Answers to questions in Lab 1: Filtering operations

Name: Jonathan Rintala **Program:** TIEMM, MAIG

Instructions: Complete the lab according to the instructions in the notes and respond to the questions stated below. Keep the answers short and focus on what is essential. Illustrate with figures only when explicitly requested.

Good luck!

1.3 Basis functions

Question 1: Repeat this exercise with the coordinates p and q set to (5, 9), (9, 5), (17, 9),

(17, 121), (5, 1) and (125, 1) respectively. What do you observe?

Answers:

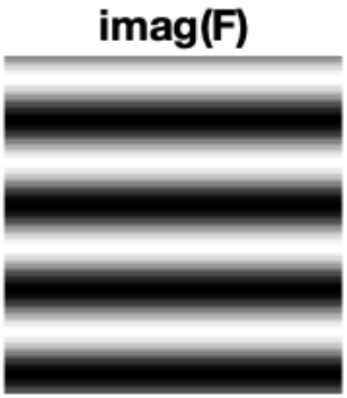
My observations are:

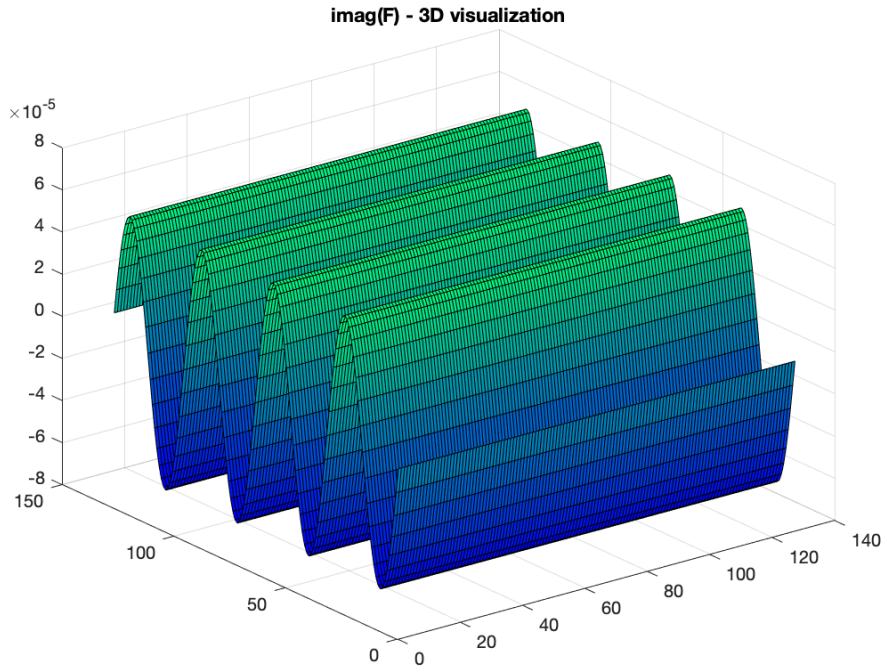
- The placement of the dot determines the waves' directions
- Distance to origin (upper left corner) from dot, determines the frequency, i.e. a dot close to the origin will yield a lower frequency, which can be observed by waves with longer wavelength, than a dot far away **(if not a multi)**
- Placing a dot on the x-axis yields vertical waves
- Placing a dot on the y-axis yields horizontal waves i.e. (125,1) or (5,1)
- Placing a dot on other coordinates yields diagonal waves

Question 2: Explain how a position (p, q) in the Fourier domain will be projected as a sine wave in the spatial domain. Illustrate with a Matlab figure.

Answers:

$$Inversion\ theorem:\ F(x) = F_D^{-1}(\hat{F})(x) = \frac{1}{N} \sum_{u \in [0..N-1]^2} [\hat{F}(u) * e^{\frac{+2\pi i u^T x}{N}}] \quad (4)$$





Figures - Displaying the imaginary part of the projection of the Fourier domain position (5,1) => the spatial domain. Visualizations in 2D resp. 3D.

Question 3: How large is the amplitude? Write down the expression derived from Equation (4) in the notes. Complement the code (variable amplitude) accordingly.

Answers:

- $Inversion\ theorem:\ F(x) = F_D^{-1}(\hat{F})(x) = \frac{1}{N} \sum_{u \in [0..N-1]^2} [\hat{F}(u) * e^{\frac{+2\pi i u^T x}{N}}] \quad (4)$
- $Eulers\ formula:\ e^{iw^T x} = e^{i(w_1 x_1 + w_2 x_2)} = cos(w^T x) + isin(w^T x) \quad (5)$

$$From\ (4) => F(m,n) = \frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} [\frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} [\hat{F}(u,v) e^{\frac{+2\pi i n v}{N}}] e^{\frac{+2\pi i m u}{M}}]$$

$$=> \{Euler's\ (5)\} =>$$

$$\frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} [\frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} [\hat{F}(u,v) cos(\frac{2\pi n v}{N} + \frac{2\pi m u}{M}) + i sin(\frac{2\pi n v}{N} + \frac{2\pi m u}{M})]]]$$

This gives us:

- Real part R(z): $\frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} [\sum_{n=0}^{N-1} [\hat{F}(u,v) cos(\frac{2\pi n v}{N} + \frac{2\pi m u}{M})]]$
- Imaginary part Im(z): $\frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} [\sum_{n=0}^{N-1} [\hat{F}(u,v) sin(\frac{2\pi n v}{N} + \frac{2\pi m u}{M})]]$

From Lecture 3 , page 39, the Fourier spectrum is given by:

- Fourier spectrum: $|F(u,v)| = \sqrt{Re^2(u,v) + Im^2(u,v)}$

Followingly, the amplitude A is given by:

- $A = max(\sqrt{R(u,v)^2 + Im(u,v)^2})$

Since, max of our cosine resp. sine will be = 1:

- $A = \frac{1}{\sqrt{MN}} max(\sum_{m=0}^{M-1} [\sum_{n=0}^{N-1} \hat{F}(u,v) * 1)]]$

Thus, implementation in Matlab:

```
amplitude = max(abs(Fhat(:)))/sz^2;
```

Note: Matlab's ifft function includes a scaling factor of $1/M^2$ and we

Finally, with known values for dimensions and max of Fourier transform we can compute A directly:

- $M = N = 128$
- $max(\hat{F}(u,v))$ = 1 (the point that is being placed out has value 1)
- $A = \frac{1}{\sqrt{128^2}} * 1 = \frac{1}{128}$

Question 4: How does the direction and length of the sine wave depend on p and q? Write down the explicit expression that can be found in the lecture notes. Complement the code (variable wavelength) accordingly.

Answers:

(1) Wavelength from lecture 03, page 22: $\lambda = \frac{2*\pi}{||(w)||} = \frac{2*\pi}{\sqrt{w_1^2+w_2^2}}$

(2) Phase (angle) of the Fourier transformation, from lecture 03, page 25:

$\phi(w_1, w_2) = tan^{-1} \frac{Im(w1,w2)}{Re(w1,w2)}$

(5) Def. from lab description: $w_D = \frac{2*\pi*u}{N}$

- Since we are calculating an angle, we should use the centered coordinates, i.e.
  - u = uc
  - v = vc
- With number of pixels: N = sz
- Thus, we are getting the following expression for lambda using (1) and (5):

  $\lambda = \frac{2*pi}{\sqrt{(\frac{2*pi*uc}{sz})^2}+\sqrt{(\frac{2*pi*vc}{sz})^2}}$
- In MATLAB:

```
w_1 = sqrt(2pi*uc)/sz;
w_2 = sqrt(2pi*vc)/sz;
wavelength = 2*pi/(w_1^2+w_2^2)
```

Question 5: What happens when we pass the point in the center and either p or q exceeds half the image size? Explain and illustrate graphically with Matlab!

Answers:

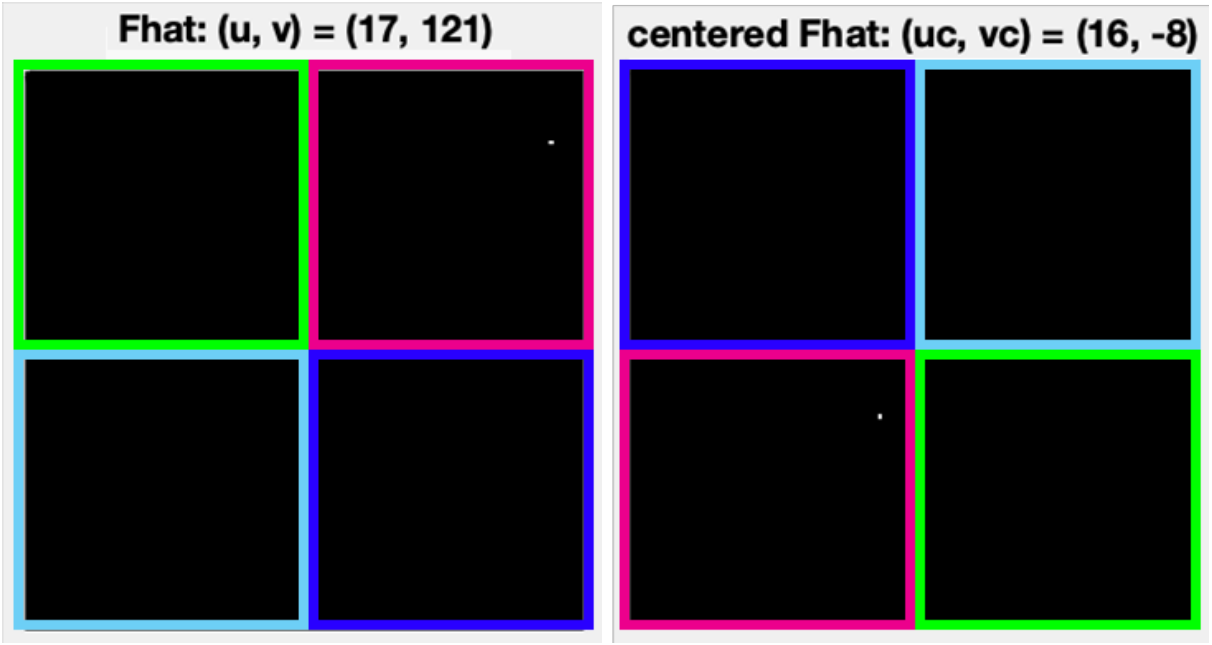
The Matlab fftshift works as explained by the help function:

```
>> help fftshift
    "fftshift Shift zero-frequency component to center of spectrum. /../
For matrices, fftshift(X) swaps the first and third quadrants and the
second and fourth quadrants."
```

This moves the default origin of the upper left corner, to instead having the center of the spectrum being the Fourier transform's center; which means we have to calculate the new position of the point **relative to** the new origin of the center. A point (p,q) that has passed the center i.e. having either $p > sz/2$ OR $q > sz/2$ will thus get its position according to:

- $uc = u - 1 - sz$
- $vc = v - 1 - sz$

This can be illustrated with the figures below:



Figures - To the left: Fhat with default origin. To the right: centered Fhat with origin in center of the image.

In the illustrations above, $u <= sz/2$ but $v > sz/2$ which means the position will be calculated as:

- $uc = u - 1 = 17 - 1 = 16$
- $vc = v - 1 - sz = 121 - 1 - 128 = -8$

Question 6: What is the purpose of the instructions following the question *What is done by these instructions?* in the code?

Answers:

The purpose of the if-statements is to, as explained previously in question 5, calculate the position after the centering of Fhat's zero component (origin) to the center of the image, in order to get the correct relative position to the origin. We take -1, since the first element in the image matrix is retrieved by Fhat(1,1), and the origin should be in the very center of the image i.e. not $sz/2$, but rather $sz/2 - 1$.

```
% What is done by these instructions?

%This checks whether the point is in the upper side of the image
(vertically)
if (u <= sz/2)
    uc = u - 1;
else
    uc = u - 1 - sz;
end
```

```
%This checks whether the point is in the left side of the image
(horizontally)
if (v <= sz/2)
    vc = v - 1;
else
    vc = v - 1 - sz;
end
```

1.4 Linearity

Question 7: Why are these Fourier spectra concentrated to the borders of the images? Can you give a mathematical interpretation? **Hint:** think of the frequencies in the source image and consider the resulting image as a Fourier transform applied to a 2D function. It might be easier to analyze each dimension separately!

Answers:

"To better understand the role of the logarithm function, see also Lab 0B "Elementary image operations"." - **SINC FUNCTION**

Discrete Fourier transformation in two dimensions - given in lecture 03, page 39:

$$\hat{F}(u,v) = \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} [\sum_{n=0}^{N-1} [f(m,n) e^{-2\pi i (\frac{mu}{M} + \frac{nv}{N})}]] \quad (1)$$

Looking at G we have:

$$f(m,n) = \begin{cases} 1, if\ 57 \leq n \leq 72 \\ 0, else \end{cases}$$

We can thus split up (1) into its two constituting sums - due to the separability property *(Property I - Lecture 4, page 9)*, which says a 2D DFT can be implemented as a series of 1D DFTs along each column followed by 1D DFTs along each row:

$$\hat{F}(u,v) = \frac{1}{\sqrt{MN}} \sum_{n=57}^{72} [e^{-2\pi i (\frac{nv}{N})} \sum_{m=0}^{M-1} [e^{-2\pi i (\frac{mu}{M})}]]$$

Now we simplify the expression, with the property of Kronecker delta function - i.e. the Dirac distribution for discrete values (Lecture 4, page 27):

$$\delta(m) = \begin{cases} 1, & if\ m = 0 \\ 0, & if\ m \neq 0 \end{cases}$$

By identifying:

$$\sum_{m=0}^{M-1} [e^{-2\pi i (\frac{mu}{M})}]] = \delta(m)$$

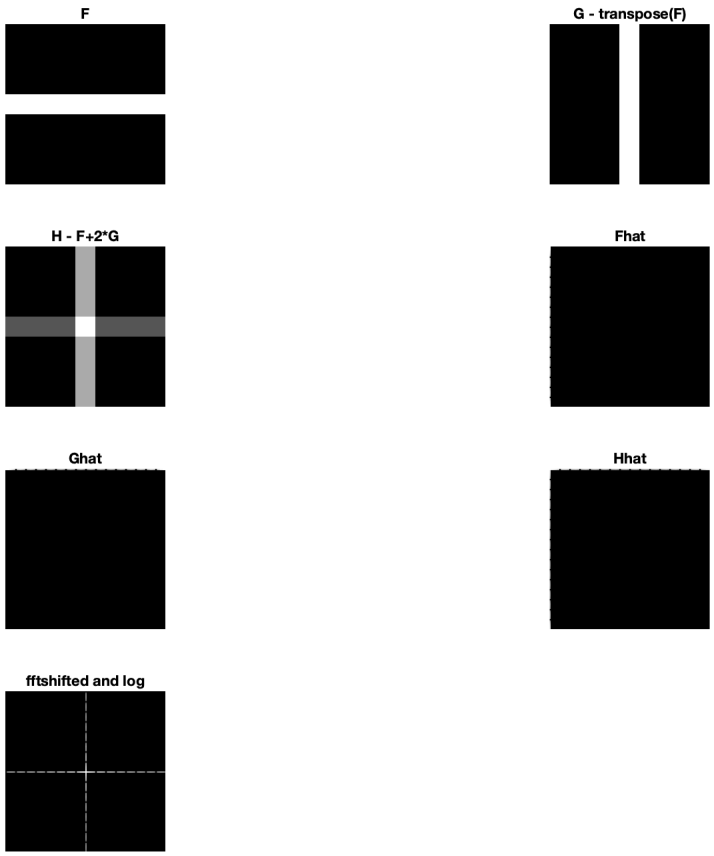
We can write the final expression as:

$$\hat{F}(u,v) = \frac{1}{\sqrt{MN}} \sum_{n=57}^{72} [e^{-2\pi i(\frac{nv}{N})}] * \delta(m)$$

This means, $\hat{F}(u,v)$ only is a non-zero value, as long as $m = 0$, which implies $\delta(m) = 1$

Thus, the Fourier spectra for G will be concentrated to the upper border.

The Fourier spectra for function F will be concentrated along the left border with the same logic (since G=F') i.e. since $\hat{F}(u,v)$ is a non-zero value as long as n=0 for F.



Question 8: Why is the logarithm function applied?

Answers:

The logarithm function will even out the distribution of the pixels, which means the difference between the lowest and highest frequencies will be reduced. This allows the low dynamic ranges to become visible and in that way allows for a more well balanced and more detailed image.

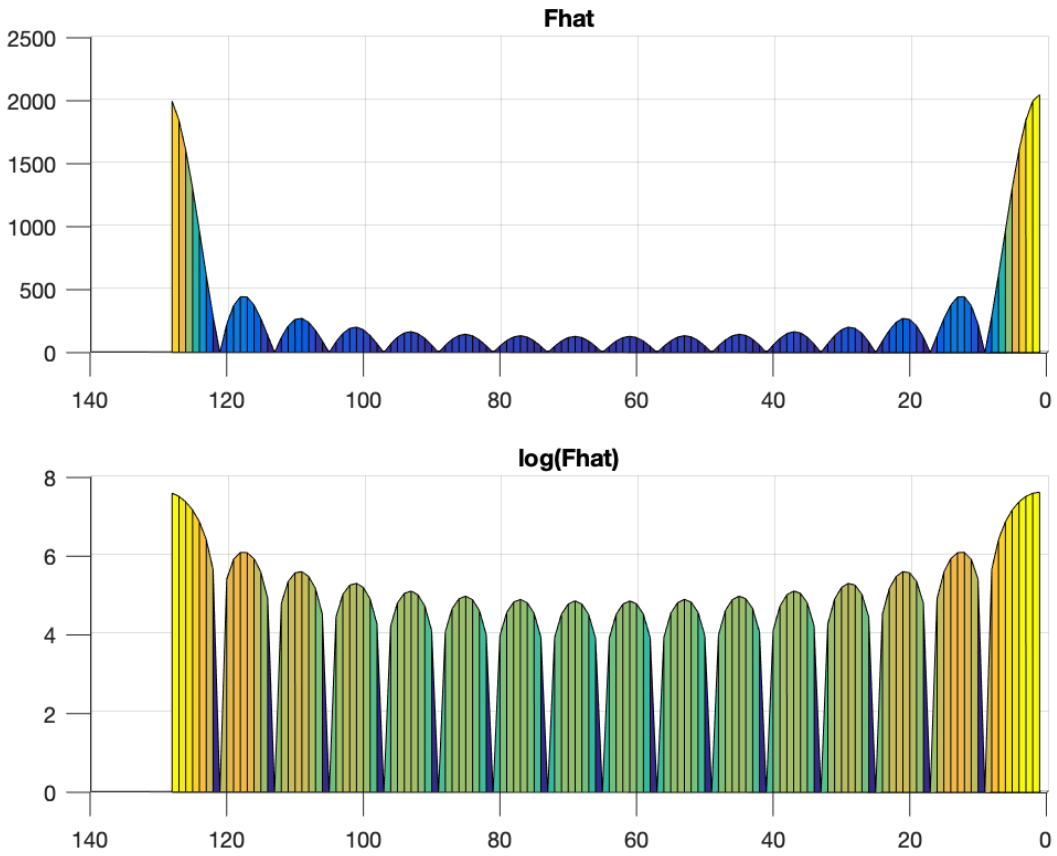


Figure - illustration comparing Fhat and log(Fhat)

Question 9: What conclusions can be drawn regarding linearity? From your observations can you derive a mathematical expression in the general case?

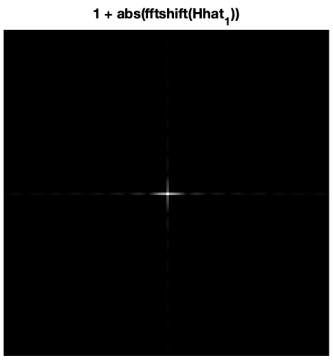
Answers:

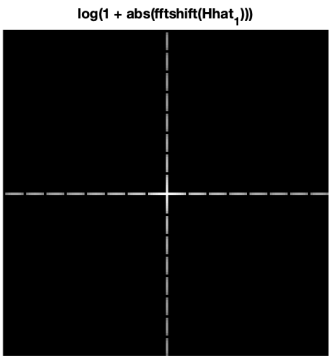
The conclusion regarding linearity that can be drawn, is the fact that the two images F and G can be combined either in the spatial or Fourier domain, the outcome of multiplying with a factor of two, will be achieved either way. Since, H becomes a linear combination of F and G according to:

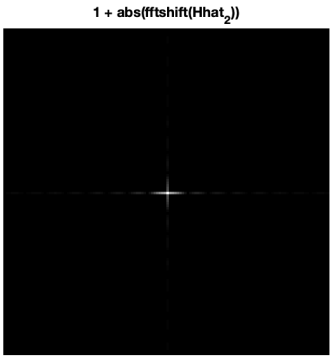
```
H = F + 2 * G;
```

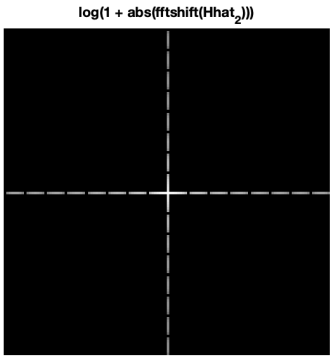
We can observe this visually by comparing the two different graphs when we have done the tranformation either before multiplication with 2 or after multiplication i.e. Hhat_1 and Hhat_2:

```
F = [ zeros(56, 128); ones(16, 128); zeros(56, 128)];
G = F';
H = F + 2 * G;

Fhat = fft2(F);
Ghat = fft2(G);

%Hhat_1: computing the linear combination before Fourier transform
Hhat_1 = fft2(H);

%Hhat_2: computing the linear combination after Fourier transform
Hhat_2 = fft2(F) + 2*fft2(G);
```









- Mathematical expression in the general case:

$$F(af_1(m,n) + bf_2(m,n)) = a\hat{f}_1(u,v) + b\hat{f}_2(u,v)$$
$$=>$$
$$af_1(m,n) + bf_2(m,n) = F^{-1}(a\hat{f}_1(u,v) + b\hat{f}_2(u,v))$$

1.5 Multiplication

Question 10*: Are there any other ways to compute the last image? Remember what multiplication in Fourier domain equals to in the spatial domain! Perform these alternative computations in practice.

Answers:

According to Theorem 2, Lecture 3, slide 31:

A convolution in the spatial domain is same as multiplication in the Fourier (frequency) domain.

$F(h * f) = F(h)F(f)$

To acquire the same results for the both, one must normalize the images in the frequency domain:

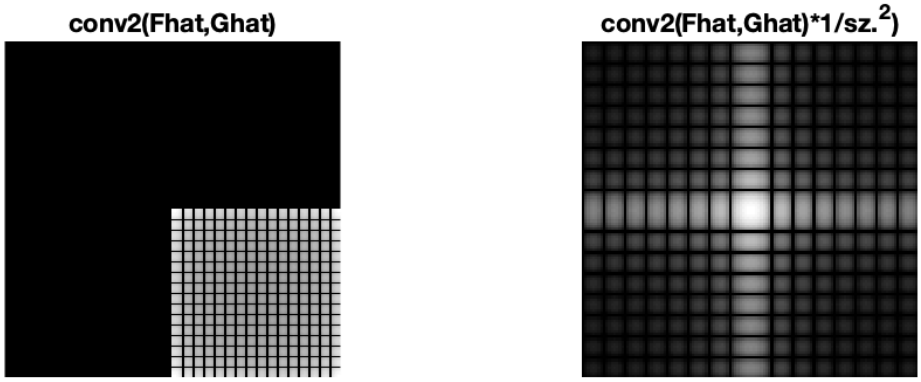


Figure 10.0 - displays the normalizing of the convolution in the Fourier domain

The point-wise matrix multiplication of $F.*G$ means only element combinations where 1*1 will yield a value of 1, everywhere else we will get zero, thus we get the spatial domain image that looks like a box:

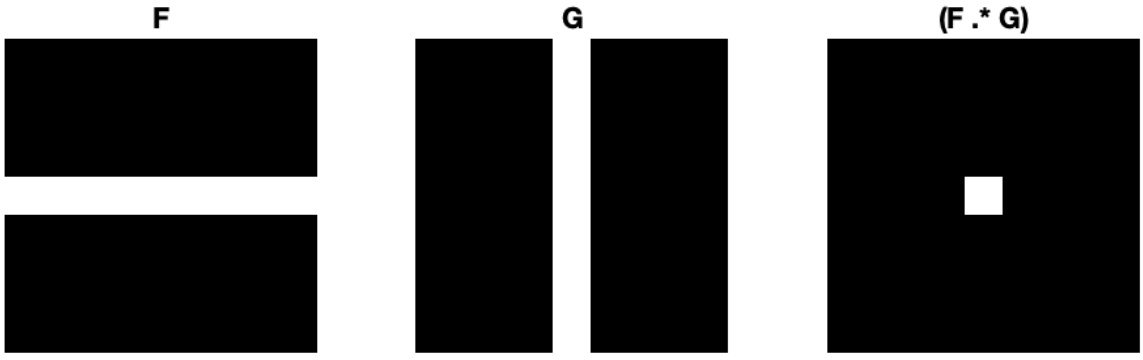


Figure 10.1 - displays multiplication in the spatial domain

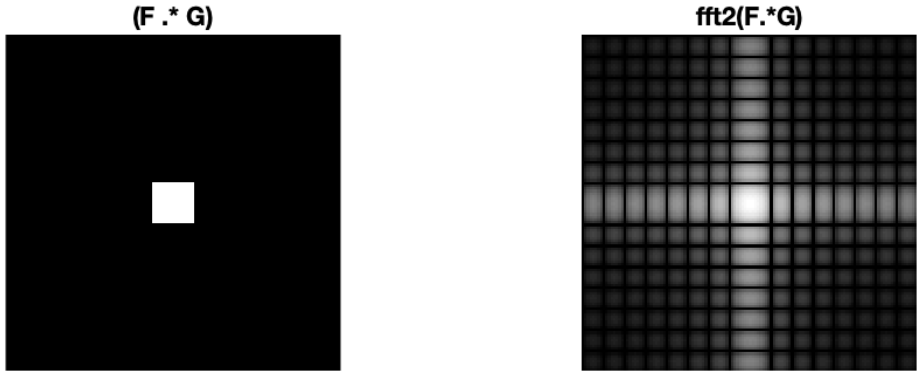


Figure 10.2 - displays the Fourier transform of the spatial multiplication

Then, we look at the Fourier transforms and the convolution of these in the Fourier domain:

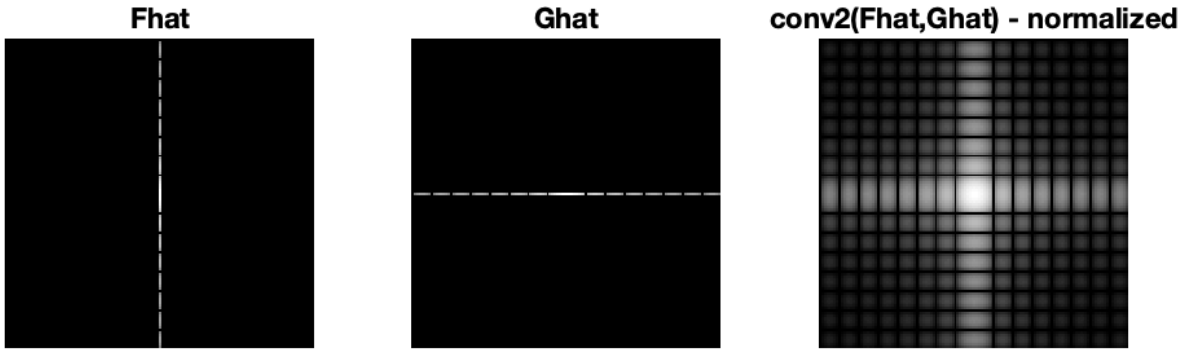


Figure 10.3 - displays $\hat{F}$ and $\hat{G}$, along with the normalized convolution

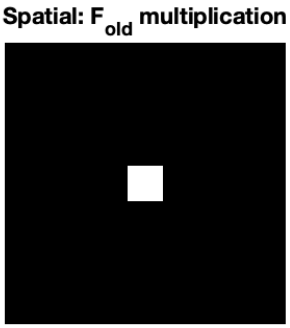
---

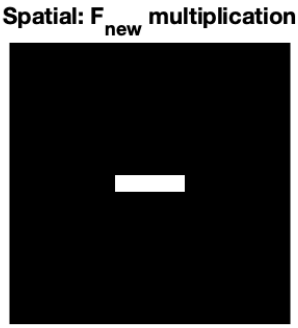
1.6 Scaling

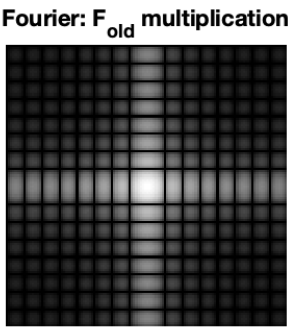
Question 11*: What conclusions can be drawn from comparing the results with those in the previous exercise? See how the source images have changed and analyze the effects of scaling.

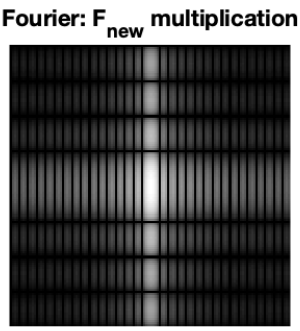
Answers:

- Previous F: 128x128 with white square









1.7 Rotation

Question 12: What can be said about possible similarities and differences? **Hint:** think of the frequencies and how they are affected by the rotation.

Answers:

- **ROTATION: !! HAVE TO ROTATE THE FREQUENCY DOMAIN IN SAME WAY - A POINT IN FREQ DOMAIN REPRESENTS THE DIRECTION OF WHICH A SINE WAVE IN THE SPATIAL DOMAIN WILL BE**
- FOR HUMAN GUASSSIAN FILTER LOOKS BETTER - FROM THEORY - HOW MUCH INFO
  - IDEAL LOW PASS FILTER - REMOVES ONLY WHAT HAS TO BE REMOVED
    - hard cut off - rippling effect
  - GAUSSIAN - ALSO SURPRESSES - NOT REALLY NECESSARY
    - slow change if long -- => frequency
    - frequency domain will be the opposite

When rotating the image in the spatial domain, the image representation in the Fourier domain rotates with it.

1.8 Information in Fourier phase and magnitude

Question 13: What information is contained in the phase and in the magnitude of the Fourier transform?

Answers:

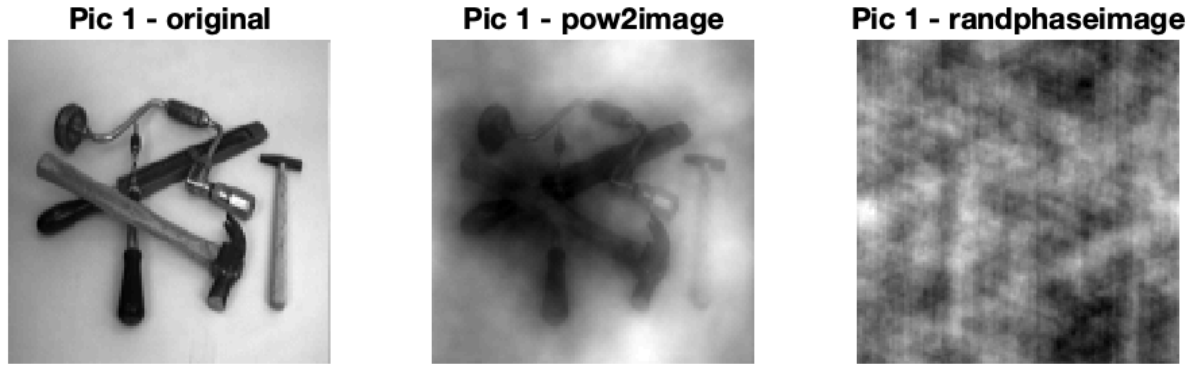


Figure 13.1 - Image 1 "few128"

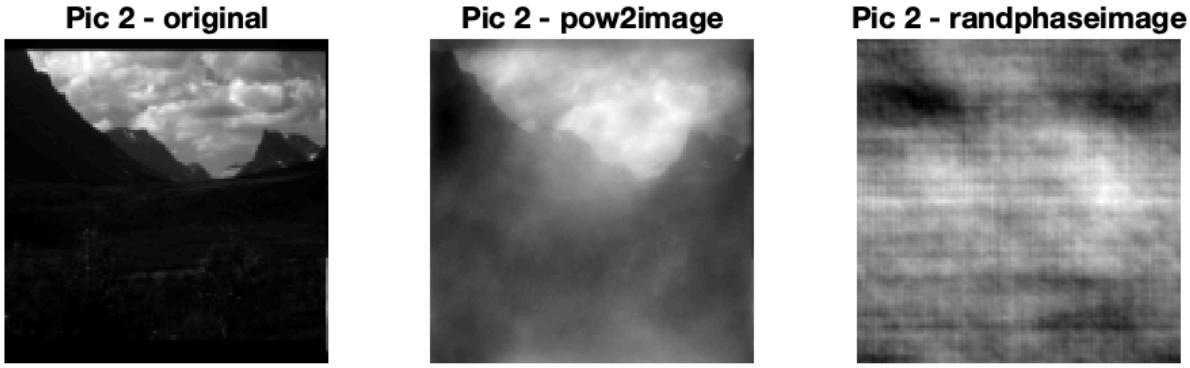


Figure 13.2 - Image 2 "nallo128"

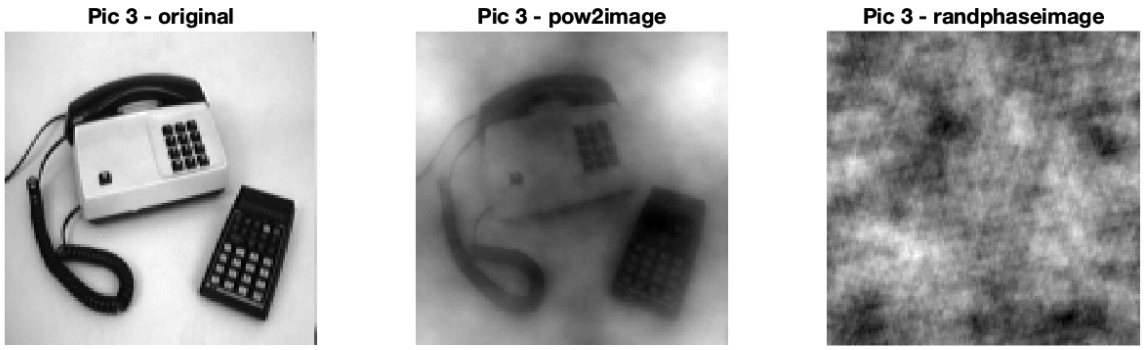


Figure 13.3 - Image 3 "phonecalc128"

Conclusion:

- The magnitude basically contains the information about the intensity of the image.

- The phase contains information about the countours and edges.
    - I.e. the position of the contours

We can see that with ***pow2image*** we still see the contours of the image, but the magnitude is not the same. This allows us to still identify the image, indicating that the phase is important for identifying the objects.

For ***randphaseimage***, we cannot see any contours of the original image, but the magnitude is still the same, however in different positions. The ratio between bright and dark pixels is is thus still about the same, but without phase there is no information about where these pixels should be located => making it difficult for identifying the objects in the image.

---

2. Gaussian convolution implemented via FFT

Question 14: Show the impulse response and variance for the above-mentioned t-values. What are the variances of your discretized Gaussian kernel for t = 0.1, 0.3, 1.0, 10.0 and 100.0?

Answers:

The impulse responses from the Gaussian kernel function (filter) are illustrated below:

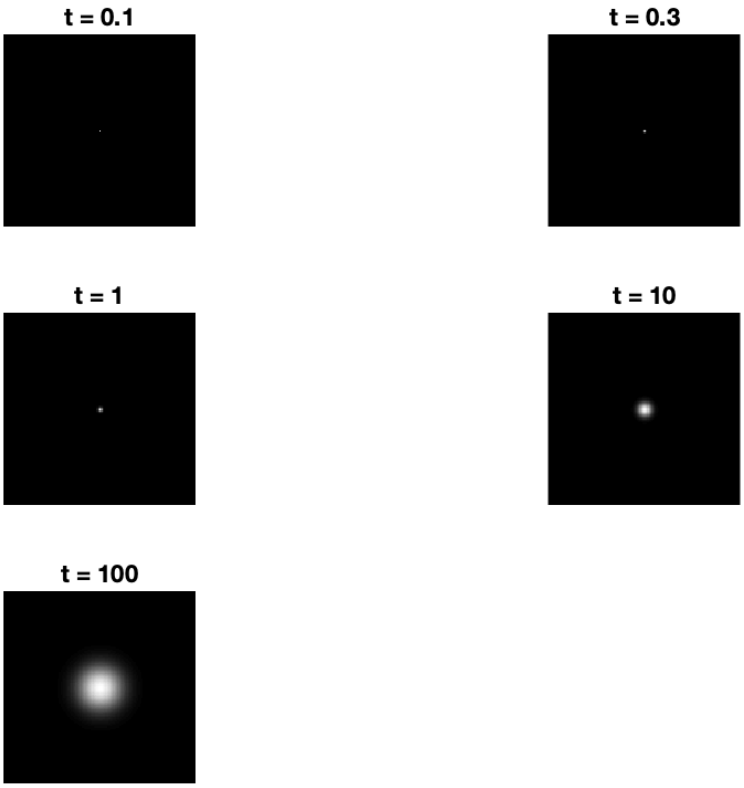


Figure 1 - Illustrating the impulse responses in 2D

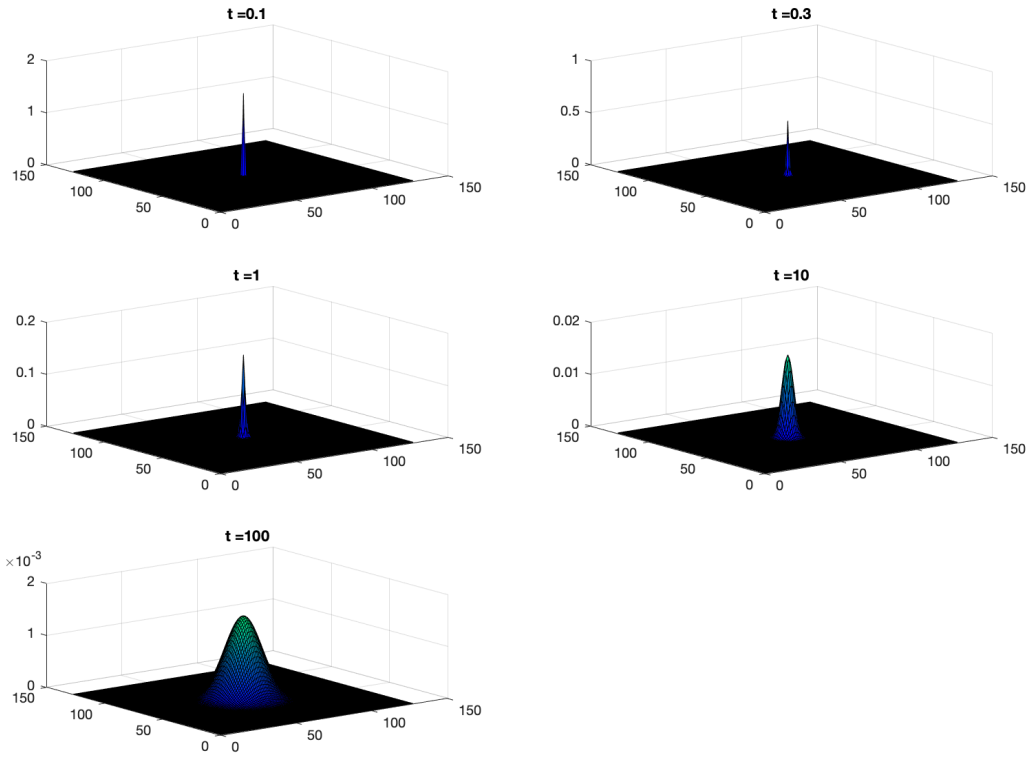


Figure 2 - Illustrating the impulse responses in 3D

t	covariance
0.1	$\begin{bmatrix} 0.0133 & 0.0000 \\ 0.0000 & 0.0133 \end{bmatrix}$
0.3	$\begin{bmatrix} 0.2811 & 0.0000 \\ 0.0000 & 0.2811 \end{bmatrix}$
1.0	$\begin{bmatrix} 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{bmatrix}$
10.0	$\begin{bmatrix} 10.0000 & 0.0000 \\ 0.0000 & 10.0000 \end{bmatrix}$
100.0	$\begin{bmatrix} 100.0000 & 0.0000 \\ 0.0000 & 100.0000 \end{bmatrix}$

Thus, we can see how a high t-value => "wide" impulse-response => high variance => i.e. a blurrier picture, with higher frequencies

- tendency if we have a low value (below 1) to become non-gaussian

---

Question 15: Are the results different from or similar to the estimated variance? How does the result correspond to the ideal continuous case? Lead: think of the relation between spatial and Fourier domains for different values of t.

Answers:

For values of $t < 1$, such as 0.1 and 0.3, the results are different from the estimated variance. For higher values of t, the values we get are similar to the correct ones:

t	abs diff variance (compared to discrete)
0.1	$\begin{bmatrix} 0.0867032748238839 & 5.96379052957304 * 10^{-14} \\ 5.96379052957304 * 10^{-14} & 0.0867032748231496 \end{bmatrix}$
0.3	$\begin{bmatrix} 0.0189461699182294 & 9.31508138947913 * 10^{-14} \\ 9.31508138947913 * 10^{-14} & 0.0189461699172603 \end{bmatrix}$
1.0	$\begin{bmatrix} 2.11231750313345 * 10^{-7} & 1.15968813537873 * 10^{-14} \\ 1.15968813537873 * 10^{-14} & 2.11231289681812 * 10^{-7} \end{bmatrix}$
10.0	$\begin{bmatrix} 4.52615722679184 * 10^{-12} & 1.21620265819178 * 10^{-14} \\ 1.21620265819178 * 10^{-14} & 4.30944169238501e-12 \end{bmatrix}$
100.0	$\begin{bmatrix} 5.60787455583522 * 10^{-7} & 9.51381657366261 * 10^{-14} \\ 9.51381657366261 * 10^{-14} & 5.60787484005232 * 10^{-7} \end{bmatrix}$

Figure 1 - Comparison of variance to built-in function $discgaussfft$

t	abs diff variance (compared to continuous)
0.1	$\begin{bmatrix} 0.0867032748147737 & 1.95594111278971e-14 \\ 1.95594111278971e-14 & 0.0867032748144079 \end{bmatrix}$
0.3	$\begin{bmatrix} 0.0189461699106196 & 1.75961603661103e-15 \\ 1.75961603661103e-15 & 0.0189461699102158 \end{bmatrix}$
1.0	$\begin{bmatrix} 2.11226521606989e-07 & 2.14436875313702e-15 \\ 2.14436875313702e-15 & 2.11226405810727e-07 \end{bmatrix}$
10.0	$\begin{bmatrix} 1.28252963804698e-12 & 1.06791235017787e-15 \\ 1.06791235017787e-15 & 1.38022926421399e-12 \end{bmatrix}$
100.0	$\begin{bmatrix} 6.71770507665315e-07 & 8.69911019992506e-17 \\ 8.69911019992506e-17 & 6.71770095550528e-07 \end{bmatrix}$

Figure 2 - Comparison of variance to continous function C

Where C is:

$ C(g(\cdot,\cdot;t)) = t \begin{bmatrix} 1&0\\0&1 \end{bmatrix}$

Conclusion: smaller differences in variance for higher values of t

---

Question 16: Convolve a couple of images with Gaussian functions of different variances (like t = 1.0, 4.0, 16.0, 64.0 and 256.0) and present your results. What effects can you observe?

Answers:

The following images 1-3 were used:

Figure 16.0 - Original images that were used

The results of convolutions with Gaussian functions of varying degree, letting $t = [1.0, 4.0, 16.0, 64.0, 256.0]$ can be seen below:

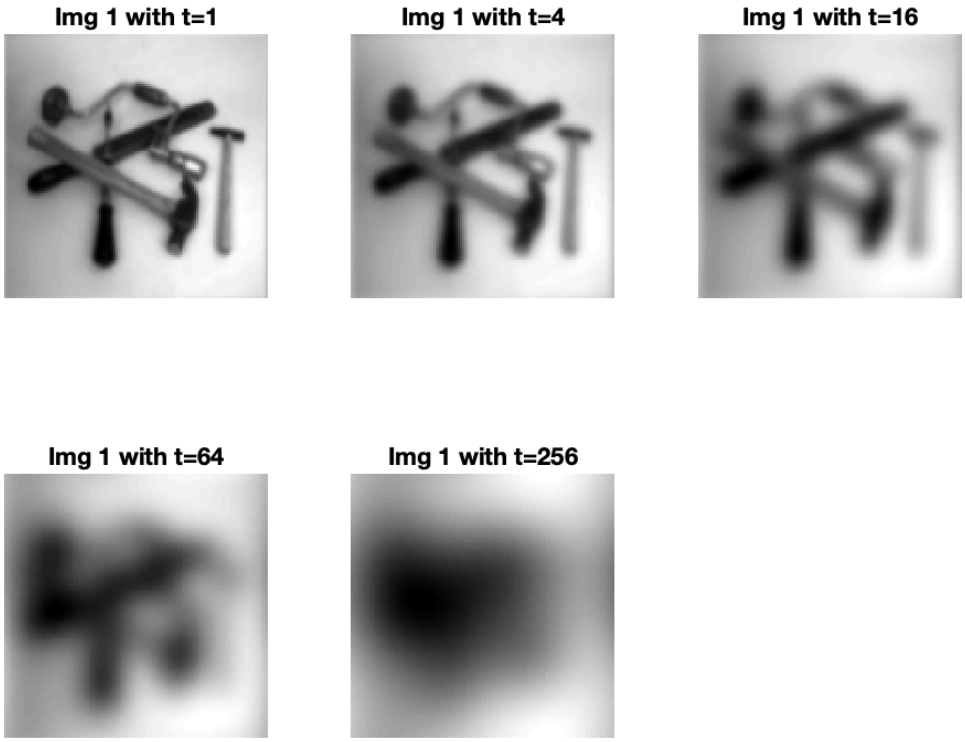


Figure 16.1 - Convolutions with different variances on image 1 ("few128")

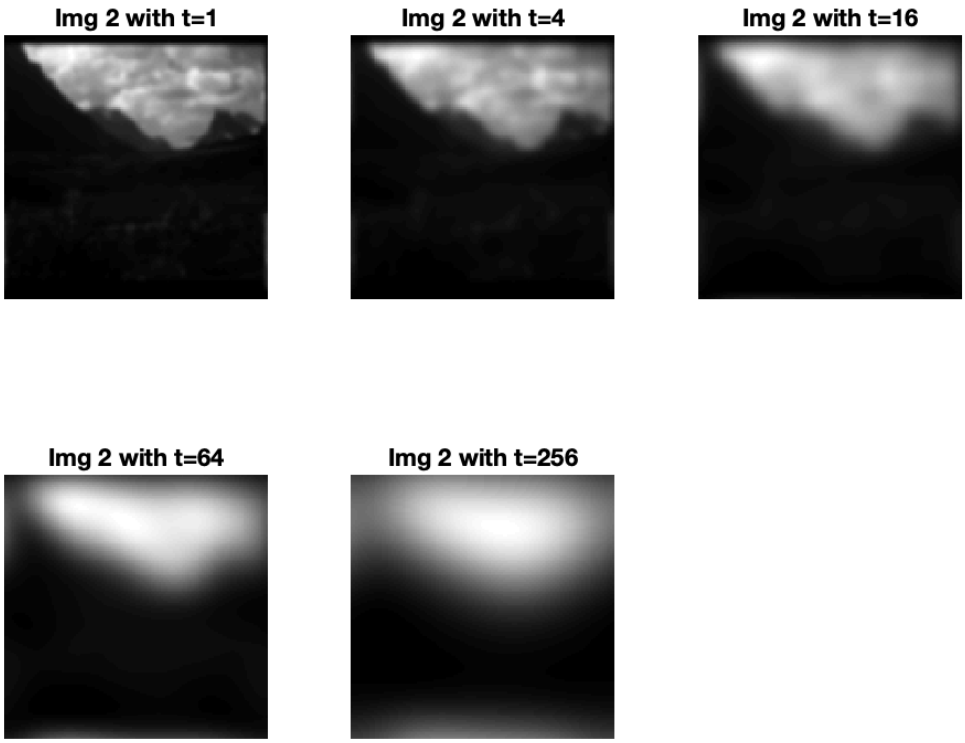


Figure 16.2 - Convolutions with different variances on image 2 ("nallo128")

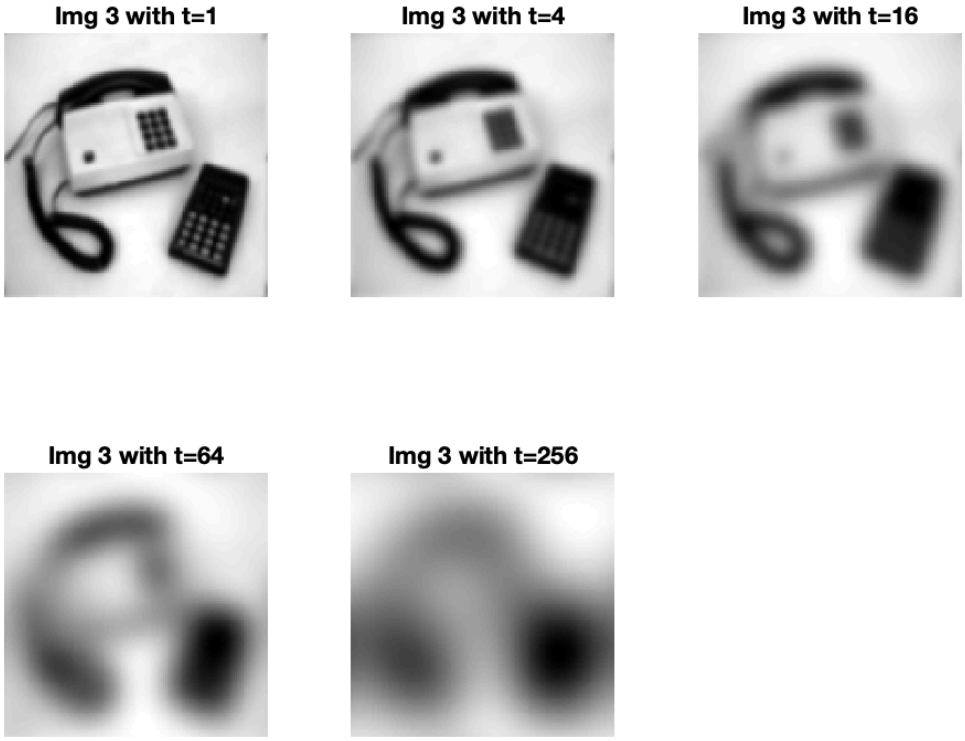


Figure 16.3 - Convolutions with different variances on image 3 ("phonecalc128")

Conclusion: Increasing the variance input to the Gaussian function for convolution, seems to yield a much blurrier results, i.e. more and more of the higher frequencies of the image is being removed (surpressed).

---

3. Smoothing

Question 17: What are the positive and negative effects for each type of filter? Describe what you observe and name the effects that you recognize. How do the results depend on the filter parameters? Illustrate with Matlab figure(s).

Answers:

	+	-
Gaussian smoothing	+ Smooths the image + Handles Gaussian noise well	- Does not preserve edges - Does not handles sap noise well
Median filtering	+ Preserves edges well + Handles both sap and gaussnoise well	- Non-distinct features get smoothened out - Image looks like it's painted when window size is increased
Ideal low-pass filtering	+ Removes white noise from sap + Easy to apply	- Could distort image and produce rippling effect - Does not perform well on neither gauss nor sap noise

The original images with added noise:

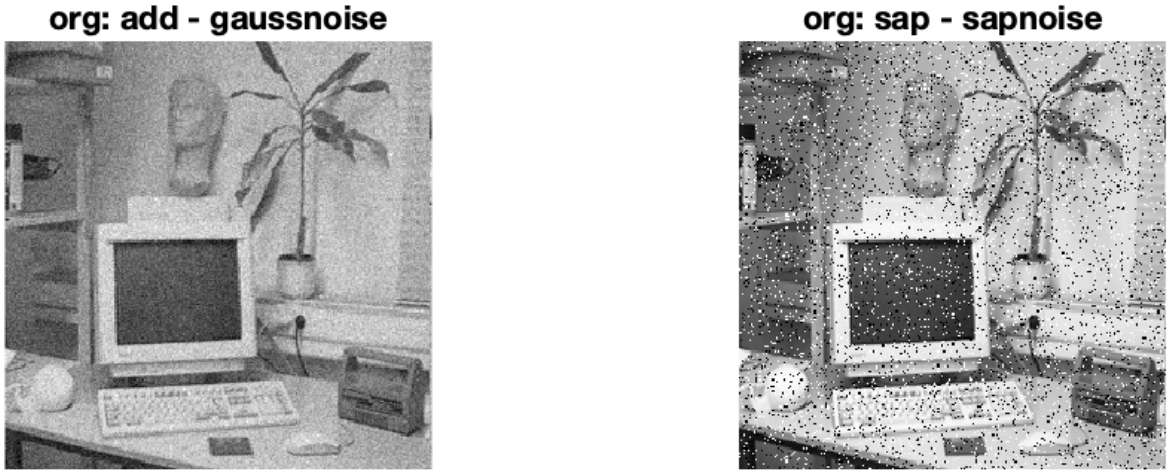


Figure 17.0 - Original images with introduced noise

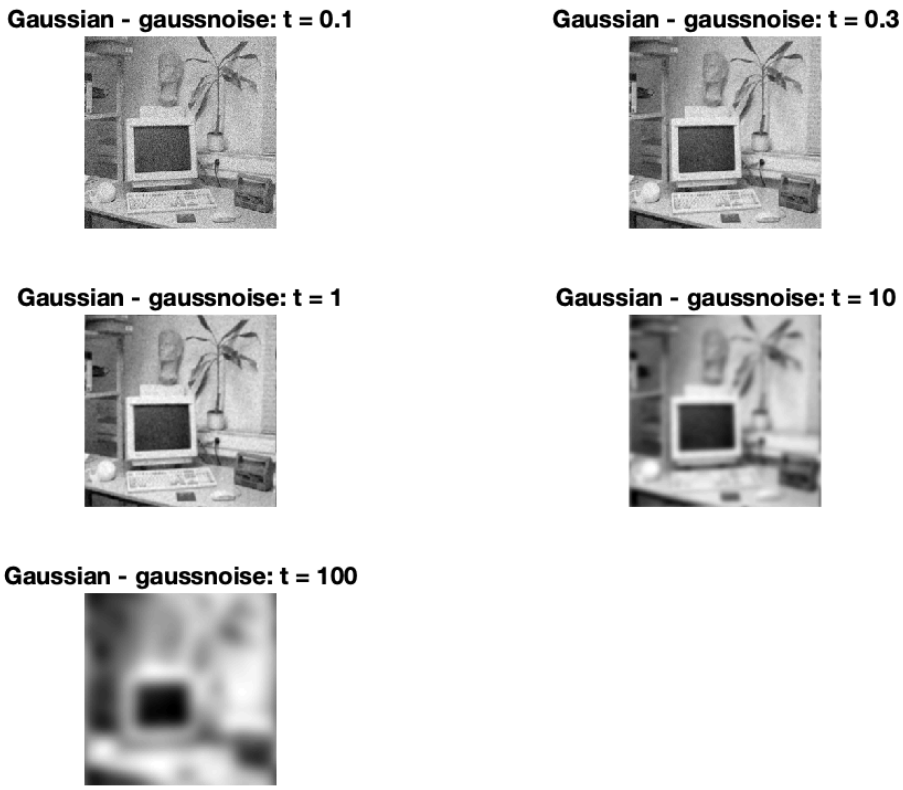


Figure 17.1 - Gaussian filter on gauss noise

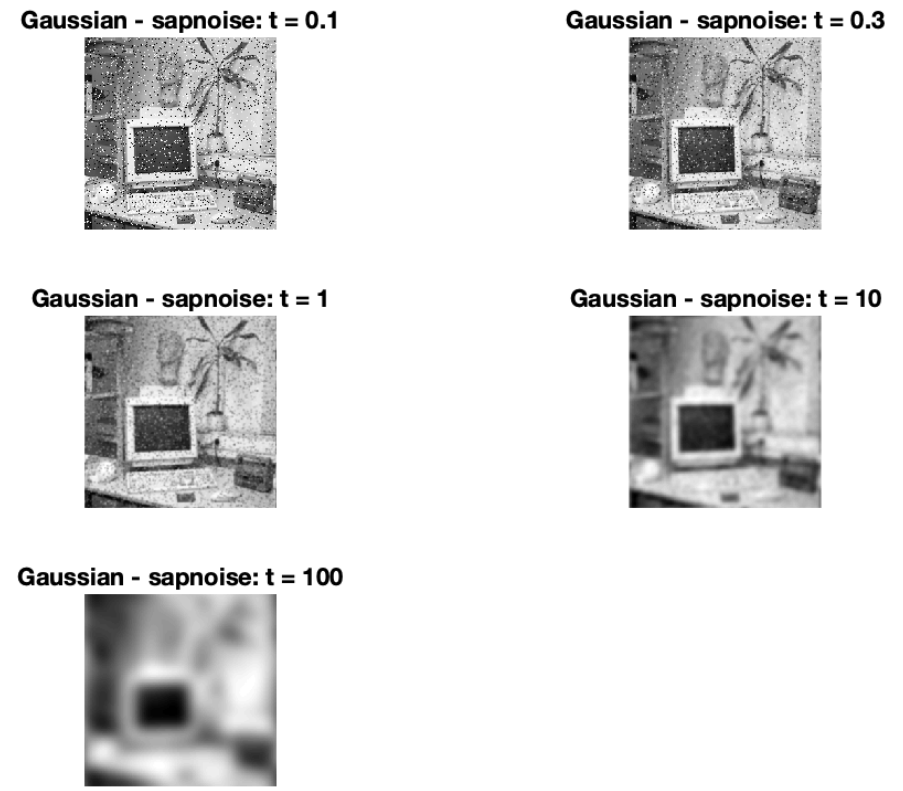


Figure 17.2 - Gaussian filter on salt and pepper noise

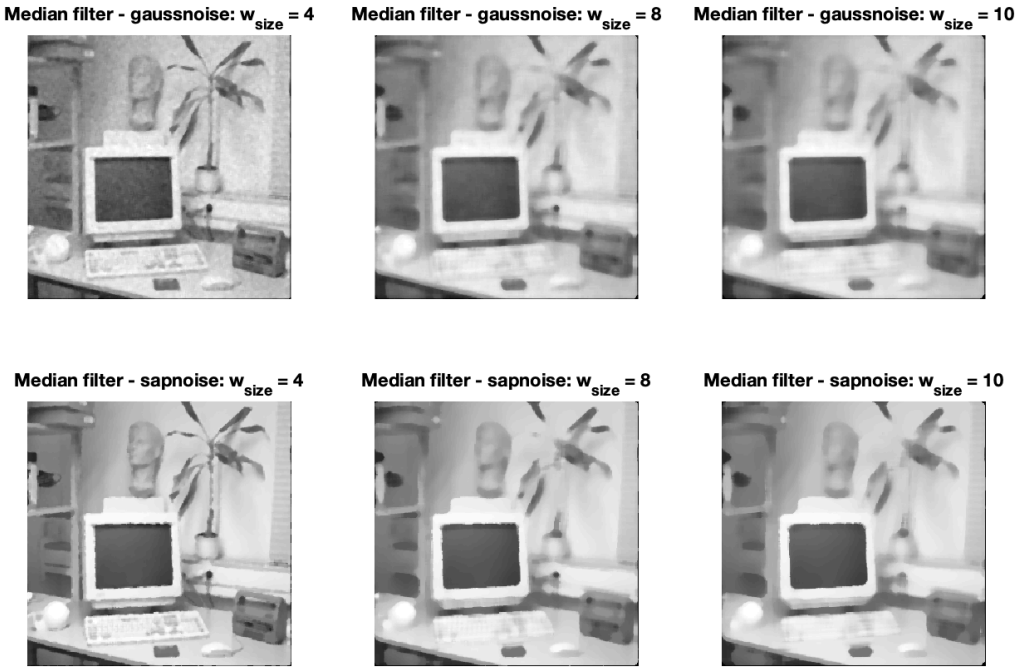


Figure 17.3 - Median filter, altering window size, on both types of noise

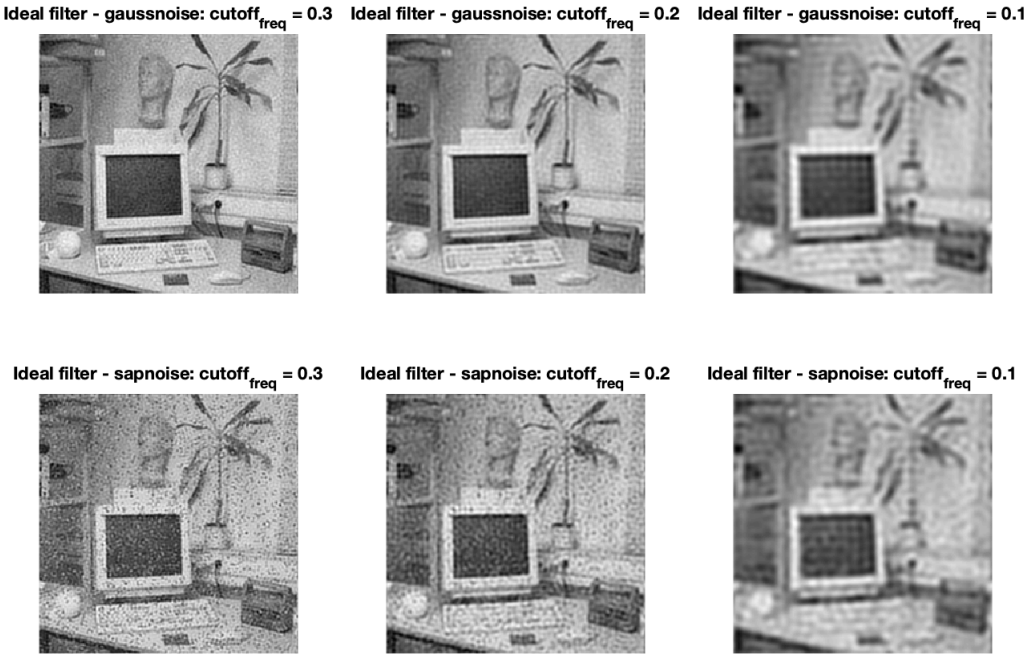


Figure 17.4 - Ideal low-pass filter, altering cut-off frequency, on both types of noise

---

Question 18: What conclusions can you draw from comparing the results of the respective methods?

Answers:

Take-aways from the compariosn of filtering methods:

- **Gaussian:**
  - Good performance on the add image (gauss noise)
  - Increasing variance t of kernel generates a blurrier image, with less noise
- **Median:**
  - Best performance overall
  - Removes noise of both types
  - And preserves the edges
- **Ideal low-pass:**
  - Lacking performance on all pictures
  - Removes high frequencies, and the lower the cut-off frequency, the more of the frequency spectra is being removed => less details and noise

---

Question 19: What effects do you observe when subsampling the original image and the smoothed variants? Illustrate both filters with the best results found for iteration i = 4.

Answers:

Figure 19.0 - Original subsampled images

Figure 19.1 - Smoothened subsampled images with gaussian filter

Figure 19.2 - Smoothened subsampled images with ideal filter

Question 20: What conclusions can you draw regarding the effects of smoothing when combined with subsampling? Hint: think in terms of frequencies and side effects.

Answers:

When using $rawsubsample(image)$ we reduce the size of the image by a factor of two in each dimension, i.e. by picking out every second pixel along each dimension.

- **Subsampling:** "refers to sampling at a rate (either in space or time) that's lower than the Nyquist criterion would indicate. It usually follows some sort of low-pass or bandpass filter that reduces the information content of the original signal to a level appropriate for the new sample rate."
- By smoothing the image prior to the subsampling, more data can be preserved, i.e. we can prevent losing information from the image
- Blurring image, reduces the frequency and reduces the Nyquist rate to better match the new subsampled image.
- This method can be used to counteract aliasing.
