**Digital Image Processing**

**HW2**

**Blind Super Resolution**

Yannay Alon 322227216  
Tomer Gavriel 322230392

**Part 1 - Theory**

Question 1

From definition of convolution using the notations from the third paragraph:

Question 2

From down sampling and convolution definition:

Question 3

By plugging in the results from q1 and q2 we get:  
By change of variables :By another change of variables :  
Since this holds for all , we have:  
Therefore,

Question 4

By q3:  
Using Fourier transform:

Substituting we have

Question 5

For we will get:

However, for we have but,   
Therefore, this assumption does not hold.

For .

Where is since the integrand is a distribution probability.

Even-though this is a gaussian, it is not equal to . For example, let , we get:

In real life, the gaussian PSF is more likely, this filter arises naturally in real life scenarios unlike the ideal rectangular filter.

Question 6

The distribution of given the corresponding patch is:  
Moreover,  
The ML for is given by the patches :  
Question 7

Assuming the kerel is fairly smooth, we can use a Laplacian operator to achieve the requested distribution. Due to the smoothness of the kernel, its second derivatives are approximately zero (up to some noise). The MAP is given according to:

Question 8

Note that the convolution with the down-sampled is a linear operation. We can denote , therefore,

To iteratively solve this problem, we denote .  
Plugging it back in:  
Therefore, solving our problem is equivalent to solving a weighted least squares problem. The solution is given by the closed formula:

Finally, the iterative algorithm is given by:  
Start with an initial guess for : .  
For each iteration :

Calculate the weights using the kernel .  
Calculate the new kernel estimation using the weights .

Question 9

We define , then:  
Using the fact that we get , plugging it in, we get:

Question 10

As we have seen in question 2:   
Using the result from question 1, . Plugging it back:

Question 11

Given the low-resolution image, we can randomly extract patches from it, zoom-in the patches using the estimated kernel and search for similar patches in the low-resolution image. Suppose we find such a patch whose scaled version is similar to other patches; we can use it to recover the true using the same algorithm as in question 8. According to question 10, using the zoomed-in version patches is equivalent to using the high-resolution patches we used before (as in question 6). Using question 8 again we can refine the estimate of the kernel and repeat the process (find better patches using the estimated kernel and then improve the estimation using the new patches).  
All in all, the new algorithm is as follows:

* Initialize
* For :
  + Initialize
  + While
    - Crop a random patch from the low-resolution image and add it to .
    - Down sample the patch after applying the kernel .
    - Search for similar patches in the low-resolution image.
  + Run algorithm (q8) with the low-resolution image and as the patch set to find the new estimation of .

Question 12

We can use MAP to recover the high-resolution image:  
Assuming we have:  
Assuming some prior over , :  
This can be seen as a regularized least-squares.

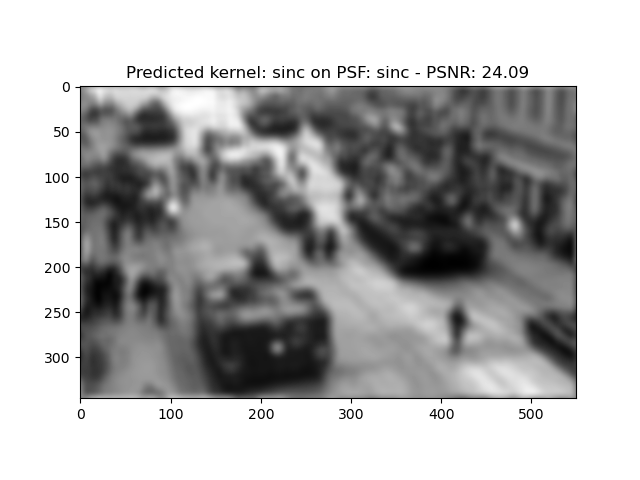
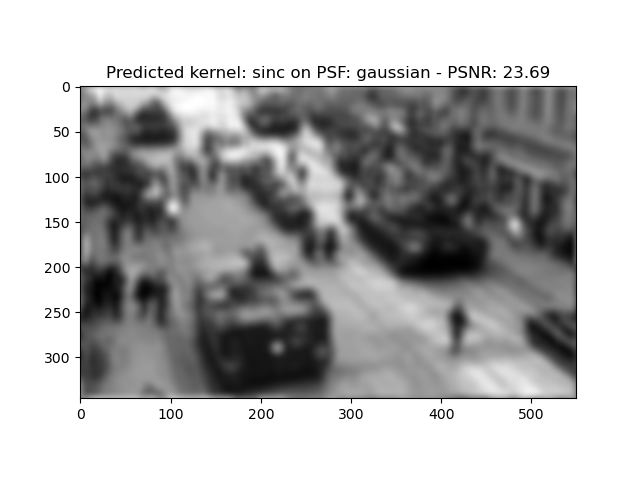
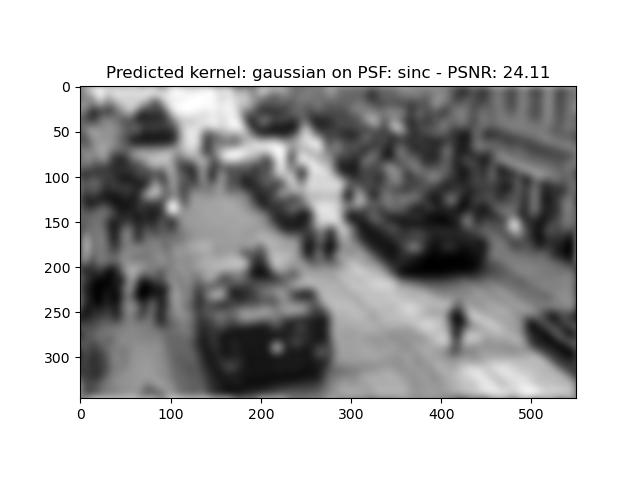
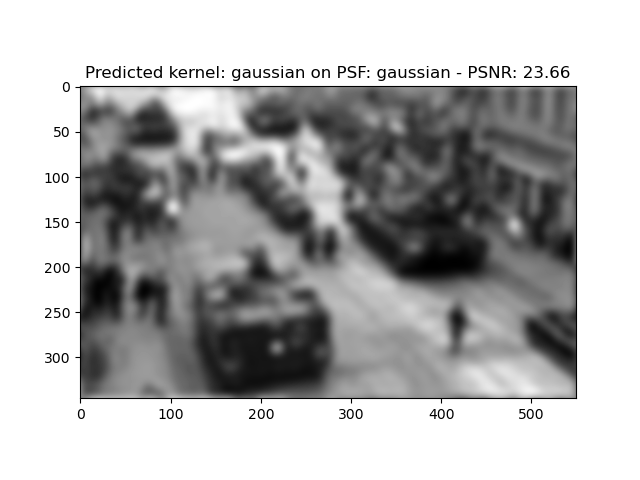
As we have seen in the lectures, this can be solved using a Wiener filter.

**Part 2 – Code**

By utilizing our algorithm and the paper “Nonparametric Blind Super-Resolution” we created a super-resolution algorithm.

Due to time complexity challenges using all patches in the image, we used a similar method to NLM to reduce the number of neighborhood patches which reduces significantly the time complexity.

Using the algorithm, we achieved the following results:



As can be seen, the sinc PSF was slightly easier to recover (can be due to some hyper-parameters). Other than that, both