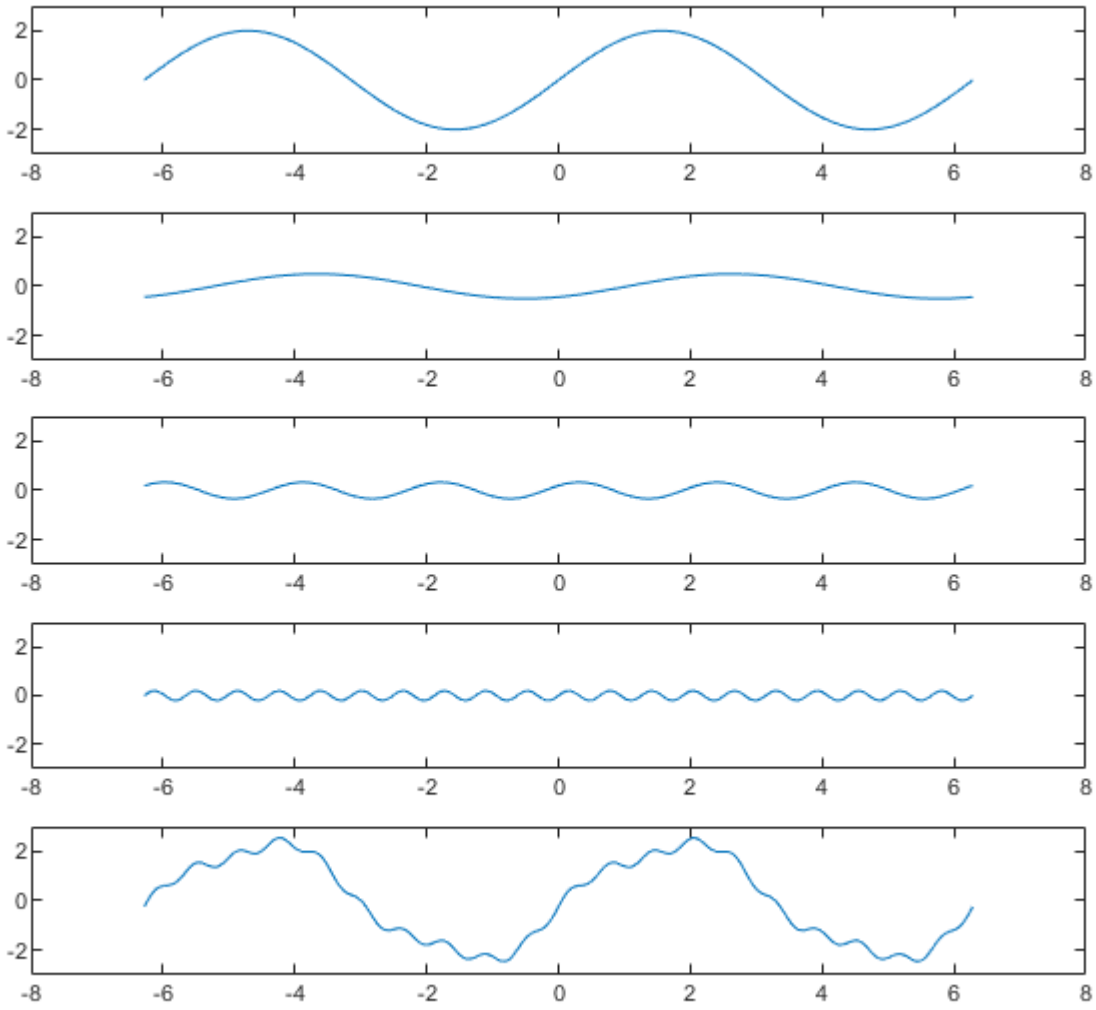
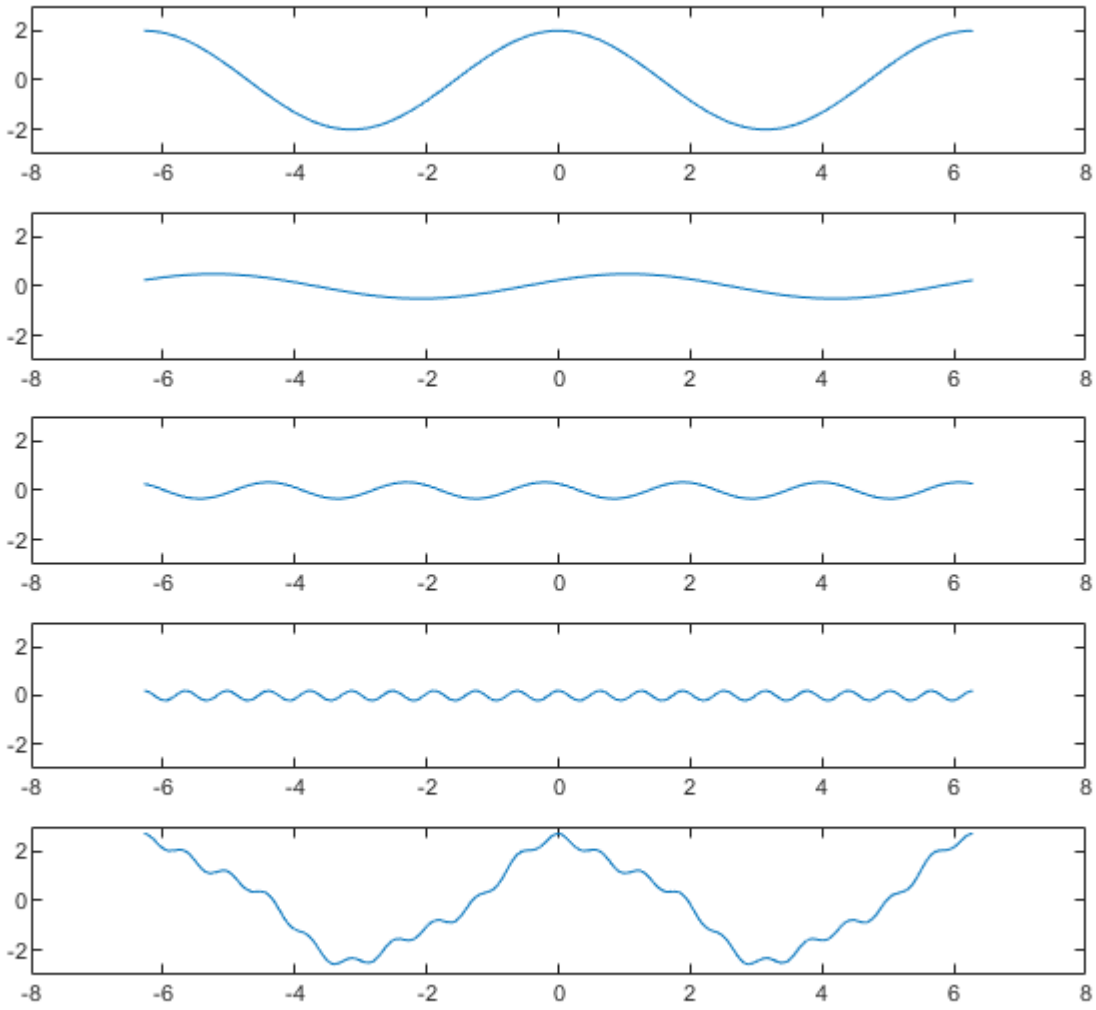
Cvičení 8

Součet funkcí

```
n = 300; % pocet bodu
% vytvoření vektoru o n hodnotách pravidelně rozmístěné mezi -2pi a 2pi
x = linspace(-2*pi,2*pi,n)';

% jednotlive funkce
y1s = 2*sin(x);
y2s = sin(x-pi/3)/2;
y3s = sin(3*x+pi/5)/3;
y4s = sin(10*x)/5;

subplot(5,1,1), plot(x,y1s);
ylim([-3,3]);
subplot(5,1,2), plot(x,y2s);
ylim([-3,3]);
subplot(5,1,3), plot(x,y3s);
ylim([-3,3]);
subplot(5,1,4), plot(x,y4s);
ylim([-3,3]);
subplot(5,1,5), plot(x,y1s+y2s+y3s+y4s);
ylim([-3,3]);
```




```
y1c = 2*cos(x);
y2c = cos(x-pi/3)/2;
y3c = cos(3*x+pi/5)/3;
y4c = cos(10*x)/5;

subplot(5,1,1), plot(x,y1c);
ylim([-3,3]);
subplot(5,1,2), plot(x,y2c);
ylim([-3,3]);
subplot(5,1,3), plot(x,y3c);
ylim([-3,3]);
subplot(5,1,4), plot(x,y4c);
ylim([-3,3]);
subplot(5,1,5), plot(x,y1c+y2c+y3c+y4c);
ylim([-3,3]);
```

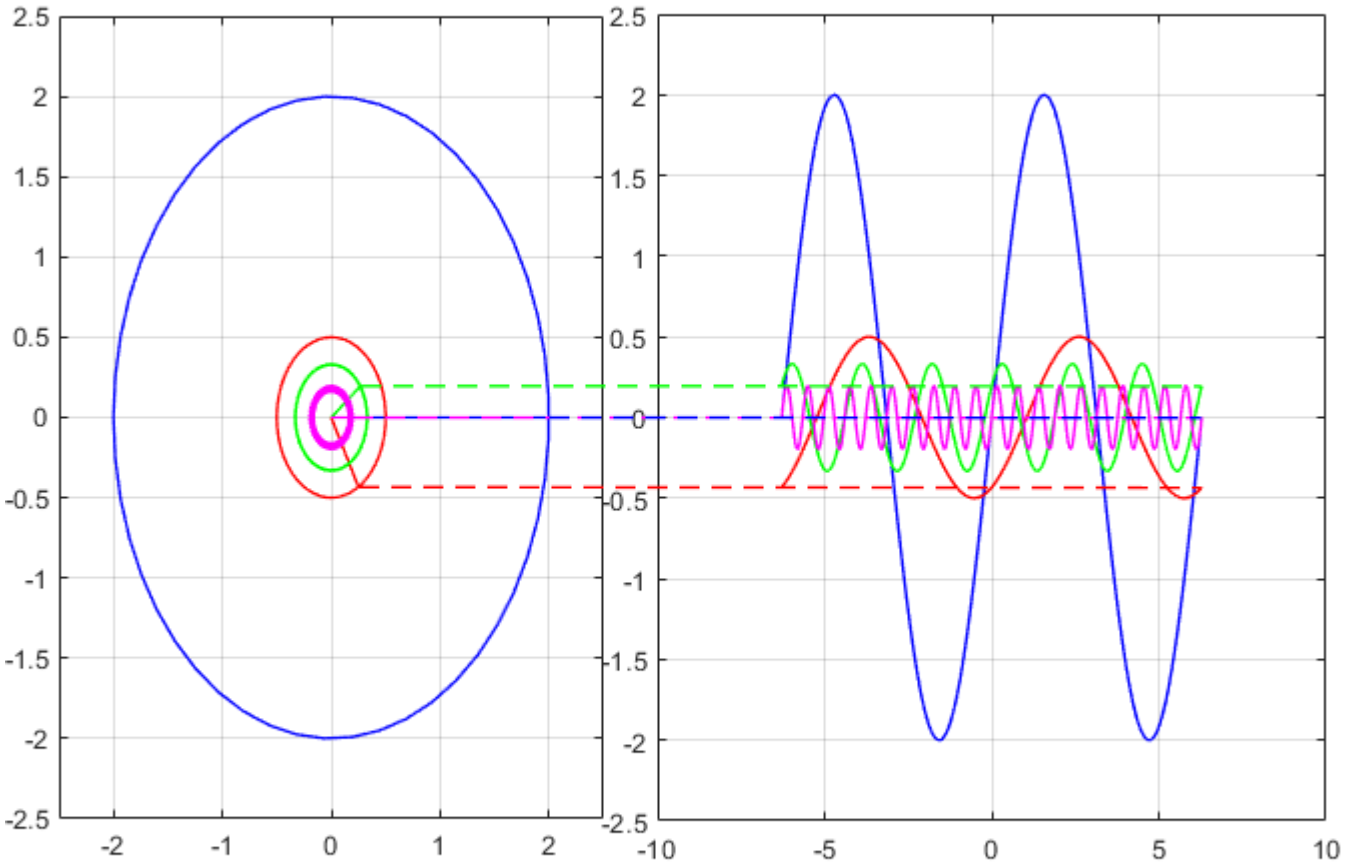


Animace

```
nt = 50; %pocet bodu na kruznici
theta  =linspace(-pi,pi,nt)';   % diskretni body na kruznici

% Kruznice
kx1 = 2*cos(theta);
ky1 = 2*sin(theta);

kx2 = cos(theta-pi/3)/2;
ky2 = sin(theta-pi/3)/2;

kx3 = cos(3*theta+pi/5)/3;
ky3 = sin(3*theta+pi/5)/3;

kx4 = cos(10*theta)/5;
```

```
ky4 = sin(10*theta)/5;

Lx=length(x);
Lw=1; %sirka cary
Fs=12;

for i=1:Lx

    f1=figure (2); clf;

    sp1=subplot(1,2,1);
    %  Funkce 1
    plot(kx1,ky1,'LineWidth',Lw,'Color','b'); hold on; grid on;
    line([0 y1c(i)],[0 y1s(i)],'Color','b','LineWidth',Lw,'LineSmoothing','on');

    set(sp1,'Position',[0.0400    0.1800    0.4    0.677]);
    xlim([-2.5 2.5]); ylim([-2.5 2.5])


line(y1c(i),y1s(i),10,'LineStyle','-','MarkerSize',8,'MarkerFaceColor','b','color','
b')

     [xf1, yf1] = ds2nfu(sp1,y1c(i),y1s(i));

    %  Funkce 2
    plot(kx2,ky2,'LineWidth',Lw,'Color','r'); hold on;
    line([0 y2c(i)],[0 y2s(i)],'Color','r','LineWidth',Lw,'LineSmoothing','on');

line(y2c(i),y2s(i),10,'LineStyle','-','MarkerSize',8,'MarkerFaceColor','r','color','
r')

    [xf2, yf2] = ds2nfu(sp1,y2c(i),y2s(i));

    %  Funkce 3
    plot(kx3,ky3,'LineWidth',Lw,'Color','g'); hold on;
    line([0 y3c(i)],[0 y3s(i)],'Color','g','LineWidth',Lw,'LineSmoothing','on');

line(y3c(i),y3s(i),10,'LineStyle','-','MarkerSize',8,'MarkerFaceColor','g','color','
g')

    [xf3, yf3] = ds2nfu(sp1,y3c(i),y3s(i));

    %  Funkce 4
    plot(kx4,ky4,'LineWidth',Lw,'Color','m'); hold on;
    line([0 y4c(i)],[0 y4s(i)],'Color','m','LineWidth',Lw,'LineSmoothing','on');

line(y4c(i),y4s(i),10,'LineStyle','-','MarkerSize',8,'MarkerFaceColor','m','color','
m')

    [xf4, yf4] = ds2nfu(sp1,y4c(i),y4s(i));
```

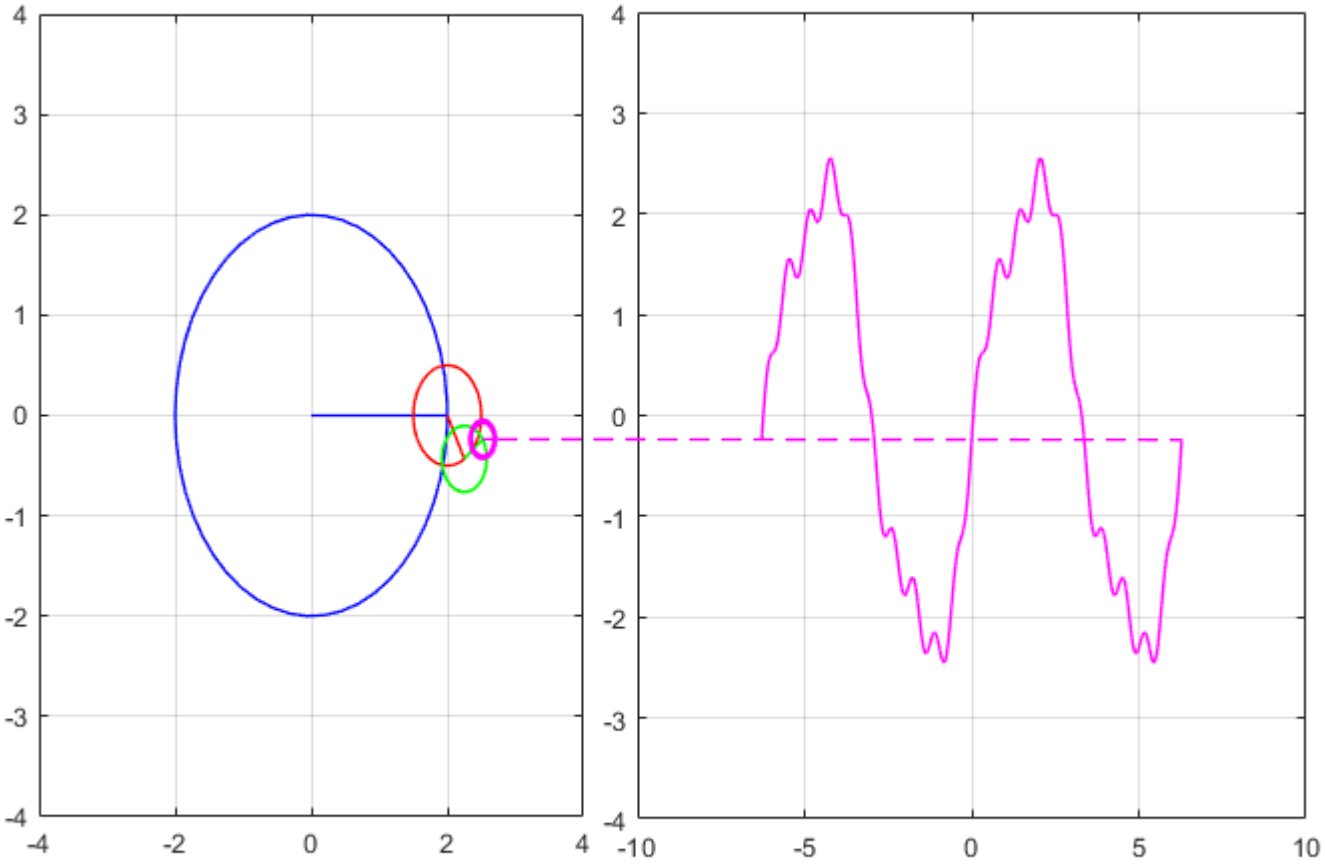
```
    sp2=subplot(1,2,2);
    %  Funkce 1

    plot(x(1:i),y1s(1:i),'LineWidth',Lw,'Color','b'); hold on; grid on;
    ylim([-2.5 2.5]); xlim([-10 10])
    set(sp2,'Position',[0.48    0.178200    0.49    0.680]);

    [xg1, yg1] = ds2nfu(x(i),y1s(i));
    annotation('line',[xf1 xg1],[yf1
yg1],'color','b','LineStyle','--','LineWidth',Lw);

    % Funkce 2
    plot(sp2,x(1:i),y2s(1:i),'g','LineWidth',Lw,'Color','r'); hold on; grid on;

    [xg2, yg2] = ds2nfu(x(i),y2s(i));
    annotation('line',[xf2 xg2],[yf2
yg2],'color','r','LineStyle','--','LineWidth',Lw);

    % Funkce 3
    plot(sp2,x(1:i),y3s(1:i),'g','LineWidth',Lw,'Color','g'); hold on; grid on;

    [xg3, yg3] = ds2nfu(x(i),y3s(i));
    annotation('line',[xf3 xg3],[yf3
yg3],'color','g','LineStyle','--','LineWidth',Lw);

    % Funkce 4
    plot(sp2,x(1:i),y4s(1:i),'g','LineWidth',Lw,'Color','m'); hold on; grid on;

    [xg4, yg4] = ds2nfu(x(i),y4s(i));
    annotation('line',[xf4 xg4],[yf4
yg4],'color','m','LineStyle','--','LineWidth',Lw);

    pause(0.01);
end
```



Animace dohromady

```
for i=1:length(x)

    f1=figure (2); clf;

    sp1=subplot(1,2,1);
    %  Funkce 1
    plot(kx1,ky1,'LineWidth',Lw,'Color','b'); hold on; grid on;
    line([0 y1c(i)],[0 y1s(i)],'Color','b','LineWidth',Lw,'LineSmoothing','on');

    set(sp1,'Position',[0.0400    0.1800    0.4    0.677]);
    xlim([-4 4]); ylim([-4 4])
```

```
line(y1c(i),y1s(i),10,'LineStyle','-','MarkerSize',8,'MarkerFaceColor','b','color','
b')

    %  Funkce 2
    plot(kx2+y1c(i),ky2+y1s(i),'LineWidth',Lw,'Color','r'); hold on;
    line(y1c(i) + [0 y2c(i)],y1s(i) + [0
y2s(i)],'Color','r','LineWidth',Lw,'LineSmoothing','on');

   %  Funkce 3
    plot(kx3+y1c(i)+y2c(i),ky3+y1s(i)+y2s(i),'LineWidth',Lw,'Color','g'); hold on;
    line(y1c(i) + y2c(i) + [0 y3c(i)],y1s(i) + y2s(i) + [0
y3s(i)],'Color','g','LineWidth',Lw,'LineSmoothing','on');

 %  Funkce 4
    plot(kx4+y1c(i)+y2c(i)+y3c(i),ky4+y1s(i)+y2s(i)
+y3s(i),'LineWidth',Lw,'Color','m'); hold on;
    line(y1c(i) + y2c(i) + y3c(i) + [0 y4c(i)],y1s(i) + y2s(i) + y3s(i) +  [0
y4s(i)],'Color','m','LineWidth',Lw,'LineSmoothing','on');

    [xf4, yf4] = ds2nfu(y1c(i)+y2c(i)+y3c(i)+y4c(i),y1s(i)+y2s(i)+y3s(i)+y4s(i));

    sp2=subplot(1,2,2);
    %  dohromady

    plot(x(1:i),y1s(1:i)+y2s(1:i)+y3s(1:i)+y4s(1:i),'LineWidth',Lw,'Color','m');
hold on; grid on;

    ylim([-4 4]); xlim([-10 10])
    set(sp2,'Position',[0.48    0.178200    0.49    0.680]);

     [xgt, ygt] = ds2nfu(x(i),y1s(i)+y2s(i)+y3s(i)+y4s(i));
    annotation('line',[xf4 xgt],[yf4
ygt],'color','m','LineStyle','--','LineWidth',Lw);

    pause(0.01);
end
```



Fourierova řada

Funkce: `Fseries.m` and `Fseriesval.m`

`[a,b] = Fseries(X,Y,n)`

fits an nth-order Fourier expansion of the form

`y = a_0/2 + Sum_k[ a_k cos(kx) + b_k sin(kx) ]`

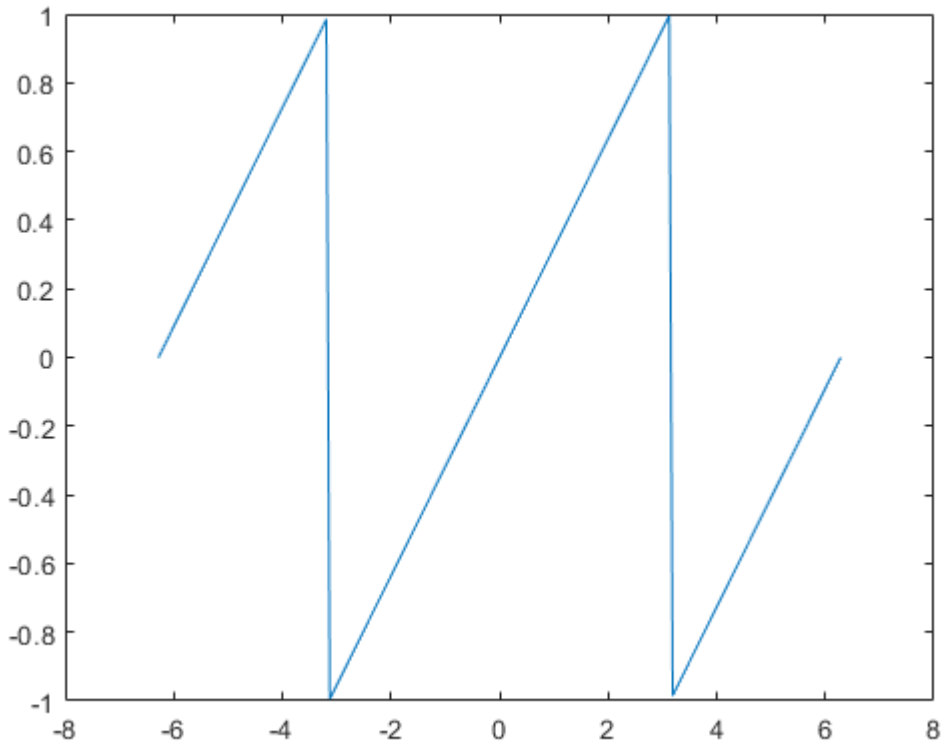
to the data in the vectors `X` a `Y`, using a least-squares fit.

`Y = Fseriesval(a,b,X)`

evaluates the Fourier series defined by the coefficients a and b at the values in the vector X.

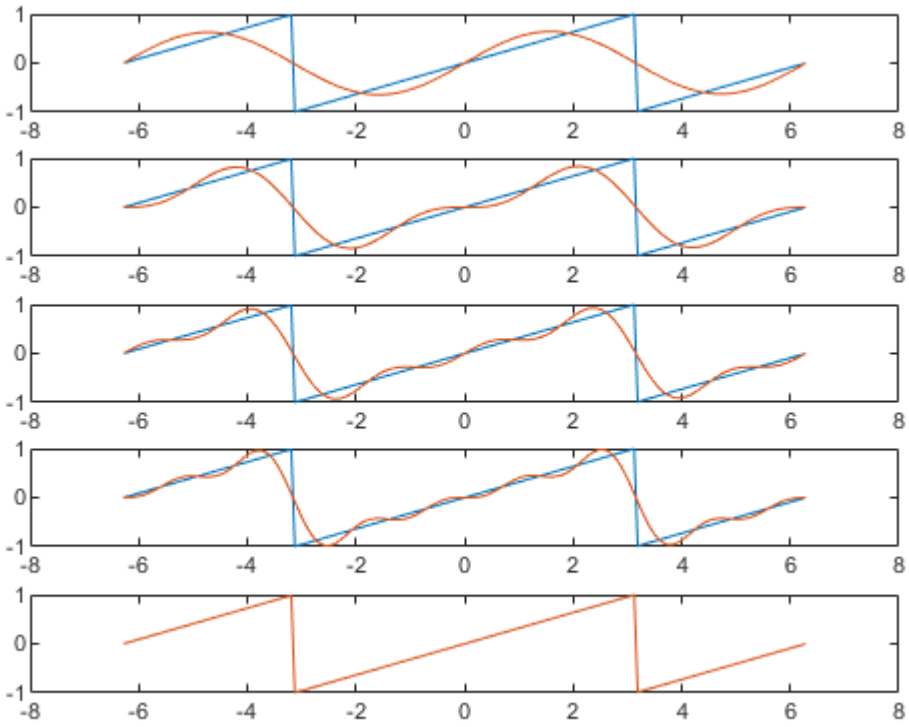
Příklad 1:

```
% Generování dat
x = linspace(-2*pi,2*pi,200)'; % vytvoření vektoru o 50 hodnotách pravidelně
rozmístěné mezi -2pi a 2pi
y = sawtooth(x + pi); % výpočet funkčních hodnot pomcí funkce sawtooth
% alternativní výpočet bez použití funkce sawtooth
%y = mod(x + pi,2*pi) - pi;
figure, plot(x,y);
```



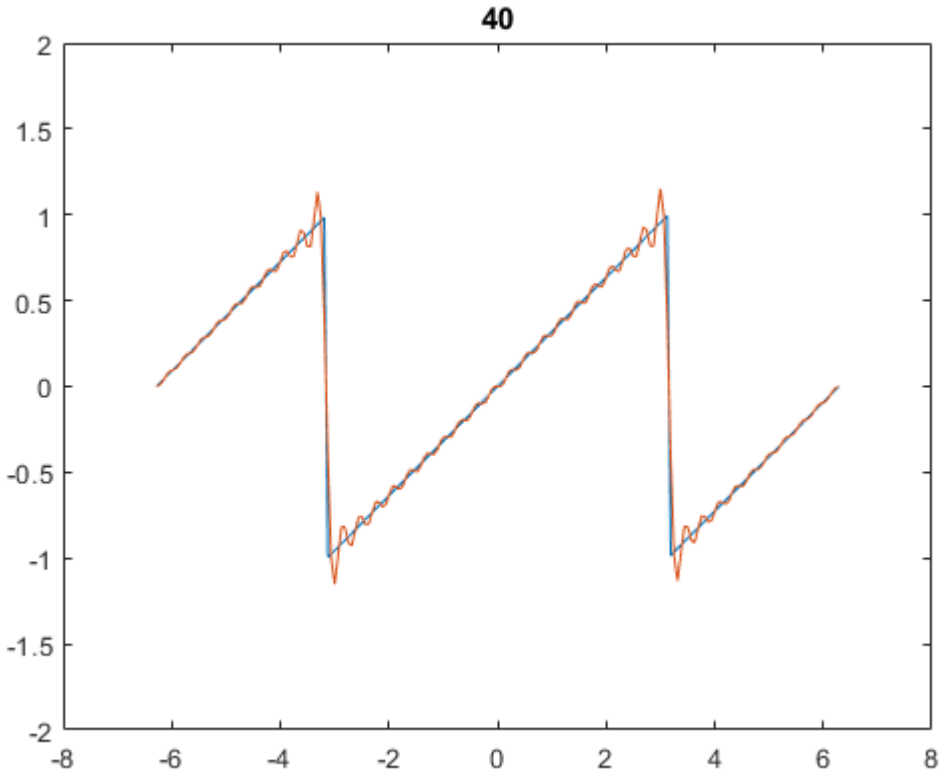

```
% řády
n = [2, 4, 6, 8, 100];

% pro všechny řády spočítáme rozvoj pomocí Fseries
for i = 1 : size(n,2)
    [a,b,yfit] = Fseries(x,y,n(i));
    subplot(size(n,2),1,i)
    plot(x,y,x,yfit);
end
```

```
Warning: Rank deficient, rank = 199, tol =  4.474240e-13.
```



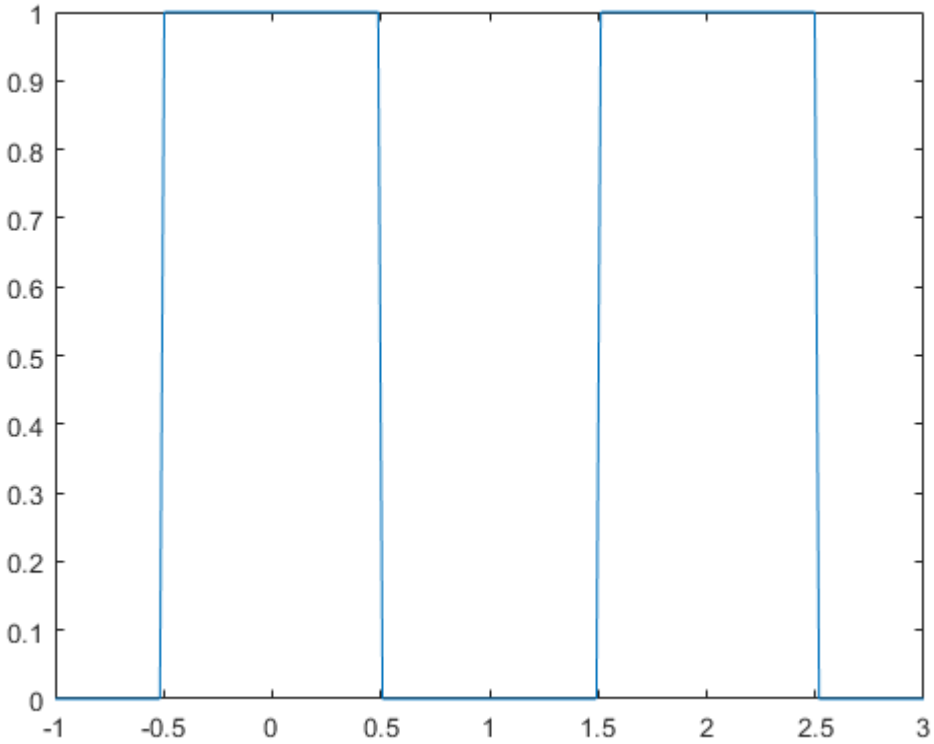
Příklad 1 animace

```
% pro všechny řády spočítáme rozvoj pomocí Fseries
figure,
for i = 2 : 2 : 40
    [a,b,yfit] = Fseries(x,y,i);
    pause(0.3);
    plot(x,y,x,yfit);
    title(num2str(i));
    ylim([-2,2]);
end
```



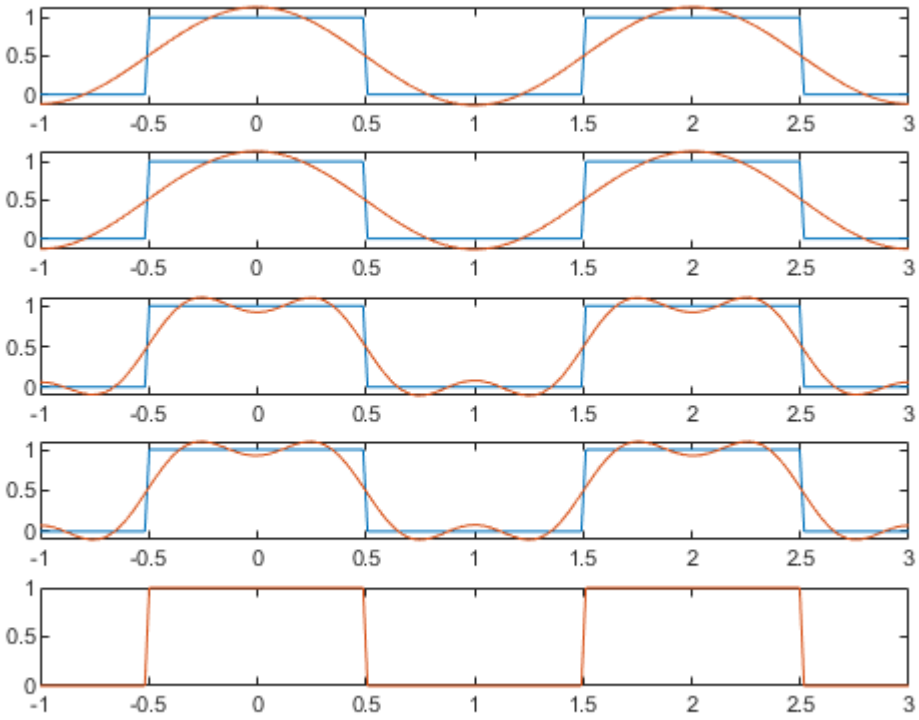
Příklad 2:

```
% Generování dat
x = linspace(-1,3,200)';
y =  double(or(and(x < 2.5, x >= 1.5),and(x< 0.5, x >= -0.5)));% výpočet funkčních
hodnot
figure, plot(x,y);
```



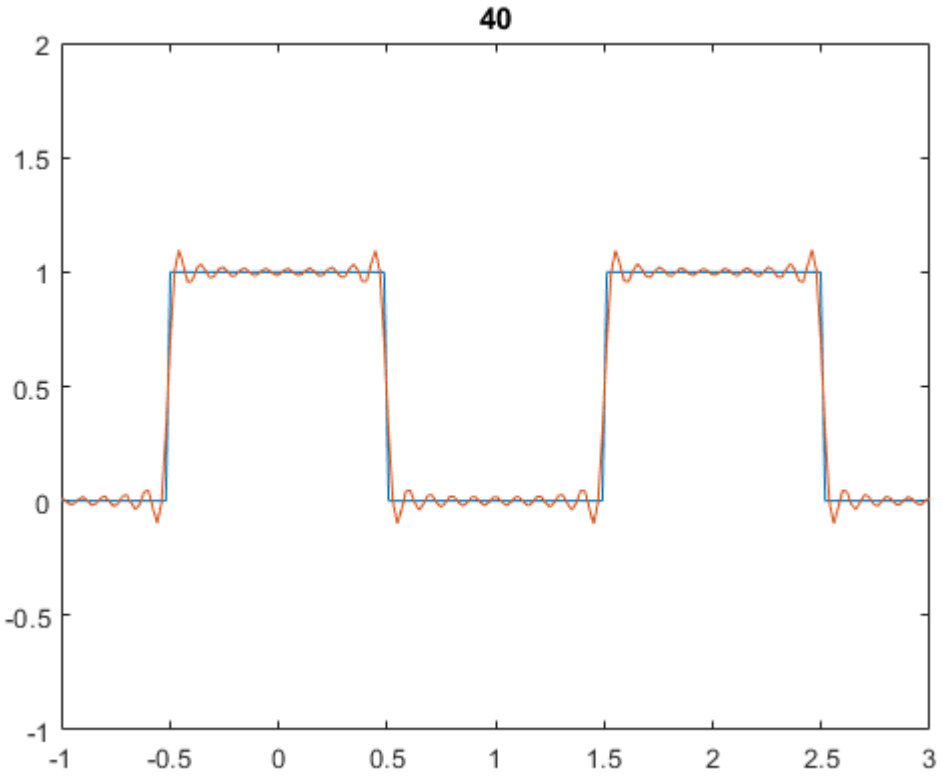

```
% řády
n = [2, 4, 6, 8, 100];

figure,
% pro všechny řády spočítáme rozvoj pomocí Fseries
for i = 1 : size(n,2)
    [a,b,yfit] = Fseries(x,y,n(i));
    subplot(size(n,2),1,i)
    plot(x,y,x,yfit);
end
```

```
Warning: Rank deficient, rank = 199, tol =  4.474240e-13.
```



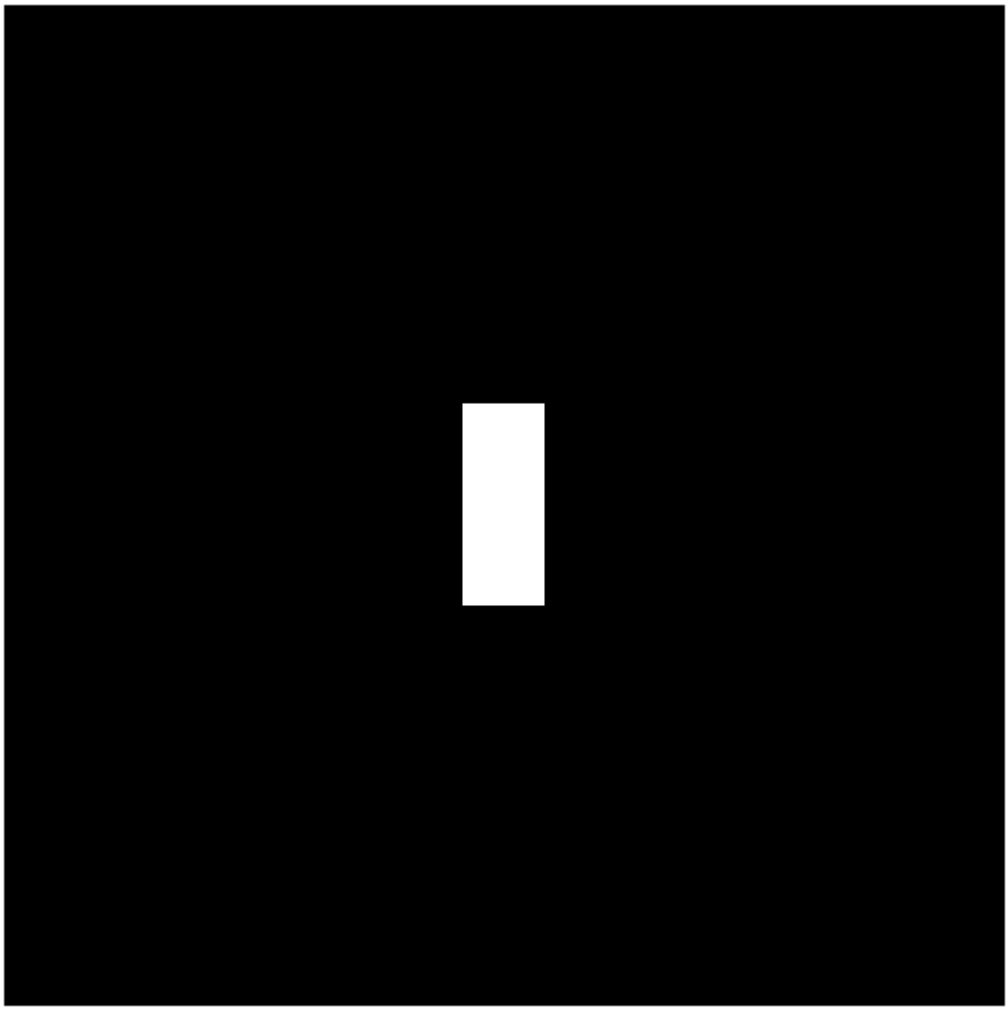
Příklad 2 animace

```
% pro všechny řády spočítáme rozvoj pomocí Fseries
figure,
for i = 2 : 2 : 40
    [a,b,yfit] = Fseries(x,y,i);
    pause(0.3);
    plot(x,y,x,yfit);
    title(num2str(i));
    ylim([-1,2]);
end
```



FREKVENČNÍ DOMÉNA

```
% Vytvoření jednoduchého obrázku
M = 500;
N = 500;
f = zeros([M,N]);
f(200:300,230:270)=1;
figure, imshow(f);
```



Fourierovo spektrum

Převod do frekvenční domény - funkce `fft2()`

```
F = fft2(f);
```

dc koeficient

```
display(F(1,1));
```

```
4141
```

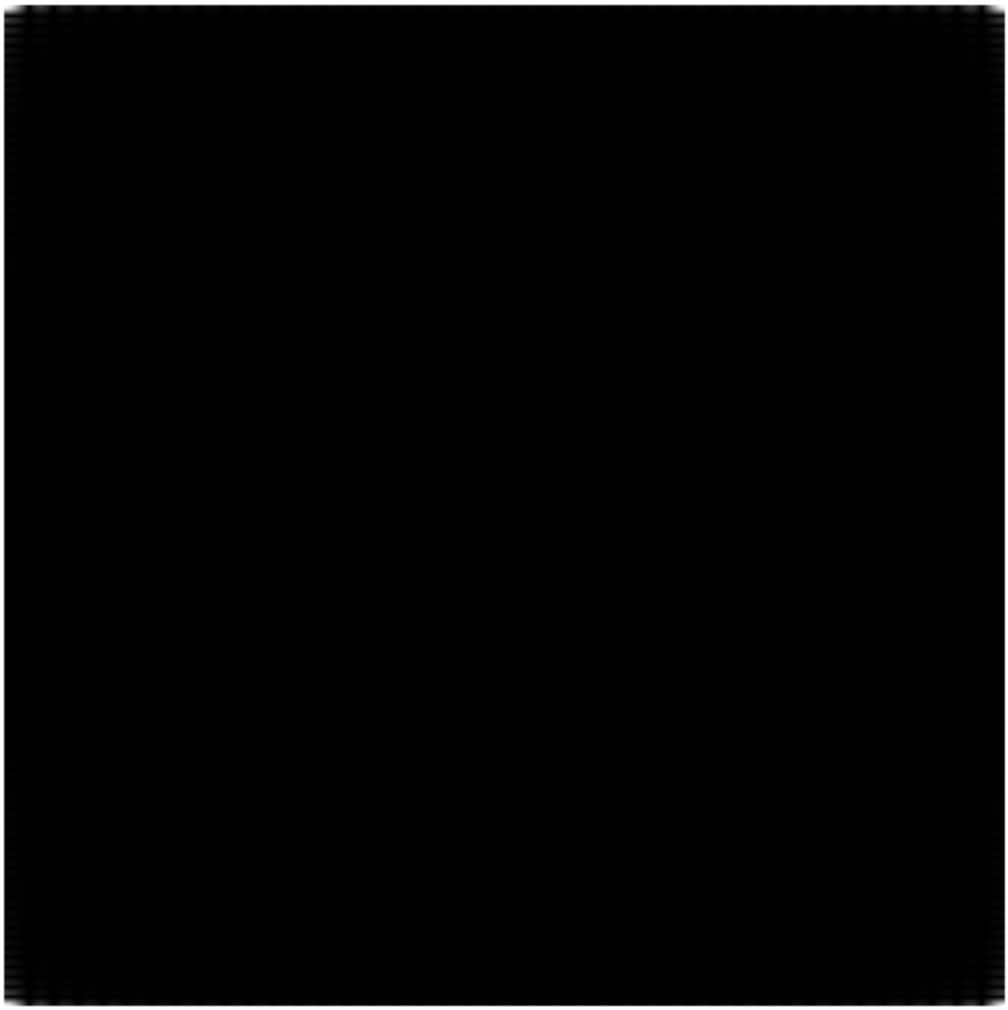
```
display(M*N*mean(f(:)));
```

```
4141
```

Spektrum

Pro výpočet budeme potřebovat obrázek rozdělit na 2 části- reálnou a imaginární. K tomu slouží funkce real a imag.

```
R = real(F);
I = imag(F);

S = sqrt(R.^2+I.^2);
figure, imshow(S, []);
```



Spektrum

v matlabu je možné k výpočtu použít funkci `abs()`

```
S2 = abs(F);
figure, imshow(S-S2,[]);
```

```
display(max(max(abs(S-S2))));
```

```
4.5475e-13
```

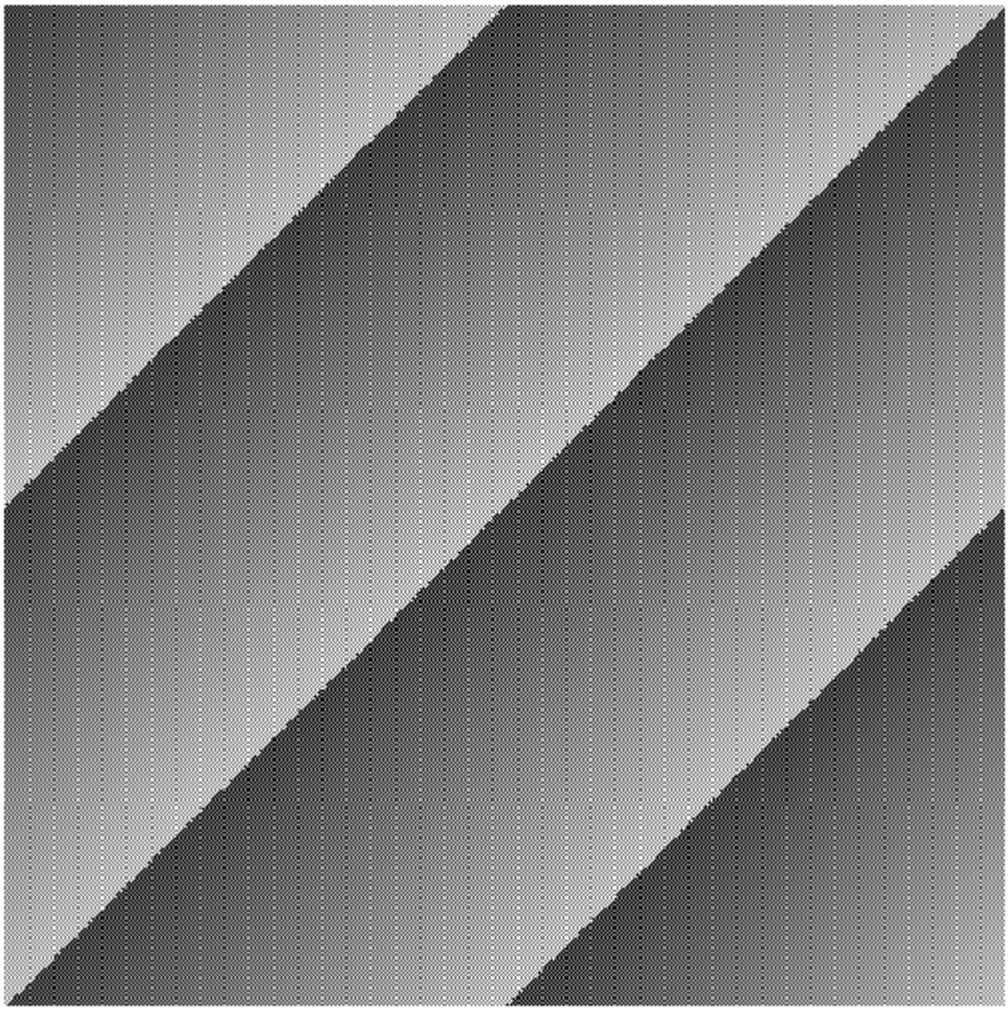
Fáze

Z reálné a imaginární složky spočítáme arcustanges pomocí `atan2()` (phi je pole velikosti obrázku obsahující úhly v radiánech v intervalu -pi pi)

```
phi = atan2(I,R);
```

zobrazení, jak vypadá fáze pro daný obrázek

```
figure, imshow(phi,[]);
```



Fáze

Případně můžeme úhly počítat přímo bez nutnosti převodu obrázku na imaginární a reálnou část pomocí funkce `angle()`

```
phi1 = angle(F);
% výsledek je stejný.
```

ze spektra a fáze umíme spočítat fourierův obraz následujícím způsobem (polární souřadnice)

```
F2 = S.*exp(i*phi);
```

ÚKOL 1

Porovnejte fázi a spektrum dvou obrázků, které obsahují stejný bílý obdélník, jen posunutý.

fftshift

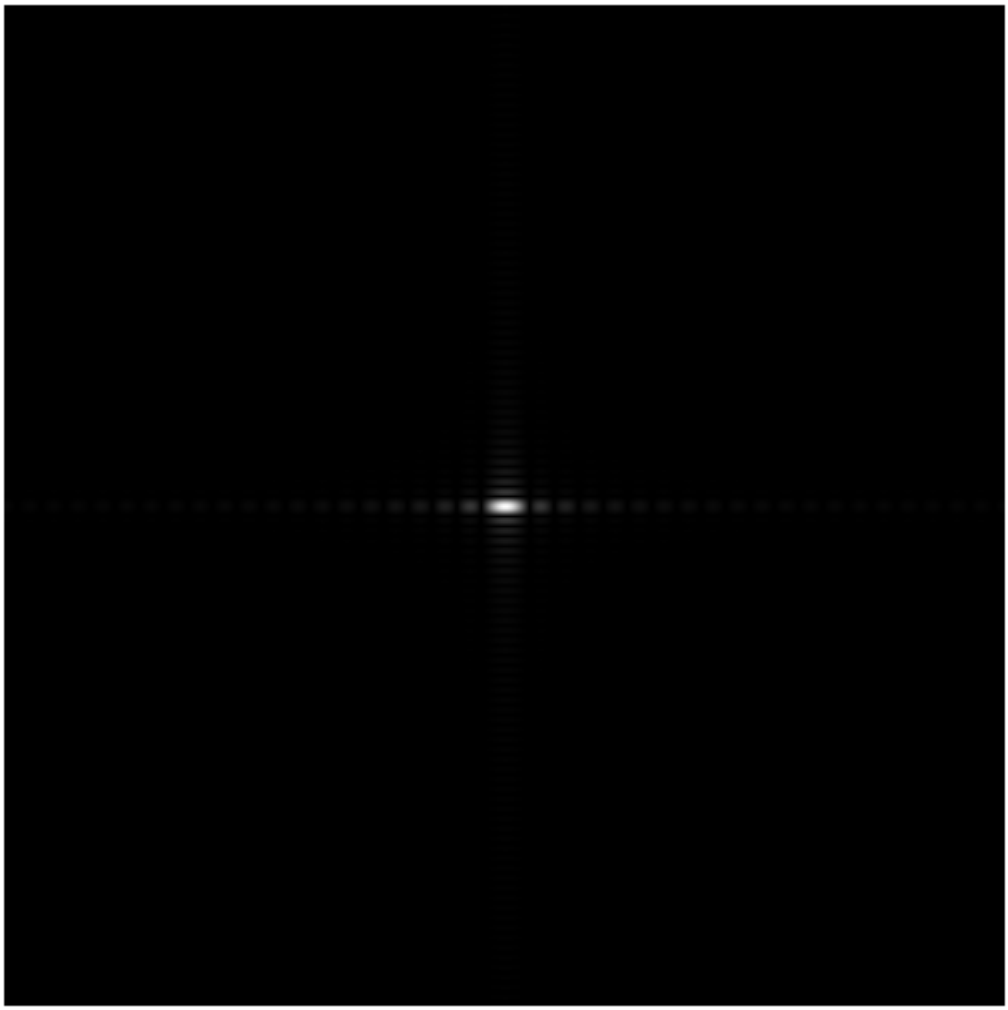
Jak víte, výsledek je periodický s periodou velikosti obrázku. Největší hodnoty jsou v rozích. Pro posun nejvyšších frekvencí doprostřed se používá funkce fftshift. Zde je ukázka, jak tato funce pracuje. Provede se posun o půl periody doprava a dolů (tedy jako by se přehodil 1. a 4. kvadrant a 2. a 3.

```
a = [1 2;
     3 4];
fftshift(a)
```

```
ans = 2×2
     4     3
     2     1
```

toto dostaneme, pokud aplikujeme `fftshift()` na náš obrázek

```
Fc = fftshift(F);
Sc = abs(Fc);
imshow(Sc, [ ]);
```



ÚKOL 2

vynásobte původní obrázek `(-1)^(x+y)` a aplikujte `fft2()` a porovnejte s `fftshift(fft2(f))`.

Rozsah spektra

```
display(min(min(Sc)));
```

```
1.8117e-04
```

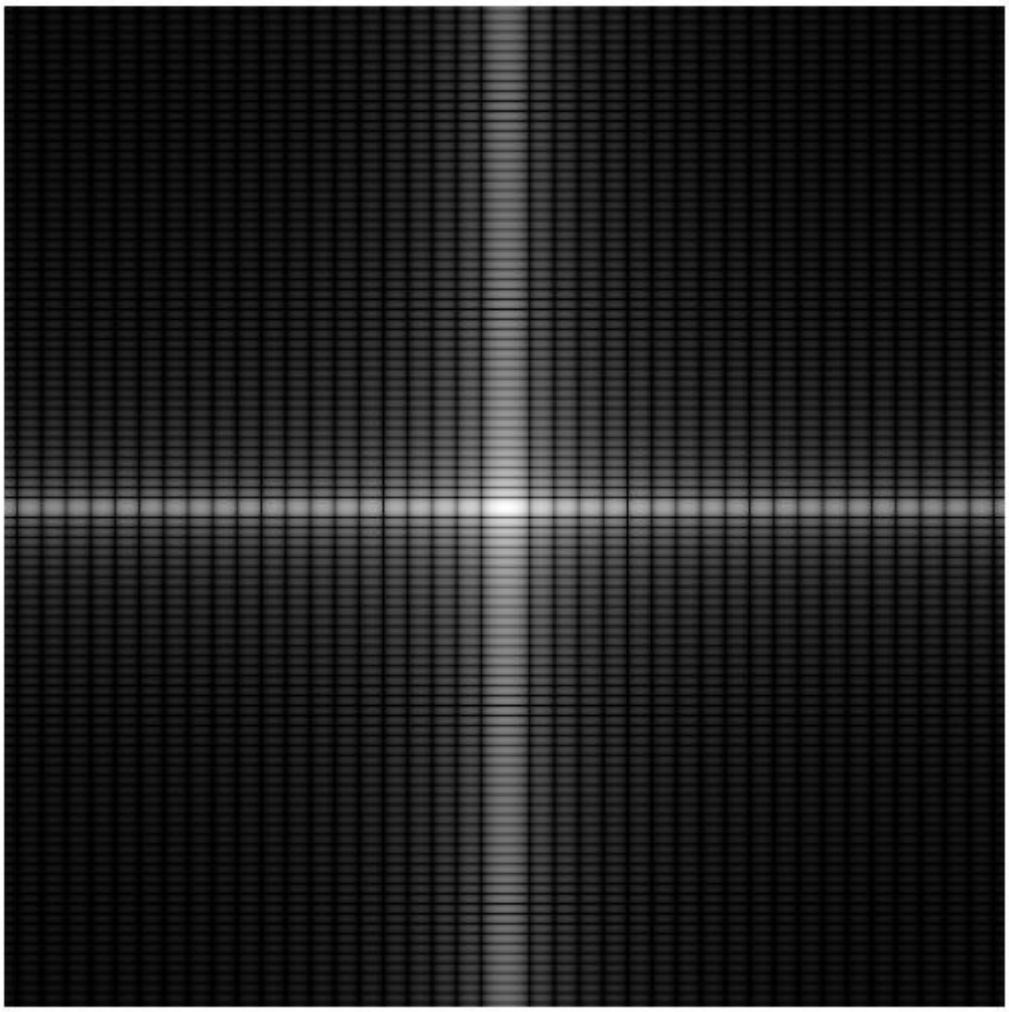
```
display(max(max(Sc)));
```

```
4141
```

Logaritmická transformace

vidíme, že rozsah hodnot v obrázku se frekvenční doméně je hodně velký. Navíc jak víte z toho, jak frekvence vypadají, tak nízkých hodnot je velká většina a objevuje se jen málo vysokých frekvencí. Proto se mnohdy pro zobrazení používá logaritmická transformace. Díky ní vyniknou i detaily v nízkých frekvencích.

```
S2 = log(1 + Sc);
figure, imshow(S2, [ ]);
```



ifftshift

pro převedení obrázku zpět do podoby, kdy jsou nejvyšší hodnoty v rozích se používá funkce `ifftshift()`. Nelze použít `fftshift()`, protože nepracují úplně stejně pro matice s lichou velikostí. Viz následující příklad

```
M = [1 2 3;
     4 5 6;
     7 8 9];
M2 = fftshift(M);
```

```
fftshift(M2)
```

```
ans = 3×3
     5     6     4
     8     9     7
     2     3     1
```

```
ifftshift(M2)
```

```
ans = 3×3
     1     2     3
     4     5     6
     7     8     9
```

```
F1 = ifftshift(Fc);
```

Pro převod obrázku zpět do prostorové domény se používá funkce `ifft2()`

```
f1 = ifft2(F);

% Díky zaokrouhlování se ve výsledku objevuje i komplexní složka
% (hooooooodně malá), tak jí můžeme zanedbat a vzít jen reálnou část.
f2 = real(f1);
imshow(f2);
```