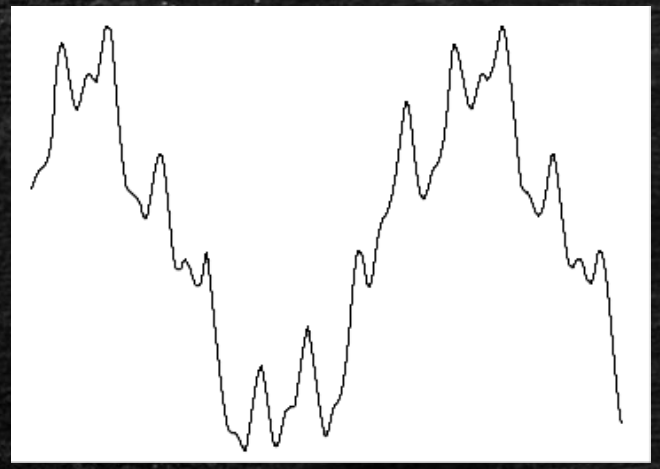
Prednáška 12

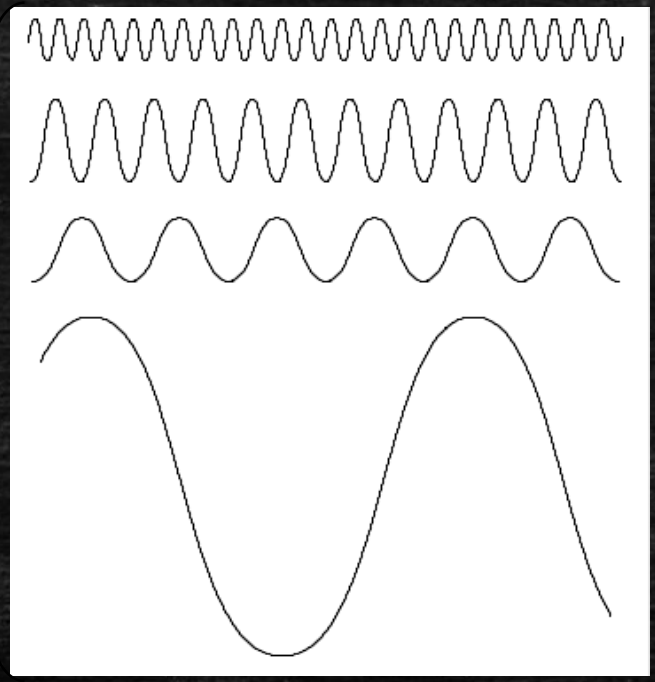
Praktikum z MATLABu
Elena Šikudová

Jean Baptiste Joseph Fourier (1768-1830)

Každá periodická funkcia sa dá vyjadriť ako vážená suma sínusov a kosínusov rôznych frekvencií.



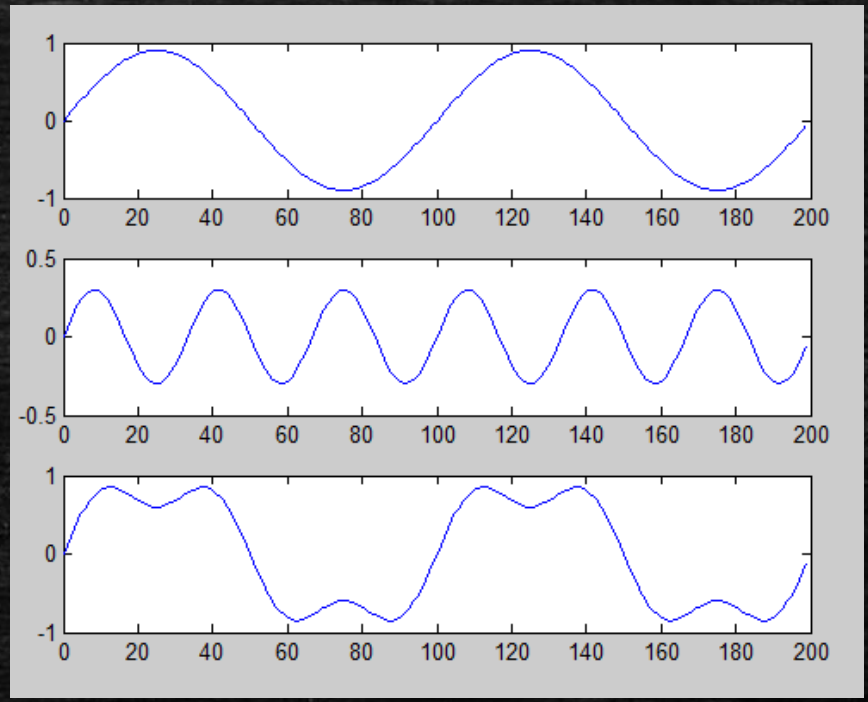
=



$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos\frac{2\pi nt}{T} + \sum_{n=1}^{\infty} b_n \sin\frac{2\pi nt}{T}$$

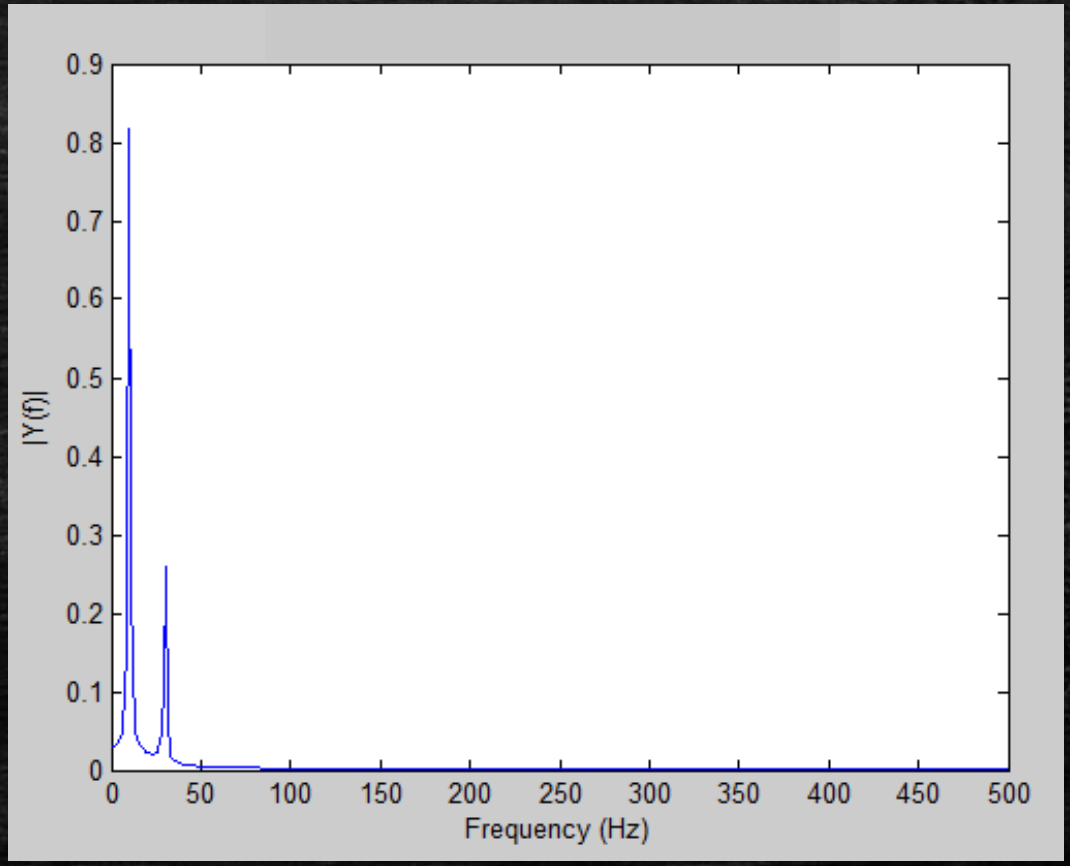
MATLAB – periodický signál

```
Fs = 1000;              % Sampling frequency
T = 1/Fs;               % Sample time
L = 1000;               % Length of signal
t = (0:L-1)*T;          % Time vector

y1 = 0.9*sin(2*pi*10*t);


y2 = 0.3*sin(2*pi*30*t);


y3 = y1 +y2;
```



MATLAB - periodický signál

```
NFFT = 2^nextpow2(L); % Next power of 2 from length of L
Y = fft(y3,NFFT)/L;
f = Fs/2*linspace(0,1,NFFT/2);
```



Fourierova Transformácia

Funkcie, ktoré nie sú periodické, ale konečné sa dajú vyjadriť ako integrál váhovaných sínusov a kosínusov

$$F(s) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi xs} dx$$

$$f(x) = \int_{-\infty}^{\infty} F(s) e^{i2\pi xs} ds$$

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dxdy$$

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) e^{j2\pi(ux+vy)} dudv$$

$$e^{j\theta} = \cos\theta + j\sin\theta$$

2D FT a IFT (pre obrazy)

DFT

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

$F(0,0) = ?$

IDFT

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,u)e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

u, v : frekvenčné premenné
x, y : priestorové premenné

2D DFT spektrá

Spektrum (magnitúda), fázový uhol a silové spektrum:

$$|F(u,v)| = \left[R^2(u,v) + I^2(u,v)\right]^{\frac{1}{2}} \quad \text{(spectrum)}$$

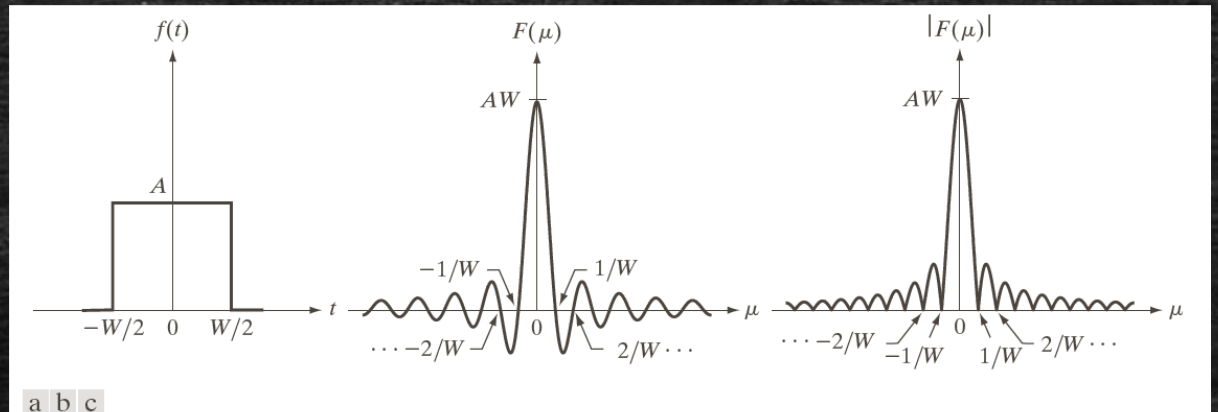
$$\phi(u,v) = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right] \quad \text{(phase angle)}$$

$$P(u,v) = |F(u,v)|^2 = R^2(u,v) + I^2(u,v) \quad \text{(power spectrum)}$$

R(u,v): reálna časť F(u,v)

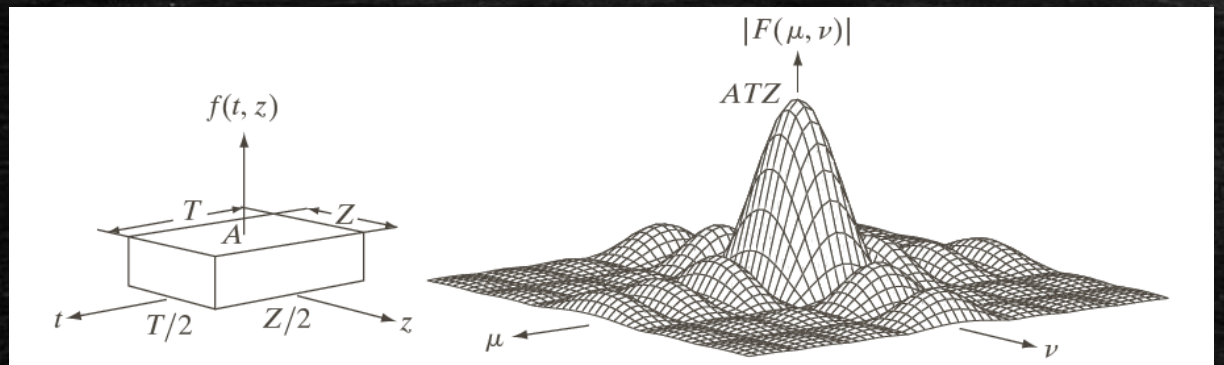
I(u,v): imaginárna časť F(u,v)

Spektrum



a b c

FIGURE 4.4 (a) A simple function; (b) its Fourier transform; and (c) the spectrum. All functions extend to infinity in both directions.



a b

FIGURE 4.13 (a) A 2-D function, and (b) a section of its spectrum (not to scale). The block is longer along the t-axis, so the spectrum is more "contracted" along the μ-axis. Compare with Fig. 4.4.

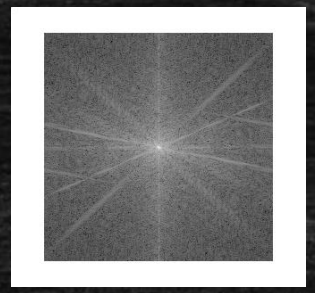
MATLAB fft2, spektrá

cameraman.tif

Vypočítajte a zobrazte:

- Fft2
- Fourier spectrum
- Phase angle
- Power spectrum

real(), imag(),...

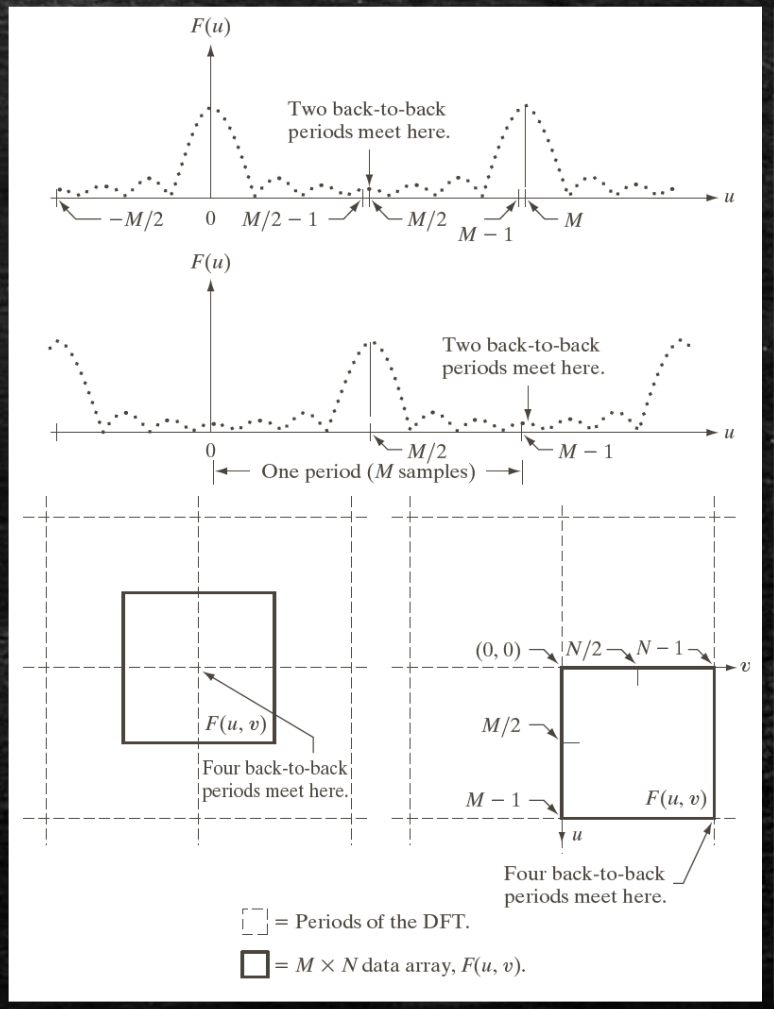


Centrovanie FT

A=reshape(1:25,5,[])

B=fftshift(A)

ifftshift(B)

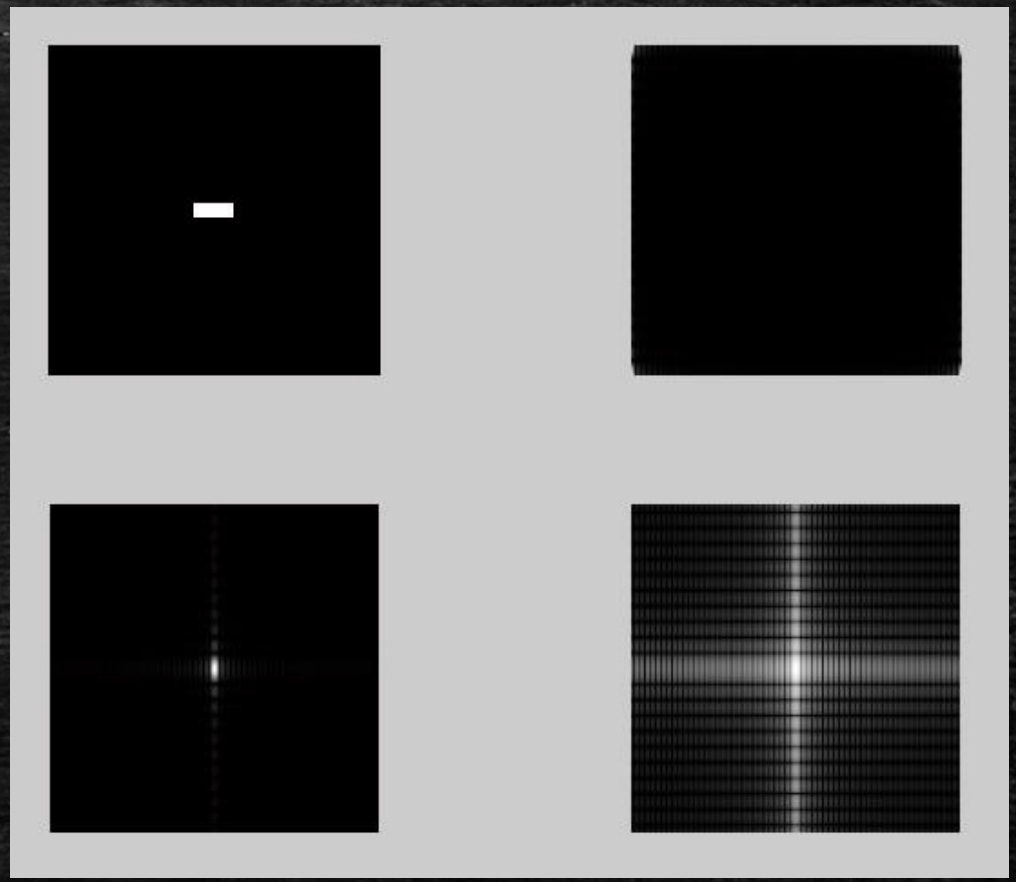


a
b
c d

FIGURE 4.23 Centering the Fourier transform. (a) A 1-D DFT showing an infinite number of periods. (b) Shifted DFT obtained by multiplying $f(x)$ by $(-1)^x$ before computing $F(u)$. (c) A 2-D DFT showing an infinite number of periods. The solid area is the $M \times N$ data array, $F(u, v)$, obtained with Eq. (4.5-15). This array consists of four quarter periods. (d) A Shifted DFT obtained by multiplying $f(x, y)$ by $(-1)^{x+y}$ before computing $F(u, v)$. The data now contains one complete, centered period, as in (b).

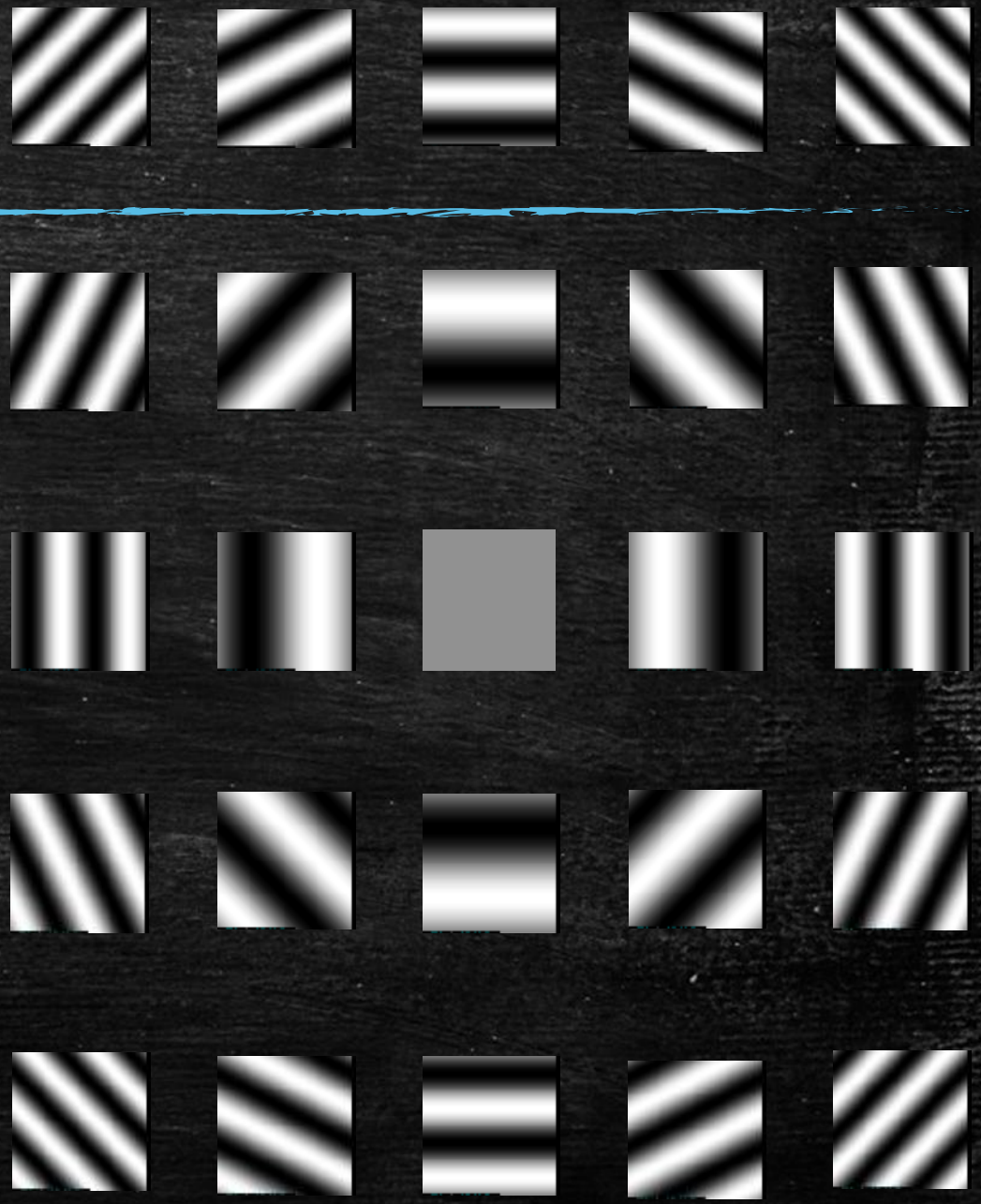
MATLAB fftshift

```
f1(245:265,225:285) = 1;
%%
F = fft2(f1);
S = abs(F);
imshow(S,[]);
%%
Fc = fftshift(F);
S1 = abs(Fc);
imshow(S1,[]);
%%
S2 = log(1+S1);
imshow(S2,[]);
```



Centrovanie FT

Bázové funkcie

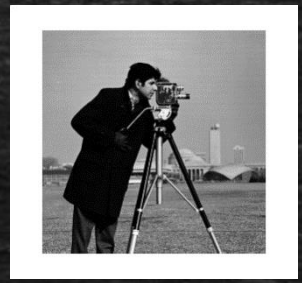


Bázové funkcie

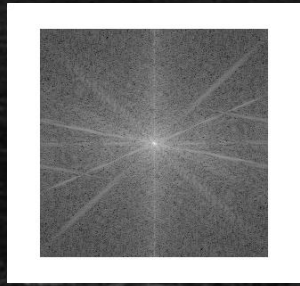
```
wavelength=100;

orientation=135;

g = gabor(wavelength,orientation);

aa=imadjust(real(g.SpatialKernel),[]);

imshow(aa)
```

FFT rekonštrukcia

```
a = imread('cameraman.tif');
figure; imshow(a,[]);
```



```
Fa = fft2(a);
figure; imshow(log(abs(fftshift(Fa))),[]);
```



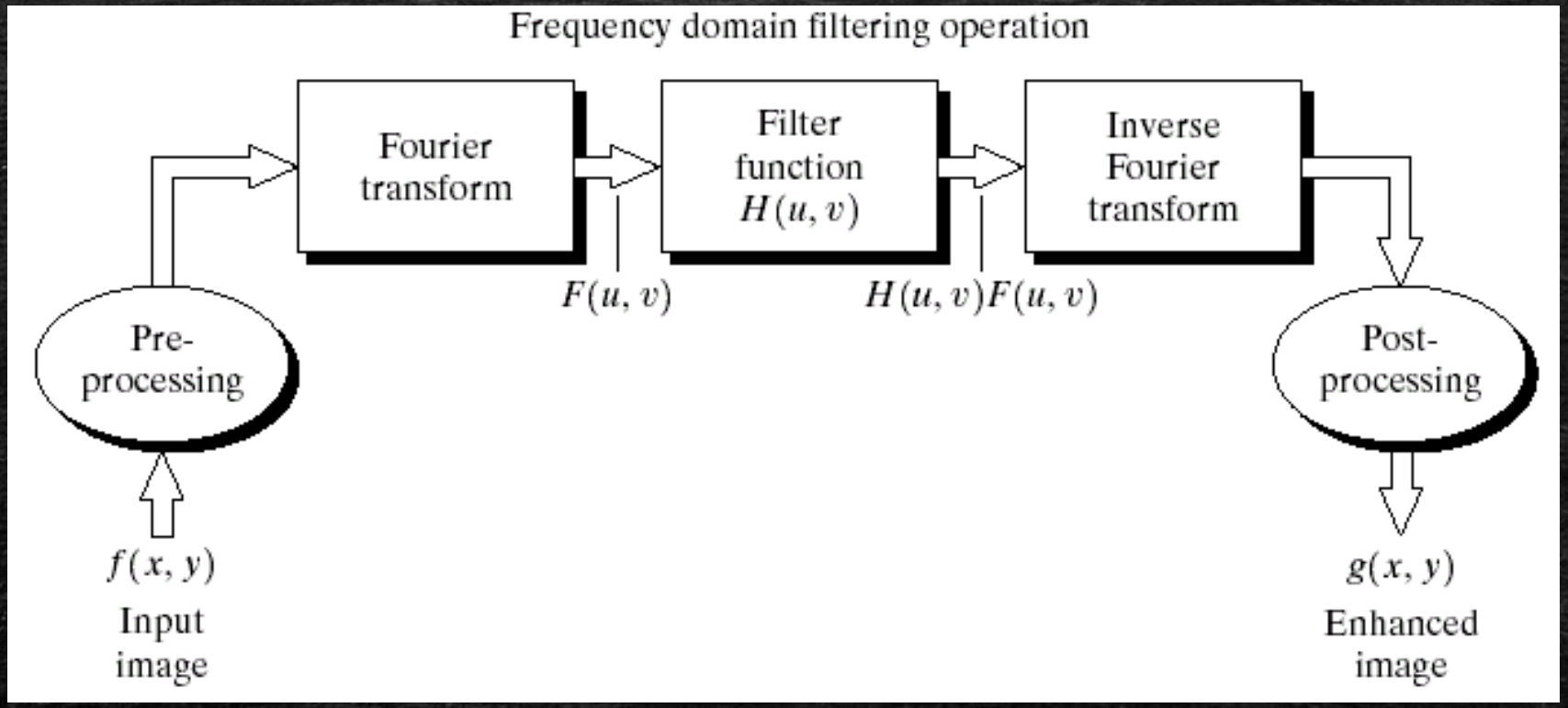
```
b = real(ifft2(Fa));
figure; imshow(b,[]);
```

Konvolučná teoréma

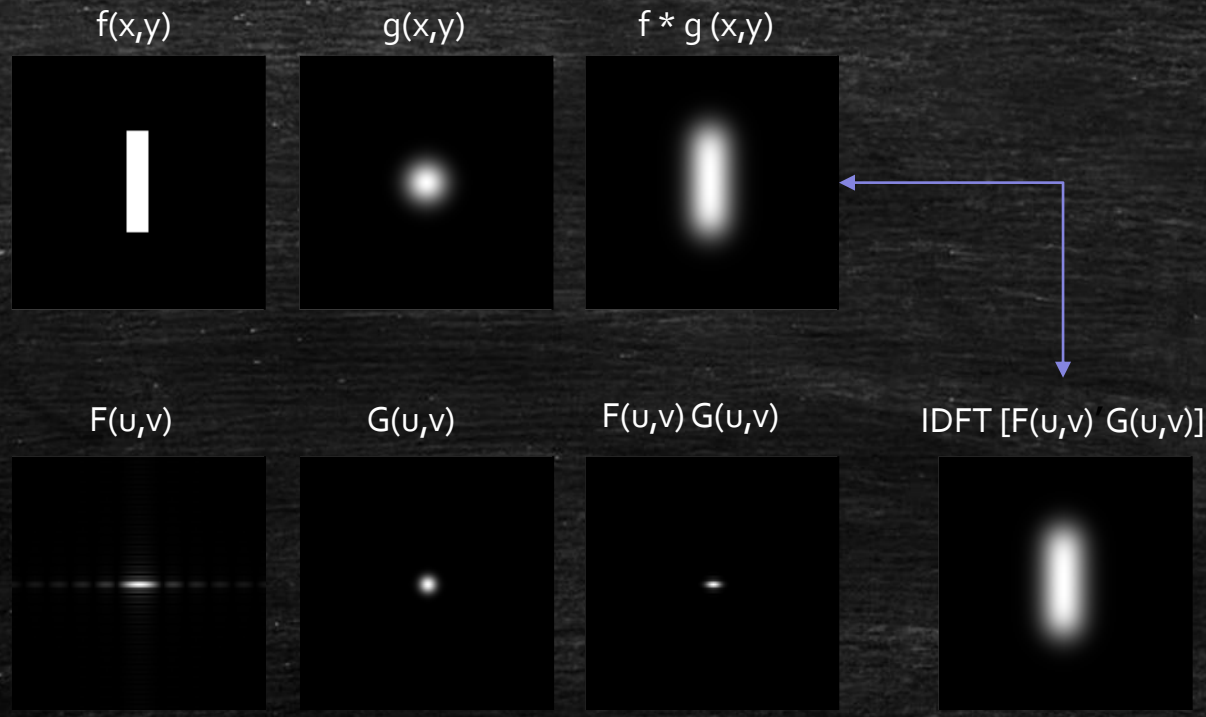
$$f(x,y) * h(x,y) \Leftrightarrow F(u,v)H(u,v)$$

$$f(x,y)h(x,y) \Leftrightarrow F(u,v) * H(u,v)$$

Filtrácia vo frekvenčnej oblasti



Filtrácia vo frekvenčnej oblasti



Frekvenčné filtre

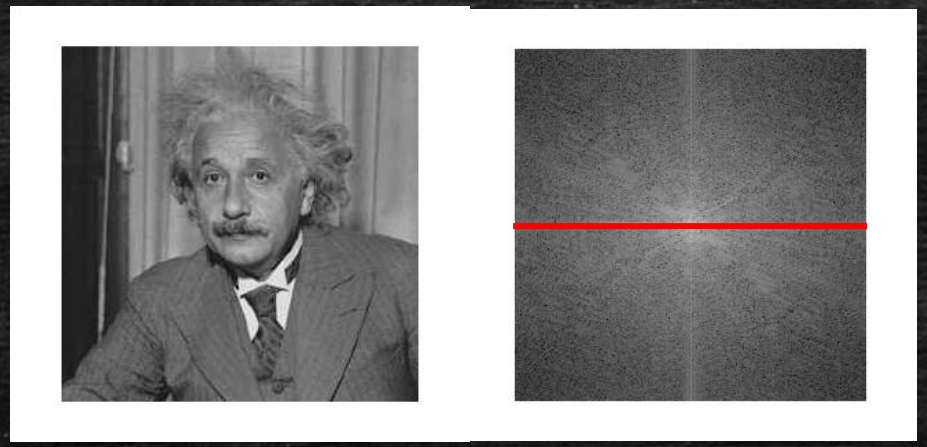
1. Dolnopriepustný filter (rozmazanie)
   Zachová nízke frekvencie

2. Hornopriepustný filter (ostrenie)
   Zachová vysoké frekvencie

3. Pásmový filter Band-pass
   Zachová určité pásmo frekvencií

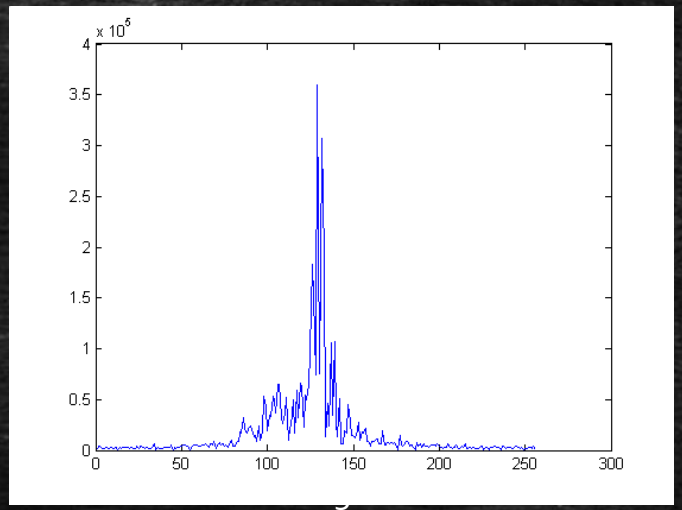
4. Pásmový filter Band-stop
   Potlačí určité pásmo frekvencií

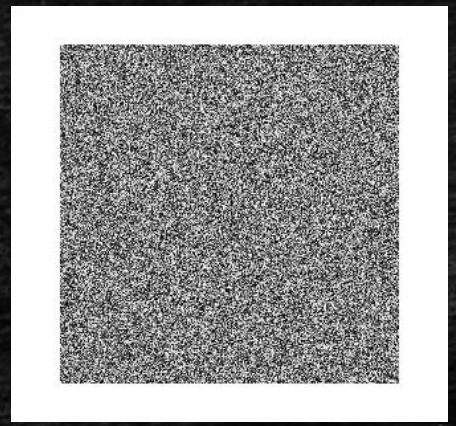
Dolnopriepustný filter



"Einstein"

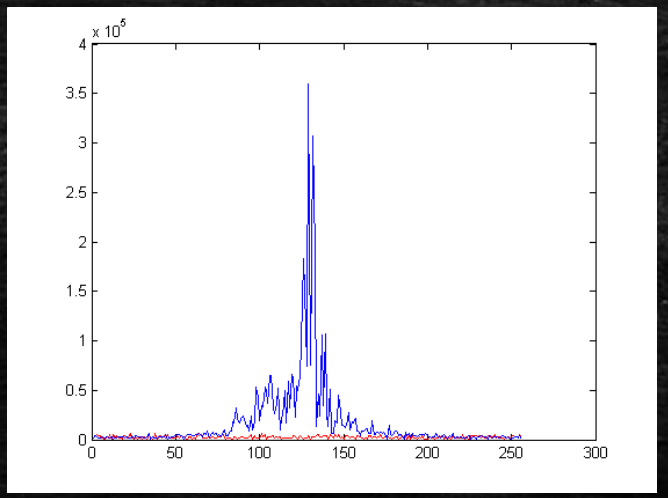
DFT of "Einstein"



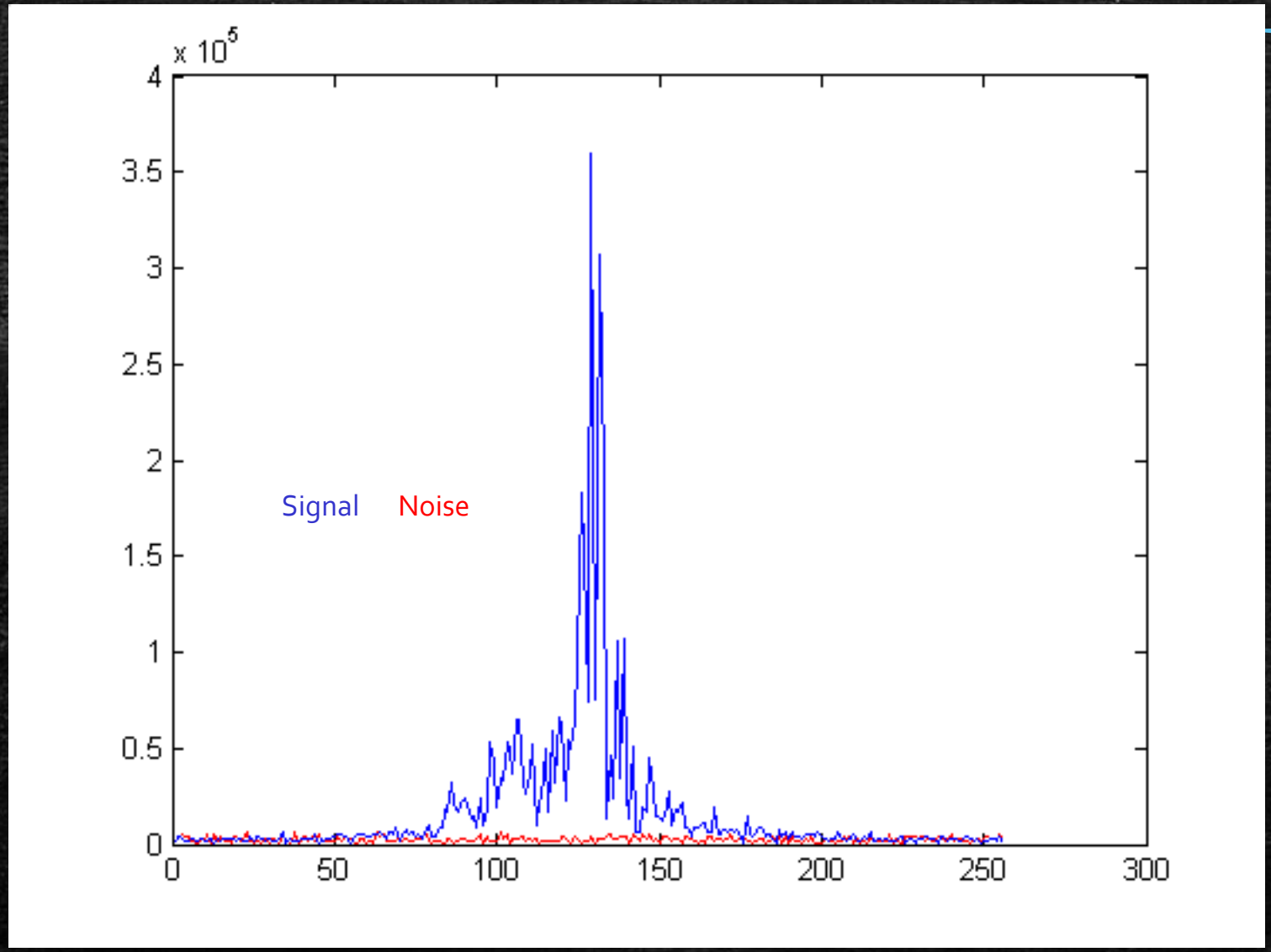


Noise=40*rand(256,256);

Signal vs Noise

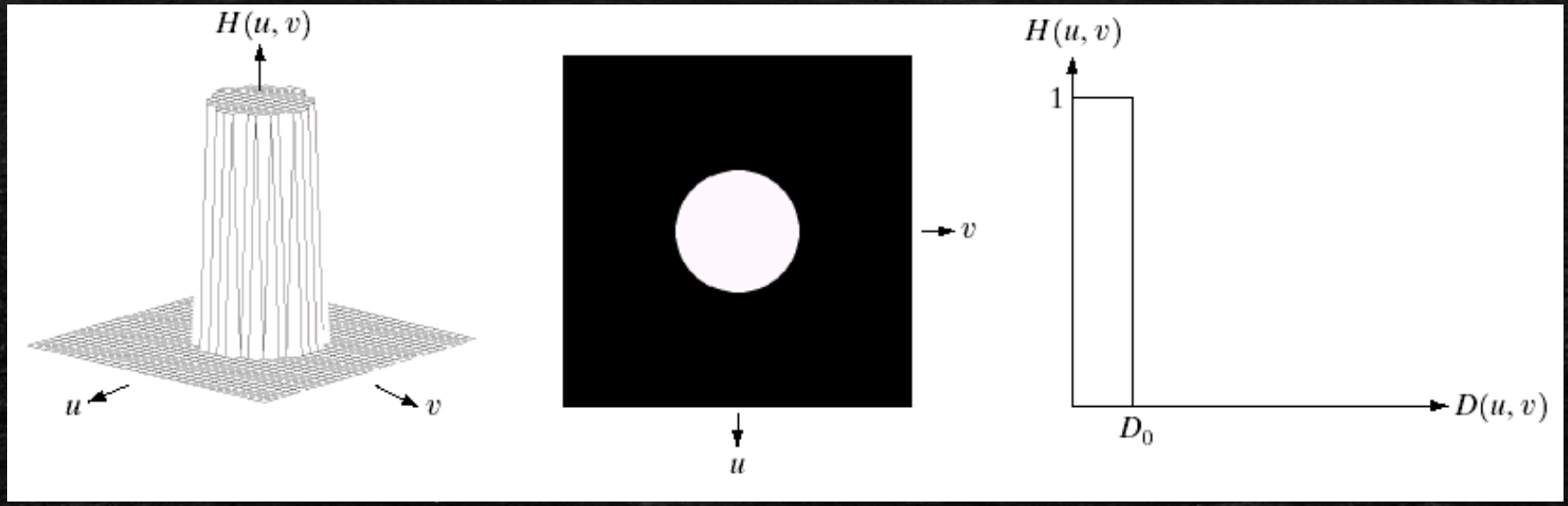


Dolnopriepustný filter



Dolnopriepustný filter: Ideal Lowpass Filters (ILPFs)

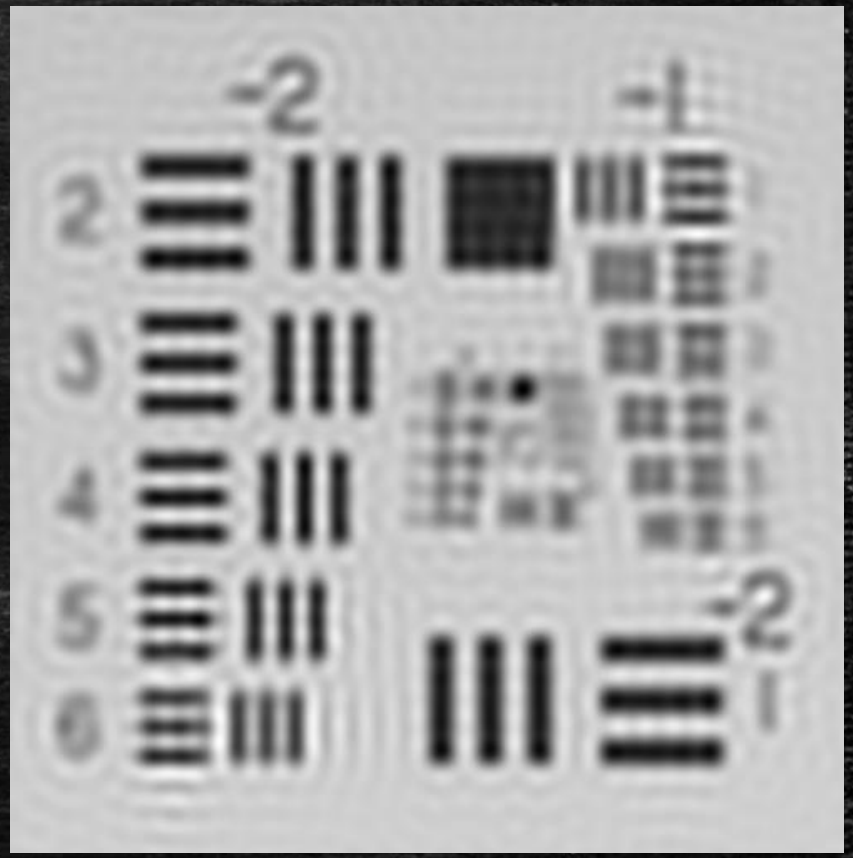
$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \geq D_0 \\ 1 & \text{if } D(u,v) < D_0 \end{cases}$$



ILPF problém

Ideálny = dajú sa presne vymedziť frekvencie

Problém: rozvlnenie rekonštruovaného obrazu

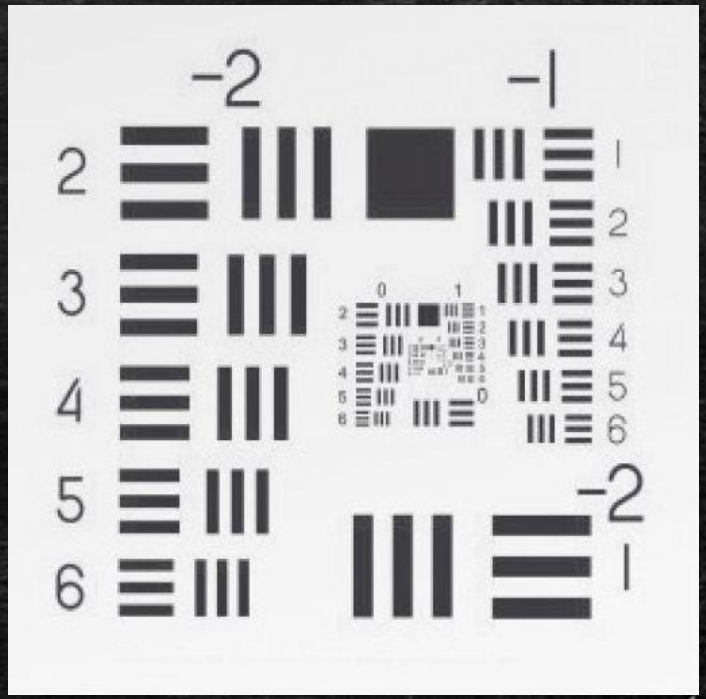


MATLAB - ILPF

Aplikujte ILPF na timthumb.jpg

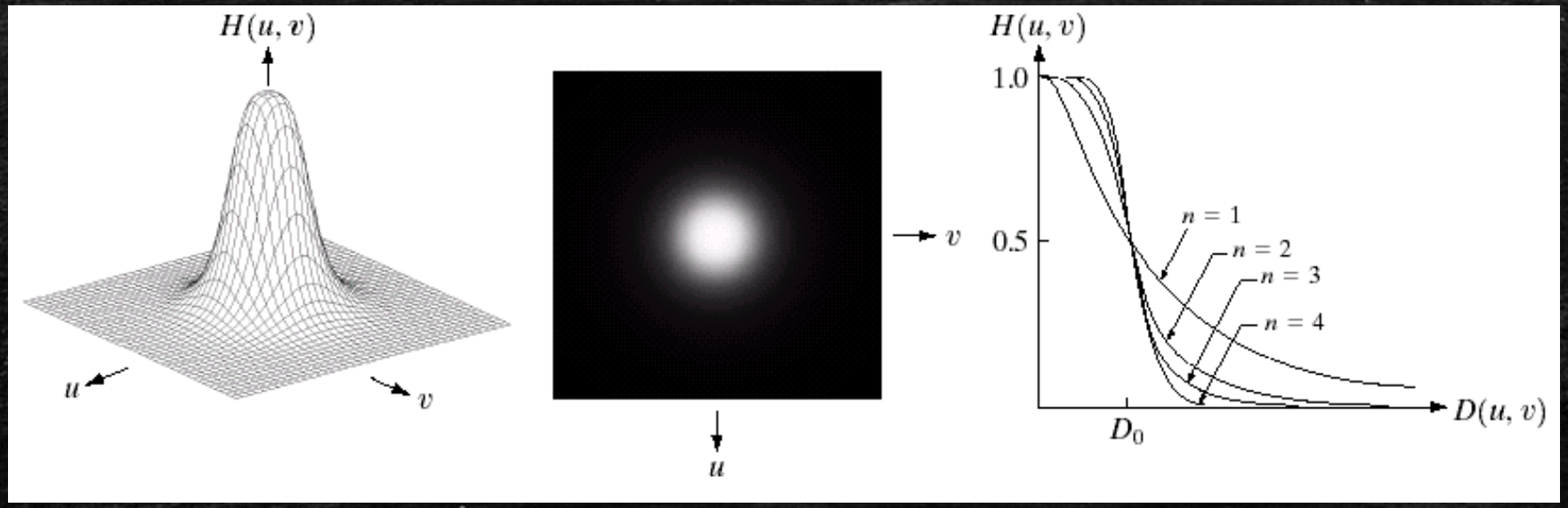
Vyskúšajte rôzne D_0

```
H=zeros(512);
radius=32;
h = imbinarize(fspecial('disk',radius),0);
H(256-radius:256+radius,256-radius:256+radius)=h;
figure, imshow(H)
```



Butterworth Lowpass Filters (BLPFs)

BLPF with order n $$H(u,v) = \frac{1}{1+\left[D(u,v)/D_0\right]^{2n}}$$



MATLAB – BLPF

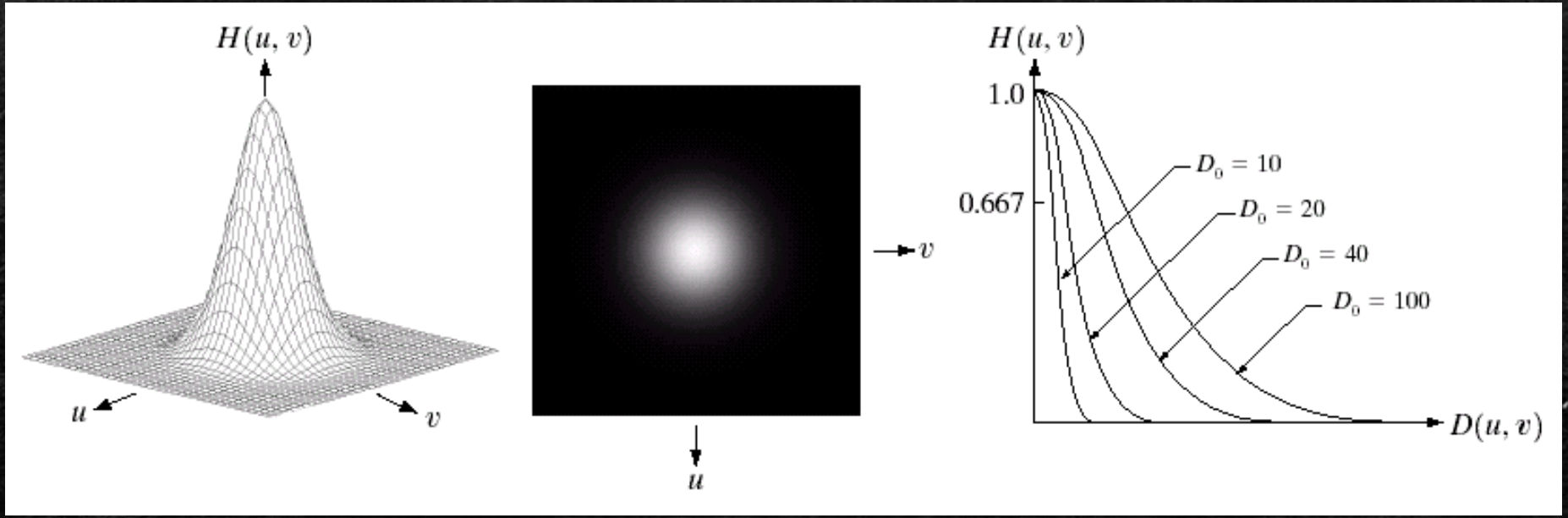
Aplikujte BLPF na timthumb.jpg

Vyskúšajte rôzne n, D_0

```
x=-256:255;
[X,Y] = meshgrid(x,x);
D=sqrt(X.^2 + Y.^2);
H=...
```

Gaussian Lowpass Filters (GLPFs)

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$



MATLAB – GLPF

Aplikujte GLPF na timthumb.jpg

Vyskúšajte rôzne D_0

H=fspecial('gaussian',hsize,sigma)

High pass filters - sharpening

Ideal highpass filter

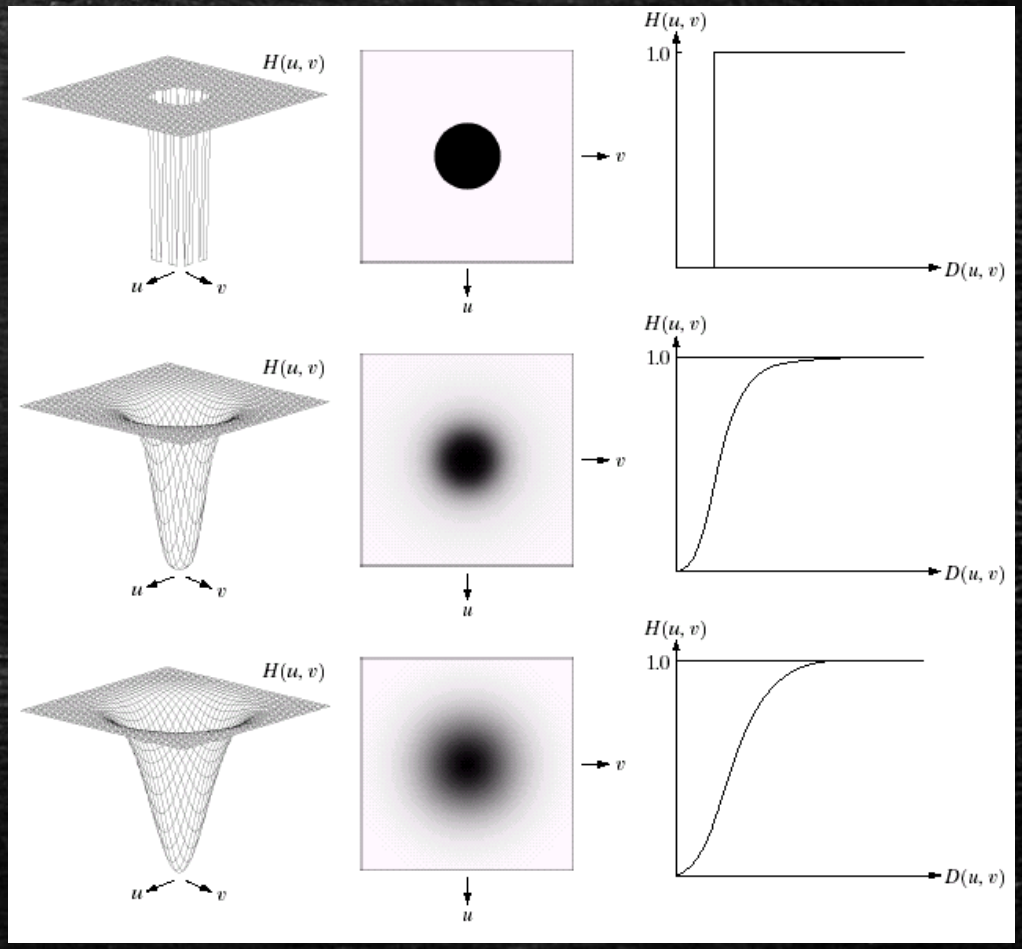
$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$$

Butterworth highpass filter

$$H(u,v) = \frac{1}{1 + \left[D_0 / D(u,v) \right]^{2n}}$$

Gaussian highpass filter

$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$$

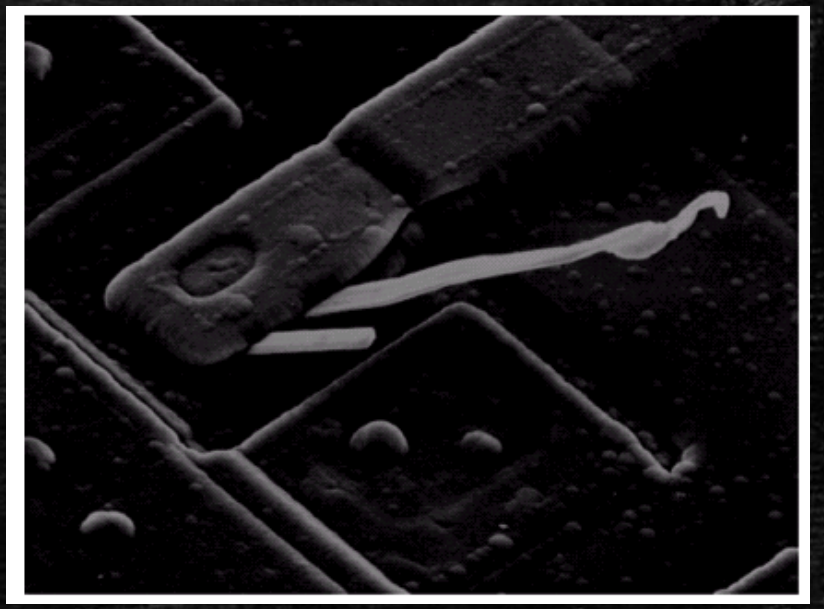


Notch filter (band stop, narrow band)

$$H(u,v) = \begin{cases} 0 & \text{if } (u,v) = (M/2, N/2) \\ 1 & \text{otherwise.} \end{cases}$$

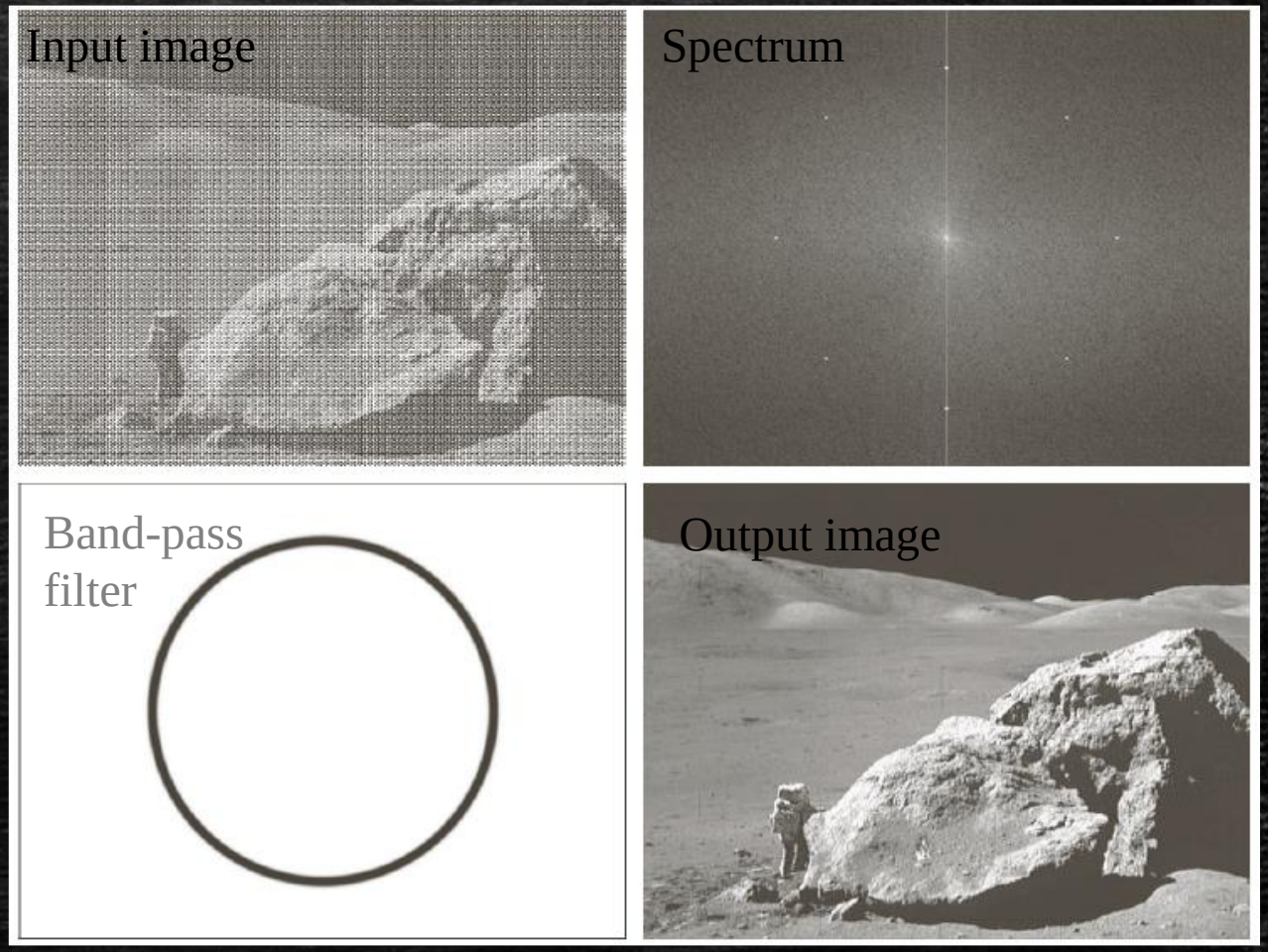
Nastaví F(o,o) na nulu. Priemerná intenzita obrazu bude o.

Dá sa použiť na potlačenie konkrétnej frekvencie a jej okolia (ako ideal, B alebo G filter)

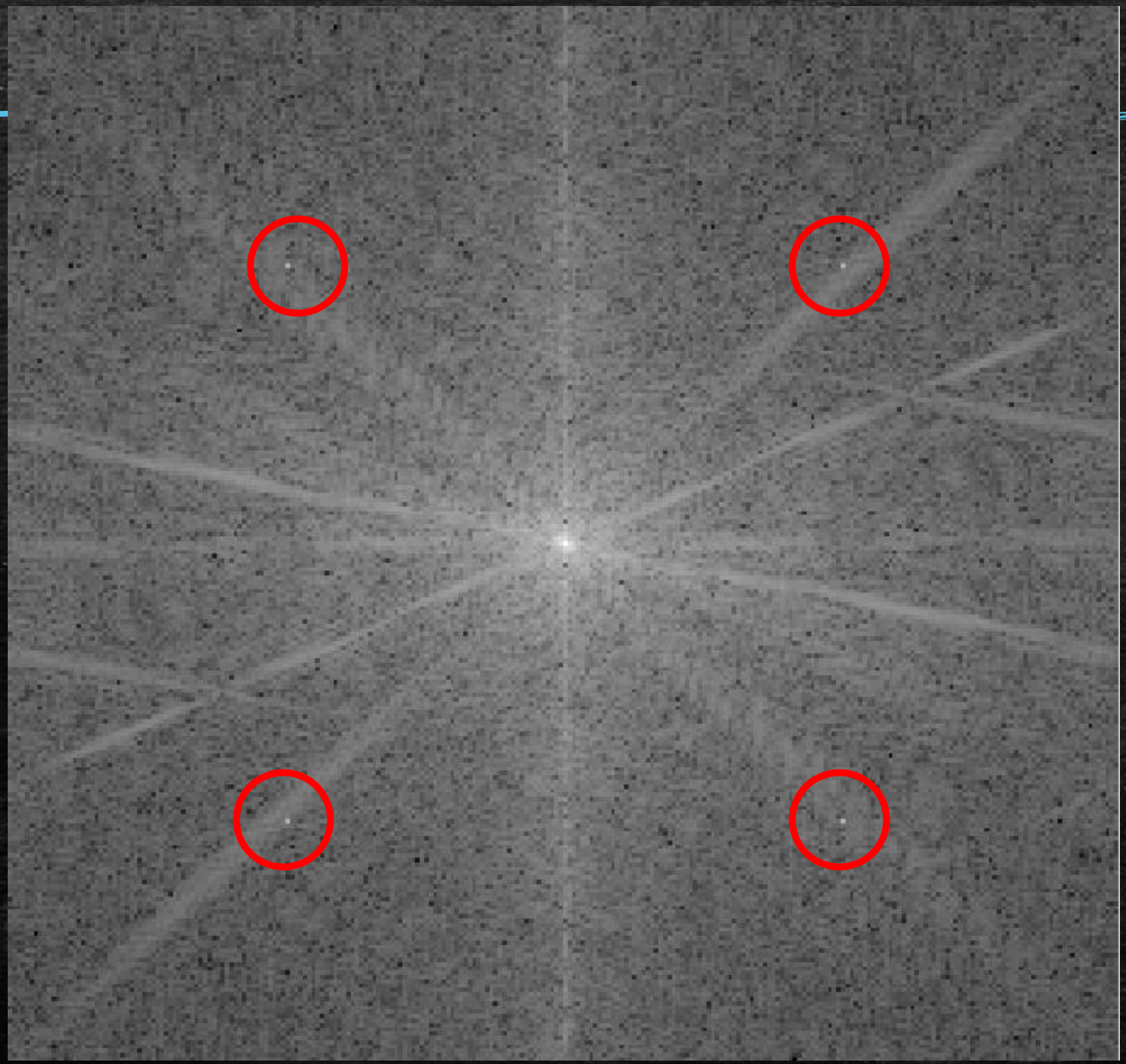


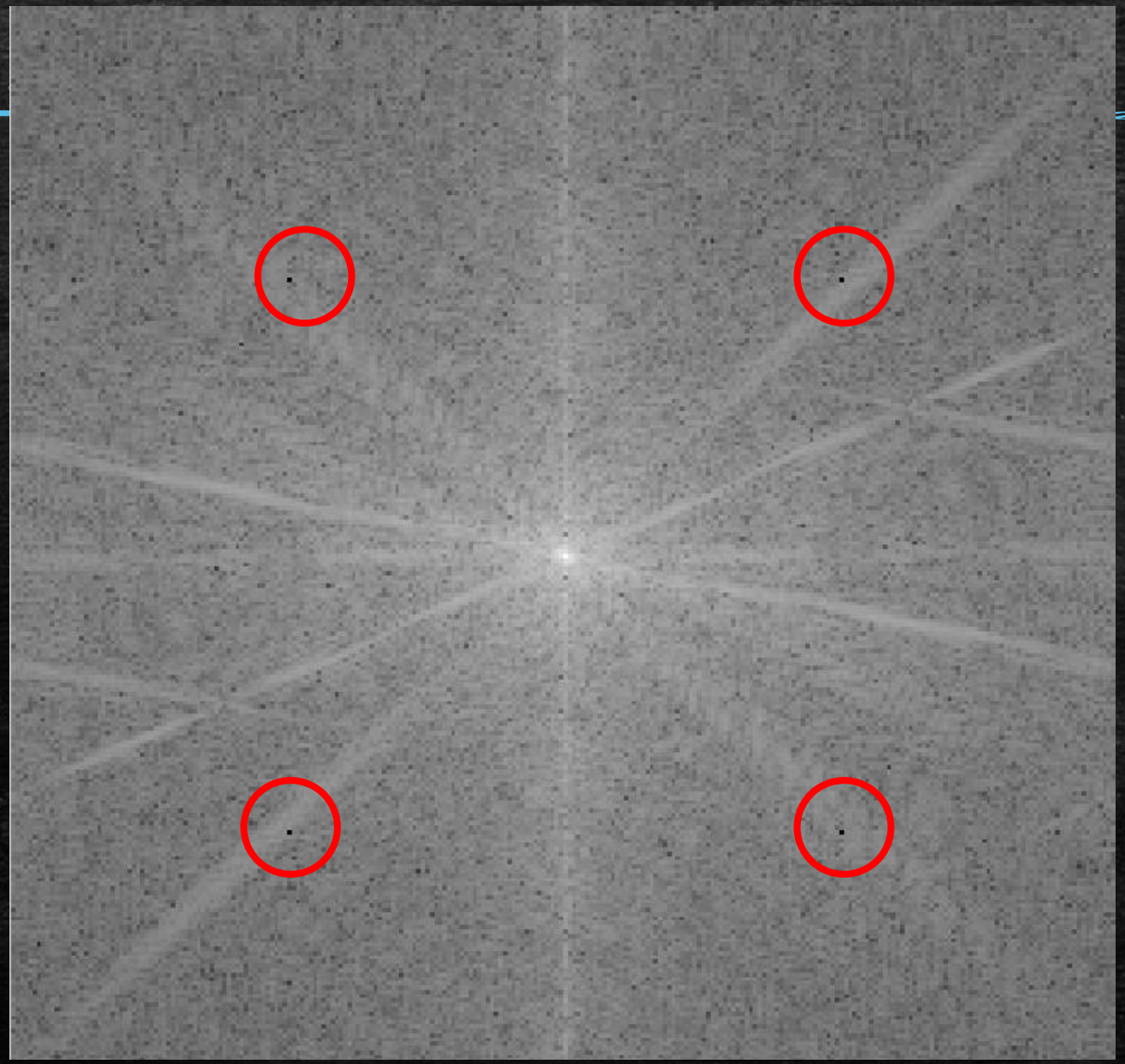
Band stop, band pass

H=1-(D>D1&D<=D2);



Other filters





MATLAB – noise reduction

```
y=double(checkerboard(2,64,64)>0.5);

a=double(imread('cameraman.tif'))+10*y;

F = fft2(a);
FF=fftshift(F);

FF(?,?)=0;

IF=fftshift(FF);
b=ifft2(IF);
figure, imshow(real(b),[])
```