Problem 5.1.5

Evan Gassiot

1 Problem

The matrix from `rectangle.jpg` is not square. We can, however, still apply matrices to shift the pixels. Find a matrix that will perform a horizontal shift of 306 pixels to `rectangle.jpg` and include the shifted image in your write up. Hint: We saw with n $\times$ n matrices that to perform a horizontal shift we multiply our matrix by a transformation matrix on the right. The transformation matrix on the right was obtained by transforming the columns of the n $\times$ n identity matrix in the same way we wanted the columns of the image matrix to be transformed. For a non-square matrix $\mathbf{X}$, we can take the same approach, but we have to start with the correct identity matrix. Think about the dimensions of the matrix you want to transform and find the matrix $\mathbf{I_R}$ such that $\mathbf{XI_R} =$ X. Manipulate the columns of $\mathbf{I_R}$ to obtain the transformation matrix. Display your matrix using `spy()`.

2 Solution

In order to apply a shift to an image matrix we will again start out with the $\mathbf{I}$ matrix as it allows us to solely focus on the order of the 1's. We will use a smaller example to order to convey the idea for a much larger example with an actual image. We will use the matrix $\mathbf{A}$ as our example here. We will let $\mathbf{A}$:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}$$

We are going to attempt to alter $\mathbf{A}$ so we get:

$$\mathbf{A} = \begin{bmatrix} 4 & 5 & 1 & 2 & 3 \\ 4 & 5 & 1 & 2 & 3 \\ 4 & 5 & 1 & 2 & 3 \end{bmatrix}$$

This matrix has had its columns shifted 2 to the right meaning our $\mathbf{T}$ matrix should also have this shift in it. We also need to figure out what size we need to make our transformation matrix $\mathbf{T}$. First off we need to figure out the size of the $\mathbf{T}$ matrix. Our current matrix is a size 3×5 but we will generalize this so we can better understand it for the MatLab coding later on.

$$size(\mathbf{A}) = m \times n$$

The size of the matrix we multiply in must be the same size as the output in order for them to be equivalent. Also the number of columns of **A** must be equal to the number of rows in matrix **T**. Therefore the solution for the size of the transformation matrix must be:

$$size(\mathbf{A} \times \mathbf{T}_{\mathrm{n}}) = (m \times n)(n \times n) = m \times n$$

Going back to our example our **T** matrix must be of size 5×5. We will now build our **T** matrix but we will shift it's columns 2 to the right and loop the 4^{th} and 5^{th} columns back to the front of the matrix so we are left with:

$$\mathbf{T} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

If we now compute the product of **AT** our result is:

$$\mathbf{AT} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 4 & 5 & 1 & 2 & 3 \\ 4 & 5 & 1 & 2 & 3 \\ 4 & 5 & 1 & 2 & 3 \end{bmatrix}$$

We now have found a solution for our shifted matrix. We can generalized the shifted matrix to be **I** with its columns shifted **r** times to the right making sure to loop the columns back around to the left hand side when they 'fall off'.

When we apply what we learned from our example to our `rectangle.jpg` with an **r** value of 306`px` we get:

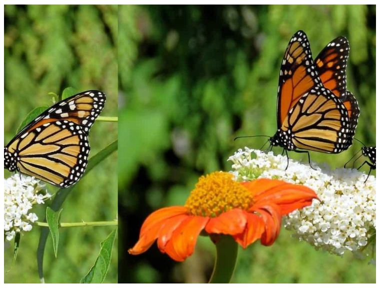

Figure 1: Image shifted 306 pixels to the right

From this we learned that we can shift images horizontally by shifting the identity matrix columns and using RHS multiplication we could produce an image with a vertical pixel shift by using a similar **I** matrix but it's size would need to be $(m \times m)$ and we would then apply LHS multiplication to receive a vertically shifted image.

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix}$$

$$size(\mathbf{T}) = (m \times m)$$

$$\mathbf{TB} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix} = \begin{bmatrix} 3 & 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix}$$

This is a vertical shift of our **A** matrix with an **r** value of 2. This shows that it is also very simple to do a vertical translation. Both of these matricies could be combined into making an image that is both shifted vertically and horizontally at the same time.

3 MatLab Code

```matlab
A = imread('rectangle.jpg'); %% Read the image
X_double = double(A); %% Convert to double
[m, n, c] = size(A); %% Get the size data from the image
r = 306; %% # of pixels to shift
E = eye(n); %% Identity
T = zeros(n);

T(:,1:r) = E(:,n-(r-1):n); %% For columns 1 to r replace them with
    the identity columns from the (n - r - 1)th column to the last
    column
T(:,r+1:n) = E(:,1:n-r); %% From column r + 1 to the last column
    replace it with the identity from 1 to n - r


X_double(:,:,1) = X_double(:,:,1) * T; %% Modify the image matrix
    to have the shifted pixels
X_double(:,:,2) = X_double(:,:,2) * T;
X_double(:,:,3) = X_double(:,:,3) * T;

figure();
hold on
imagesc(uint8(X_double)) %% Print the image
hold off
```