonrasında bu kitaplardan hangisinin ortalama en yüksek memnuniyet puanına sahip olduğunu tespit etmek istemiştir.




```
% 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//////////////////////////////
% Here, we find the indices of the websites that each
% book is sold at minimum price.
[~, min_price_ind_vec] = min(transpose(price_matrix));
% ////////////////////////////////////////////////////
```




```
% Yellow Matrix//////////////////////////////////////
% happiness_matrix(:, min_price_ind_vec, i) -> This
% code creates 10x10 matrix by taking all rows and the
% columns in min_price_ind_vec from happiness matrix
% of user i. This means that first column of the
% output matrix is the second column of happiness
% matrix of user i (min_price_ind_vec(1) = 2). The
% second column of the output matrix is the third
% column of happiness matrix of user i
% (min_price_ind_vec(2) = 3). Thus, this code creates
% the front face of tha box above when i = 1. Each
% element on the left diagonal represents the index of
% the website that sells the book at the minimum
% price.

% diag(happiness_matrix(:, min_price_ind_vec, i)) ->
% This code creates a vector containing the elements
% in the left diagonal of the ith face of the box
% above.
yellow_matrix = [diag(happiness_matrix(:, min_price_ind_vec, 1))...
                 diag(happiness_matrix(:, min_price_ind_vec, 2))...
                 diag(happiness_matrix(:, min_price_ind_vec, 3))...
                 diag(happiness_matrix(:, min_price_ind_vec, 4))];
% //////////////////////////////////////////////////

% Green Vector/////////////////////////////////////////
green_vector = mean(transpose(yellow_matrix));
% //////////////////////////////////////////////////

% Book Selection From Green Vector/////////////////////
% If there are more than one book with the same
```

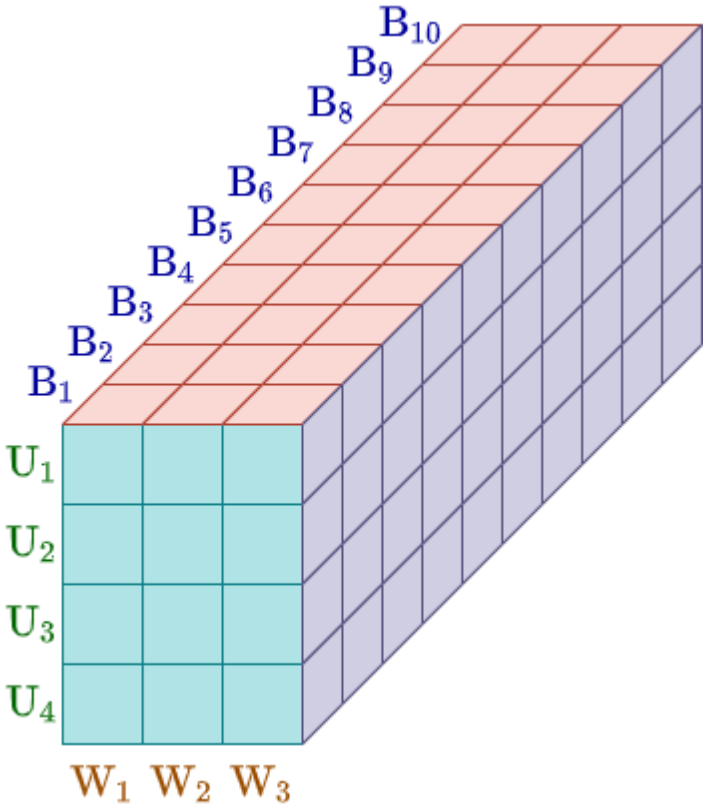
```
% maximum average happiness score, max function
% provides only the first index.
[first_max, first_max_ind] = max(green_vector);

% Then, we take all the indices that have the maximum
% average happiness score.
all_max_indices = find(green_vector == first_max);

% Since they all have the same average happiness score
% , we take the one with the minimum price.
[~, min_price_ind] = min(price_matrix(all_max_indices,...
    min_price_ind_vec(all_max_indices)));

book_choice1 = all_max_indices(min_price_ind);
website_choice1 = min_price_ind_vec(book_choice1);
% ////////////////////////////////////////////////////
```

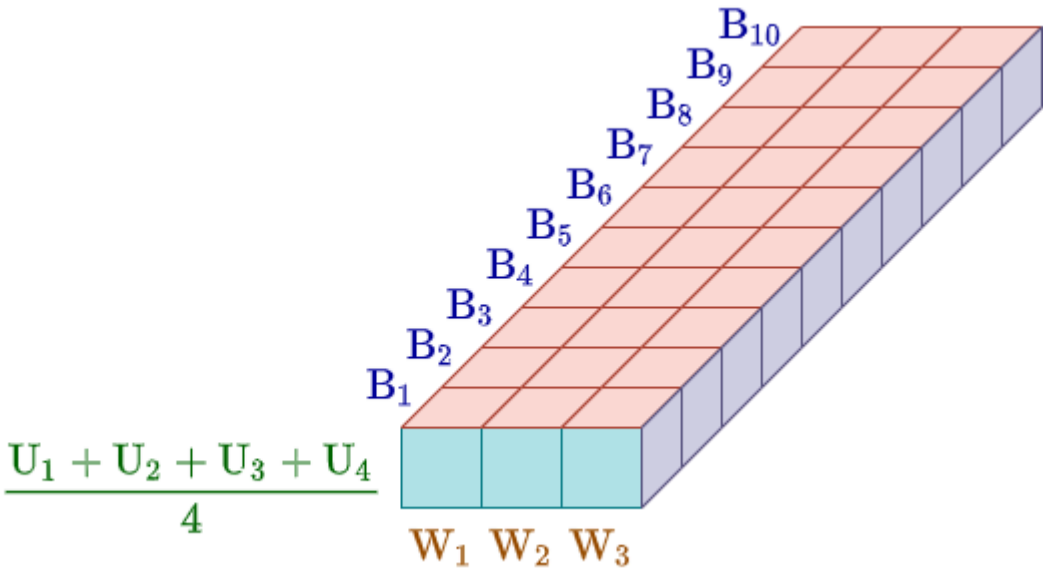
3) Kitap kurdumuz başka bir şekilde daha hesap yaparak en verimli seçeneği bulmak istemektedir. Bunun için şimdi de her bir kitabın her bir siteden satışı sonrası kullanıcıların memnuniyet puanlarının ortalamasını almak istemektedir.



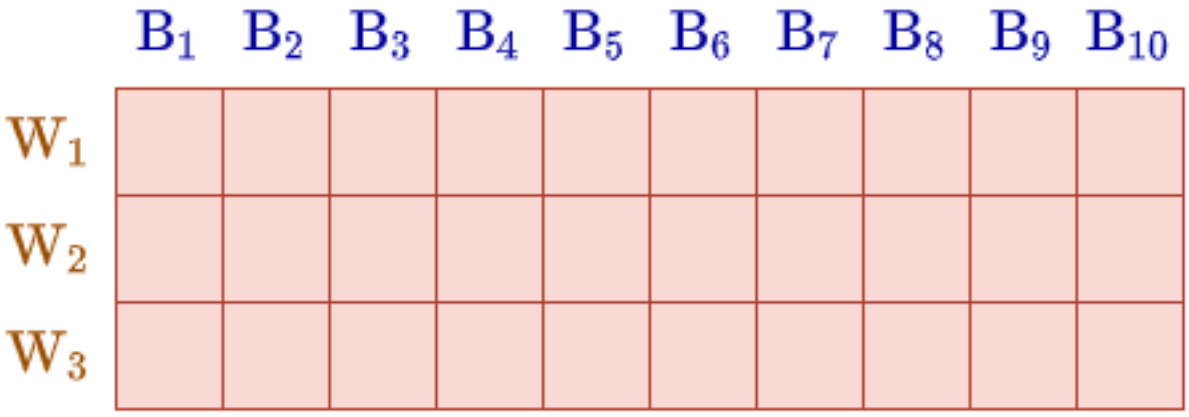

```
% Reshaped Box/////////////////////////////////////////
happiness_matrix_permute = permute(happiness_matrix, [3 2 1]);
% ////////////////////////////////////////////////////

% Average Happiness Scores//////////////////////////
% The figure below is obtained by the following code.
mean_happiness_matrix = mean(happiness_matrix_permute);
```

```
% ////////////////////////////////////////////////////
```




```
% Dimension Reduction//////////////////////////////////
% The figure below is obtained by the following code.
mean_happiness_matrix = squeeze(mean_happiness_matrix);
% ////////////////////////////////////////////////////
```




```
% Reshape Matrix//////////////////////////////////////
mean_happiness_matrix = transpose(mean_happiness_matrix);
% ////////////////////////////////////////////////////
```

4) Memnuniyet ortalamalarını bulduğuna göre elinde her kitabın her bir siteden satışına dair ortalama memnuniyet puanını içeren bir tablo ve bir de ücret bilgisini içeren ikinci bir tablo bulunmaktadır. Düşük ücretli ve maksimum ortalama memnuniyet puanına sahip seçeneği bulmak istediği için aşağıdaki gibi bir skor tanımlamıştır:

$$S = \frac{\text{Memnuniyet Puani}}{\text{Ücret}}$$

Buna göre her kitabın her bir siteden satışına dair skoru hesaplamak ve en yüksek skora ait seçeneği bulmak istemektedir.

```
% Score Matrix//////////////////////////////////////
score_matrix = mean_happiness_matrix ./ price_matrix;
% //////////////////////////////////////////////////

% Book Selection From Score Matrix//////////////////
[max_val_vec, max_row_ind_vec] = max(score_matrix);
[~, max_col_ind] = max(max_val_vec);

book_choice2 = max_row_ind_vec(max_col_ind);
website_choice2 = max_col_ind;
% //////////////////////////////////////////////////
```