Color Normaliser

November 1, 2019

1 Homework 9 - Q2

1.1 Problem statement:

> Create a 2D matrix T of dimensions (100 x 100) with random numbers. Apply the computation represented by the following pseudo code – say 50 times – and see what does it do to the array. Visualize using pcolor plot and comment on what you observed.

```
counter = 1
Step-1:
For each i, j (skipping the ones on the boundary):
Tnew(i,j) = T(i,j) + 0.2*(T(i+1,j) + T(i-1,j) + T(i,j+1) + T(i,j-1) - 4*T(i,j))
Step-2:
T = Tnew
Step-3:
pcolor(T)
counter++;
Go back to Step-1 till the counter reaches 50.
```

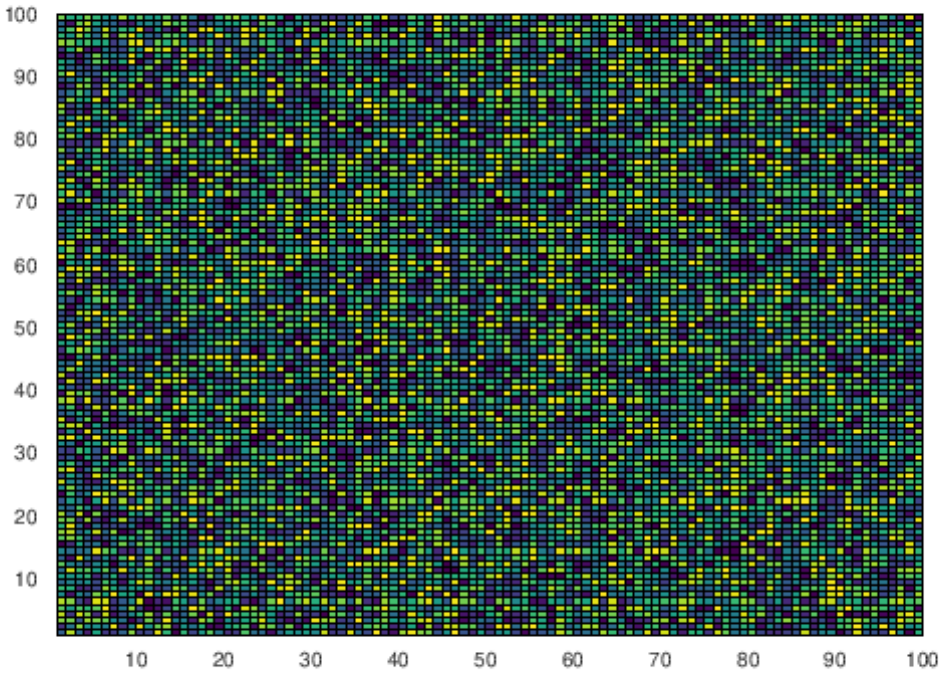
Here, we implement an algorithm to normalize the difference between random points, and then see the final image

In the cell below, we create a random 100×100 matrix, which is displayed using the `pcolor` command of Octave.

(I've suppressed the output using the `;` as it isn't very pleasing to see a 100×100 matrix :)

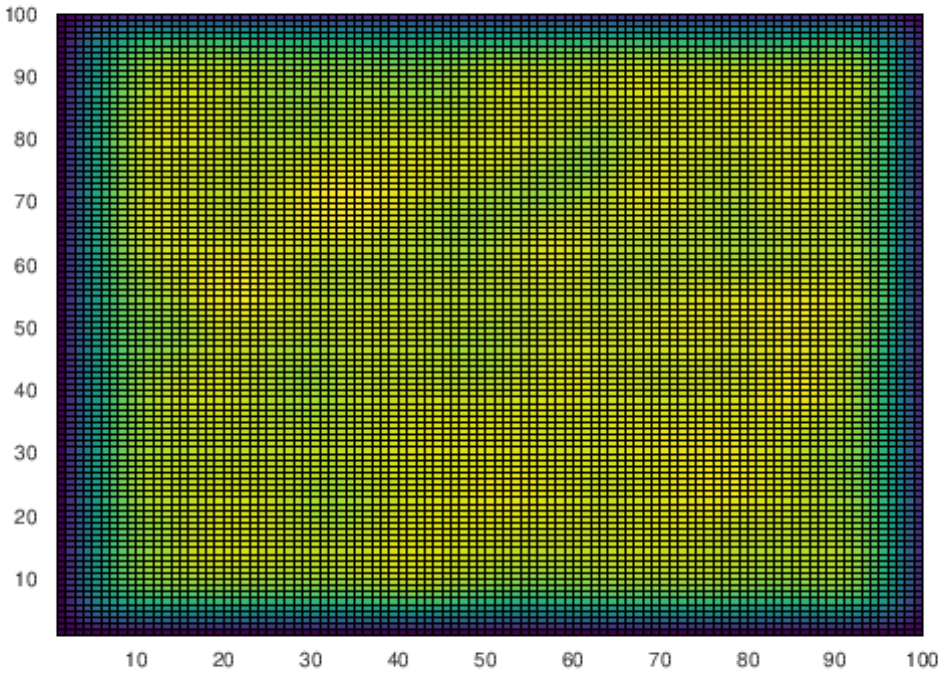
[1]:
```
A = rand(100);
B = zeros(100);

pcolor(A)
```



In the cell below, we perform the task as mentioned in the question, and obtain the following resultant matrix:

[2]:
```
for iterator = 1:50
    for i = 2:size(A,1)-1
        for j = 2:size(A,2)-1
            B(i,j) = A(i,j) + 0.2*(A(i+1,j) + A(i-1,j) + A(i,j+1) + A(i,j-1) -␣
 ↪4*A(i,j));
        end;
    end;
    A = B;
    pcolor(A)
end;
```



This code takes a little while to run, but the final output is dark at the edges, and bright at the centre.

Thus, the code was able to reduce the differences of values of the points, and we finally obtained a homogenous arrangement, distributed about the centre.