Počítačová podpora v elektrotechnice

Ing. Petr Kropík, Ph.D.
pkropik@kte.zcu.cz

Ing. Lenka Šroubová, Ph.D.
lsroubov@kte.zcu.cz

Oddělení informatiky
Katedra teoretické elektrotechniky
Fakulta elektrotechnická
Západočeská univerzita v Plzni

Místnost: EK602

FACULTY OF ELECTRICAL ENGINEERING
UNIVERSITY OF WEST BOHEMIA

Trojdimenzionální grafy

Vykreslení vrstevnic

`contour(X,Y,Z)` – plošné (pohled shora) dvourozměrné vykreslení např. vrstevnic, ekvipotenciál atp.

`contour(X,Y,Z,n)` – chci vykresli *n*-úrovní (v podstatě určím jak hustě se mají kreslit vrstevnice), není to povinný parametr, např. `contour(X,Y,Z,30)`

`contourf(X,Y,Z)` – vrstevnice s barevnými plochami

`contour3(X,Y,Z)` – trojrozměrné vykreslení např. vrstevnic, ekvipotenciál atp.

`pcolor(X,Y,Z)` – obdoba `contour`, ale vykresleno jako barevné plochy

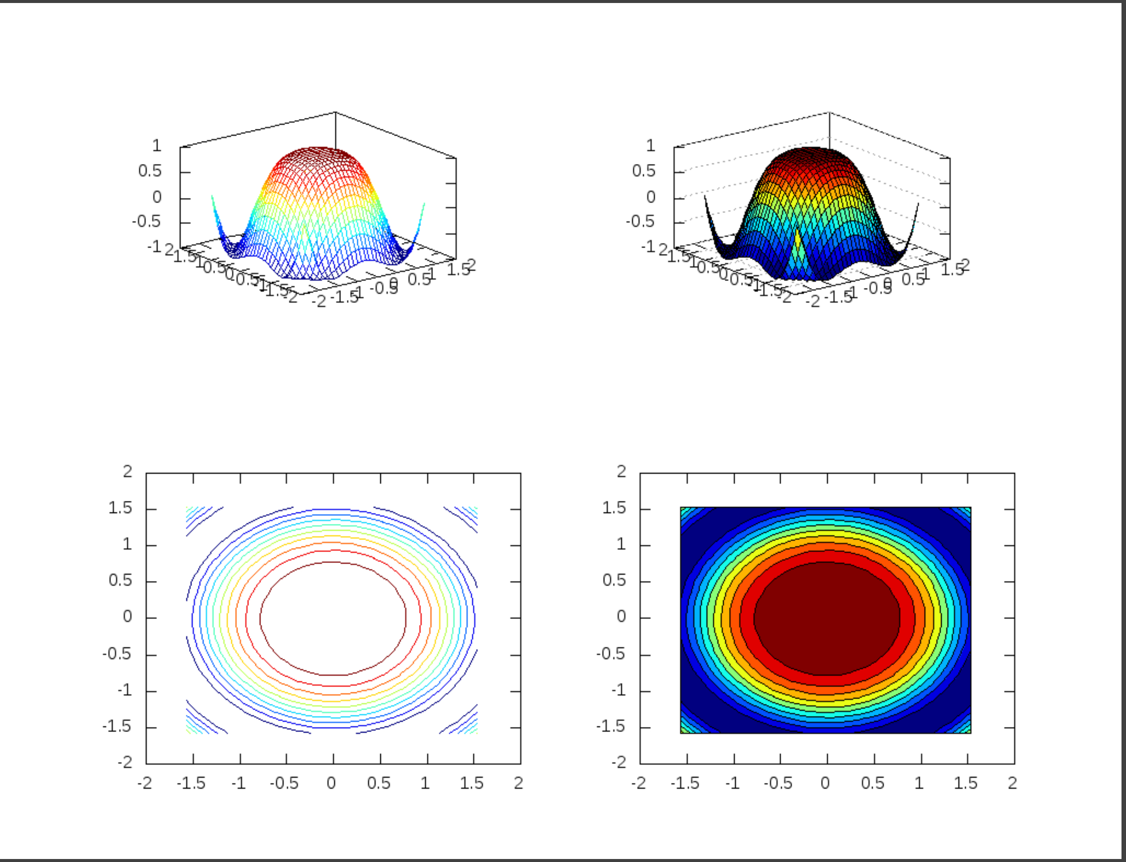
Trojdimenzionální grafy

`mesh()`, `surf()` – lze použít pro zobrazení stejné funkce

Příklad:

Graf funkce $z = \cos(x^2 + y^2)$ pro x, y od $-\pi/2$ do $\pi/2$ a jeho vrstevnice.

```
[X,Y] = meshgrid(-pi/2:0.1:pi/2);
Z = cos(X.^2+Y.^2);
subplot(2,2,1)
mesh(X,Y,Z)
subplot(2,2,2)
surf(X,Y,Z)
subplot(2,2,3)
contour(X,Y,Z)
subplot(2,2,4)
contourf(X,Y,Z)
```

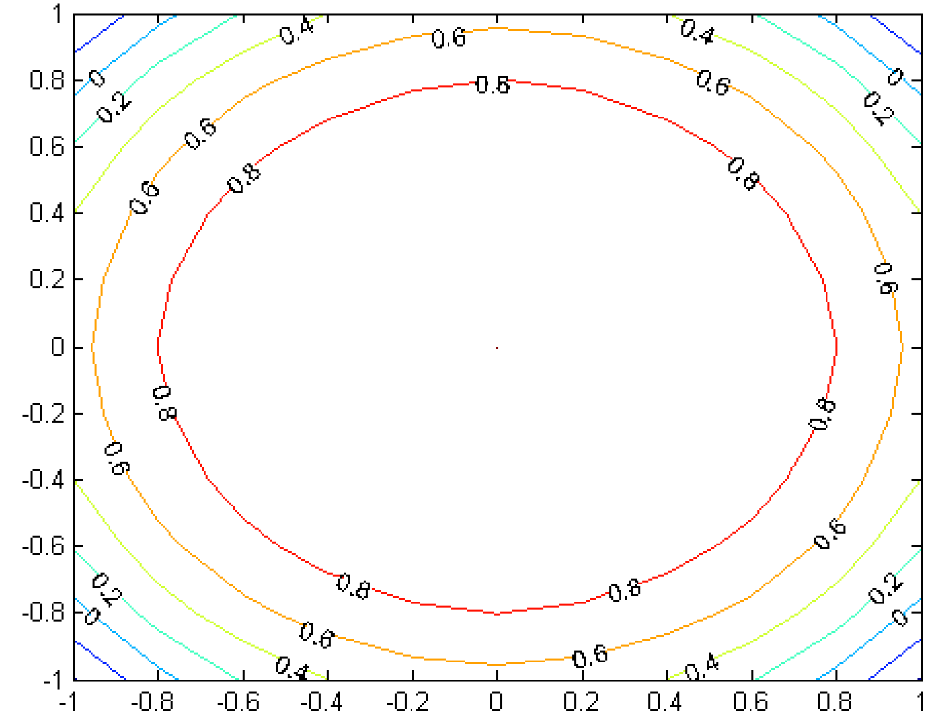


Trojdimenzionální grafy

Příklad:

Graf funkce $z = \cos(x^2 + y^2)$ pro x, y od **-1** do **1** s krokem 0.2 a vykreslení vrstevnic ke grafu

```
[X,Y] = meshgrid(-1:0.2:1);
Z = cos(X.^2+Y.^2);
[C,h]=contour(X,Y,Z);
clabel(C,h);
```



contour()

- vrstevnice grafu

clabel()

- popisky vrstevnic

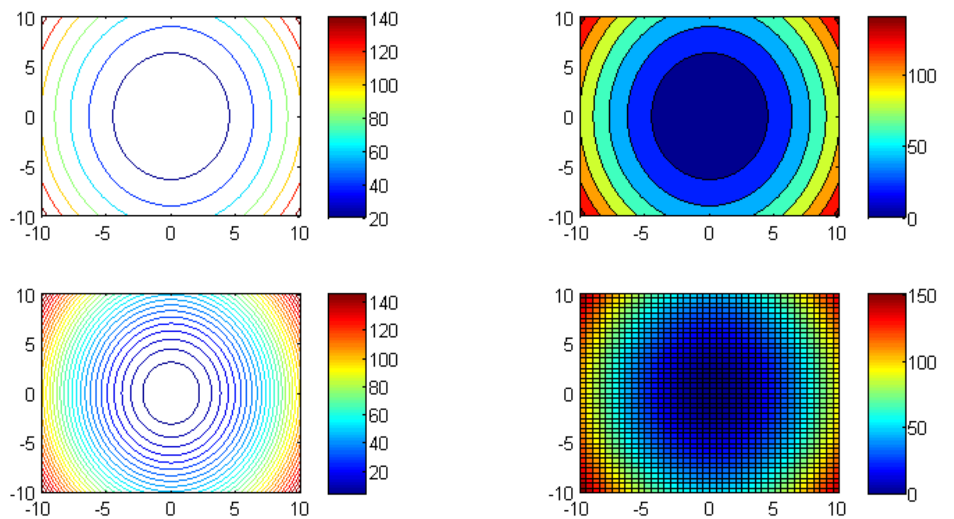
Trojdimenzionální grafy

Příklad:

Vrstevnice grafu funkce $z = \cos(x^2 + 0.5y^2)$ pro x, y od -10 do 10

```
x = [-10:0.5:10];
y = x;
[X,Y] = meshgrid(x,y);
Z = X.^2+0.5*Y.^2;
subplot(2,2,1)
contour(x,y,Z)
colorbar
subplot(2,2,2)
contourf(x,y,Z)
colorbar
subplot(2,2,3)
contour(x,y,Z,30)
colorbar
```

```
subplot(2,2,4)
pcolor(x,y,Z)
colorbar
```



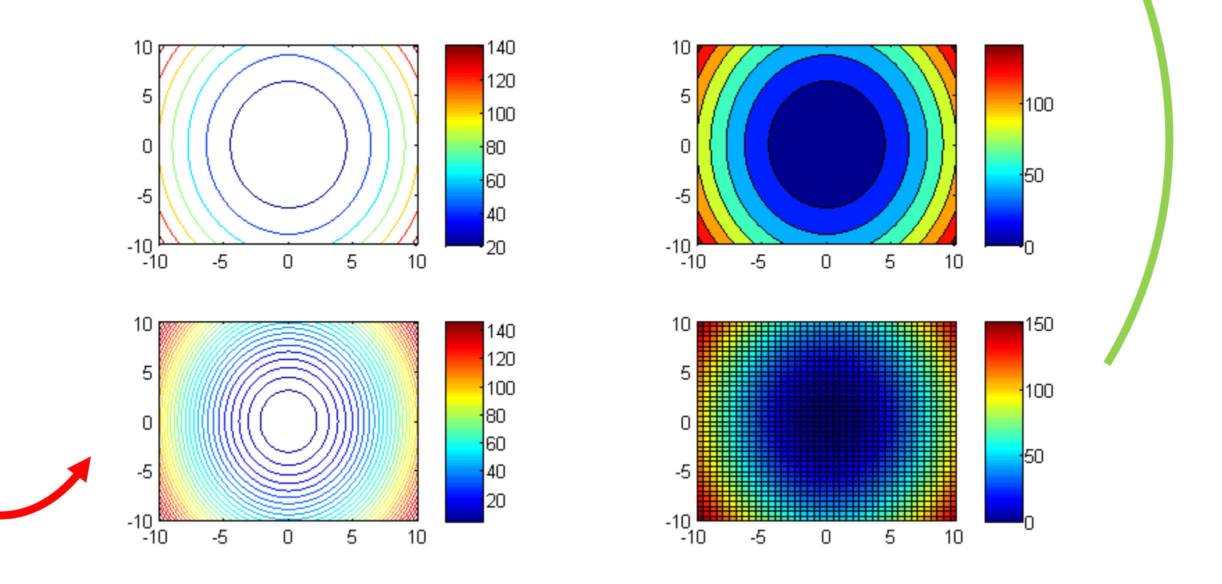
Trojdimenzionální grafy

Příklad:

Vrstevnice grafu funkce $z = \cos(x^2 + 0.5y^2)$ pro x, y od -10 do 10

```
x = [-10:0.5:10];
y = x;
[X,Y] = meshgrid(x,y);
Z = X.^2+0.5*Y.^2;
subplot(2,2,1)
contour(x,y,Z)
colorbar
subplot(2,2,2)
contourf(x,y,Z)
colorbar
subplot(2,2,3)
contour(x,y,Z,30)
colorbar
```

```
subplot(2,2,4)
pcolor(x,y,Z)
colorbar
  % barevná stupnice
```



Trojdimenzionální grafy

Zobrazení **gradientů funkce**:

`[FX,FY]=gradient(F)` – výpočet gradientu matice **F**, výsledkem jsou dvě matice podobně jako u `meshgrid()`.

`quiver(parametry)` – neumí počítat gradient – umí vykreslit graf ze šipek, které mohou představovat gradient v daném bodě.

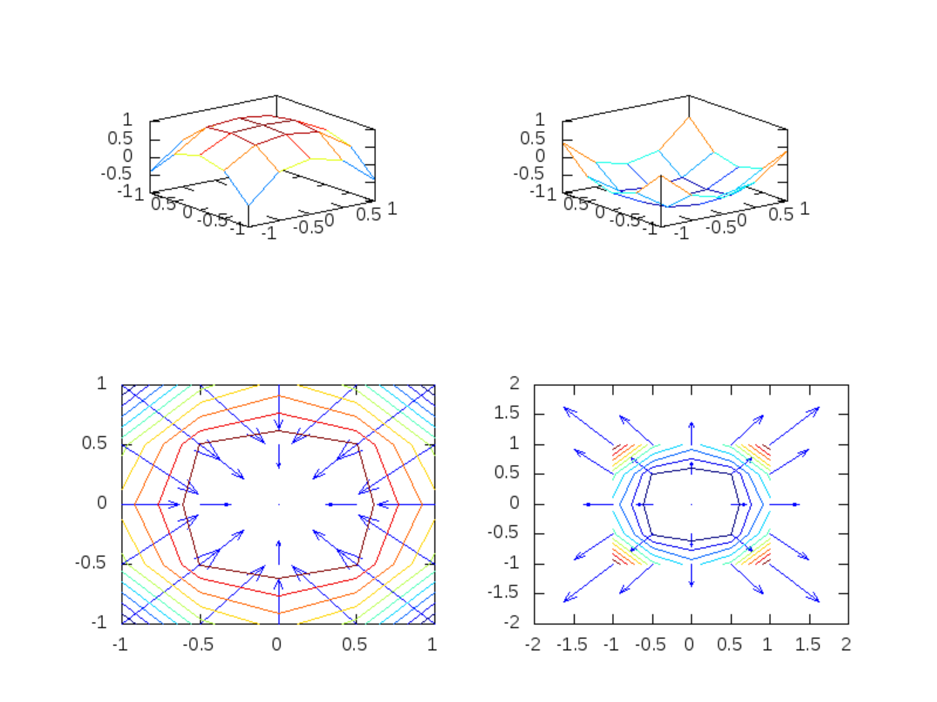
(Gradient je v obecném smyslu slova směr růstu, matematickým výsledkem je vektorové pole vyjadřující směr a velikost změny skalárního pole, gradienty jsou tedy vektory, jejichž složky tvoří jednotlivé parciální derivace funkce vyjadřující dané skalární pole. **FX** odpovídá $\partial \mathbf{F}/\partial x$, **FY** pak $\partial \mathbf{F}/\partial y$.)

Příklad: Graf funkce: $z_1 = \cos(x^2 + y^2)$ a funkce $z_2 = -\cos(x^2 + y^2)$ a vykreslení jejich vrstevnic a gradientu, pro x, y od -1 do 1 s krokem 0,5.

```
[X,Y]=meshgrid(-1:0.5:1);
Z = cos(X.^2 + Y.^2);
subplot(2,2,1)
mesh(X,Y,Z)
subplot(2,2,3)
[GX,GY]=gradient(Z);
quiver(X,Y,GX,GY)
hold on
contour(X,Y,Z)
hold off

Z2 = -cos(X.^2 + Y.^2);
subplot(2,2,2)
mesh(X,Y,Z2)
```

```
subplot(2,2,4)
[GX2,GY2]=gradient(Z2);
quiver(X,Y,GX2,GY2)
hold on
contour(X,Y,Z2)
hold off
```



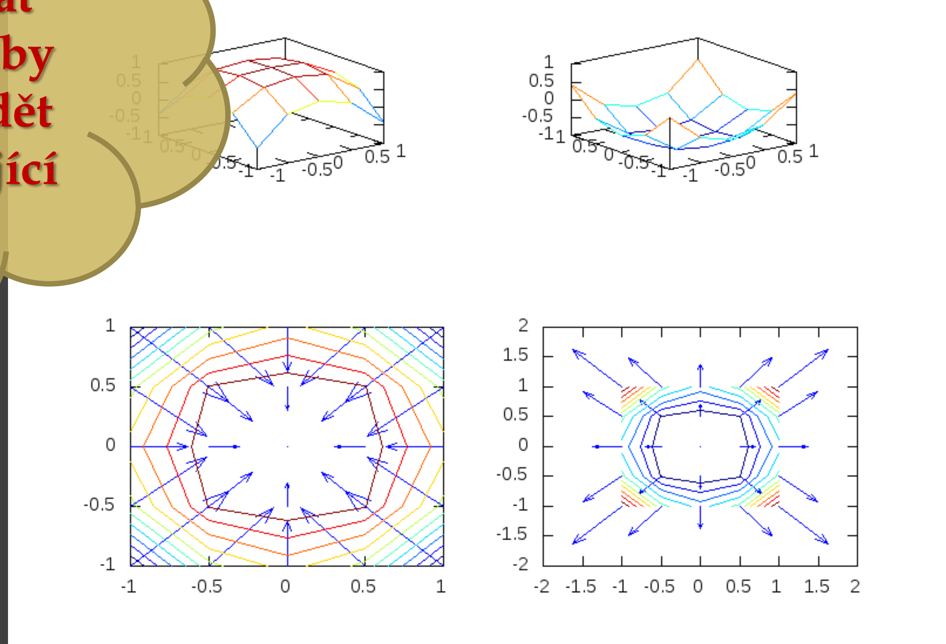
Příklad: Graf funkce: $z_1 = \cos(x^2 + y^2)$ a funkce $z_2 = -\cos(x^2 + y^2)$ a vykreslení jejich vrstevnic a gradientu, pro x, y od -1 do 1 s krokem 0,5.

```
[X,Y]=meshgrid(-1:0.5:1);
Z = cos(X.^2 + Y.^2);
subplot(2,2,1)
mesh(X,Y,Z)
subplot(2,2,3)
[GX,GY]=gradient(Z);
quiver(X,Y,GX,GY)
hold on
contour(X,Y,Z)
hold off

Z2 = -cos(X.^2 + Y.^2);
subplot(2,2,2)
mesh(X,Y,Z2)
```

```
subplot(2,2,4)
[GX2,GY2]=gradient(Z2);
quiver(X,Y,GX2,GY2)
hold on
contour(X,Y,Z2)
hold off
```

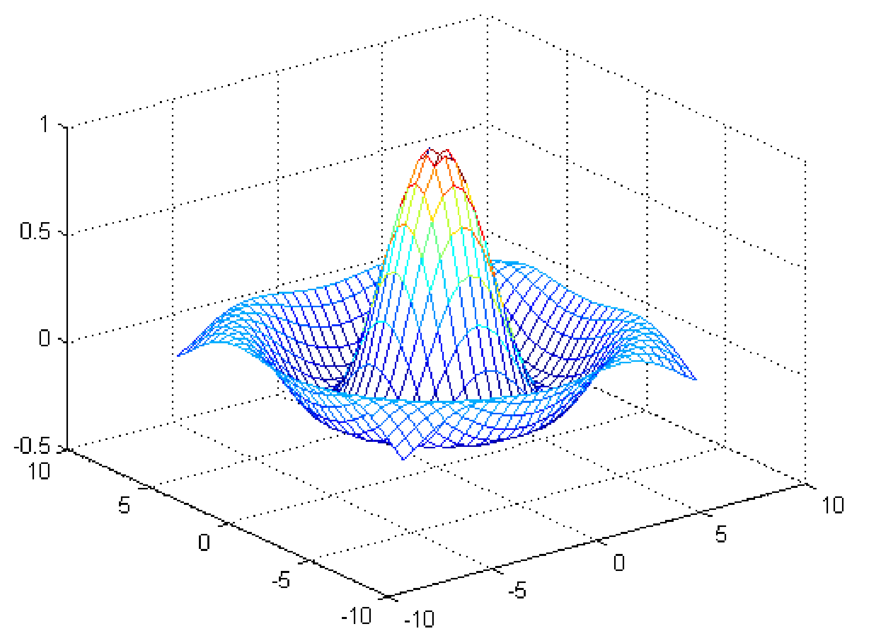
Krok tentokrát zvolen větší, aby byly dobře vidět šipky vyjadřující gradient.

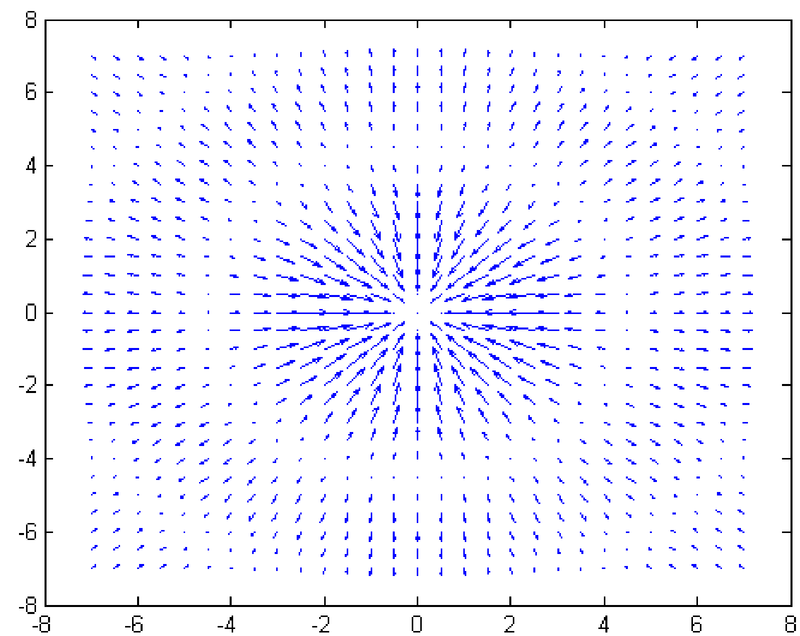


Příklad:

Graf funkce: $$z(x,y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$ a vykreslení jejího gradientu pro x, y od -7 do 7 s krokem 0,5.

```
[X,Y] = meshgrid(-7:0.5:7);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
figure % nové grafické okno
[Px,Py] = gradient(Z,.5,.5);
quiver(X,Y,Px,Py);
```





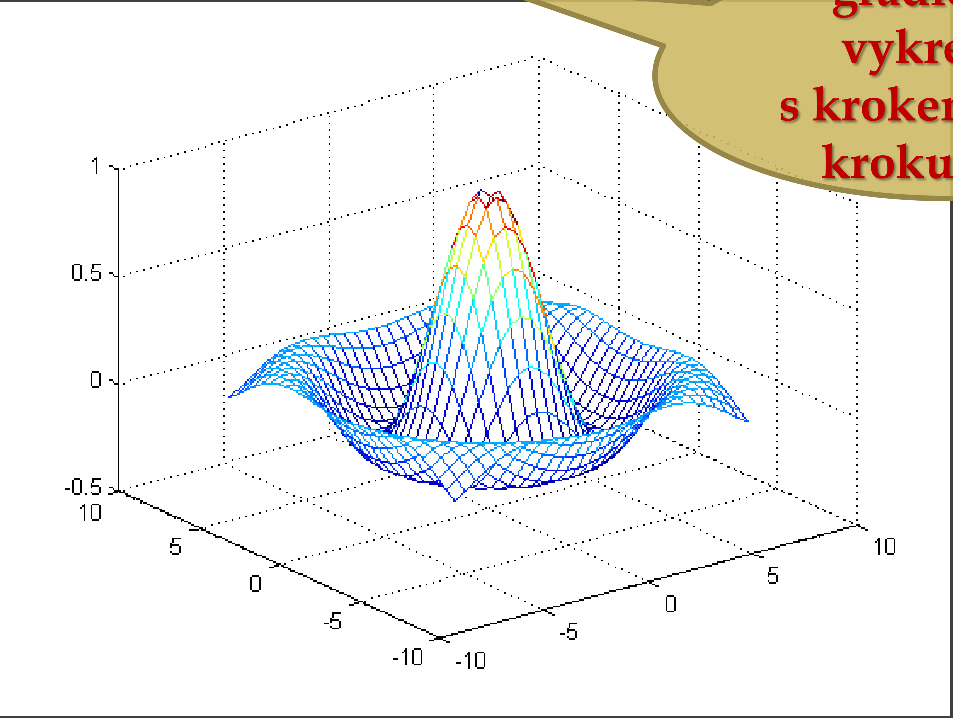
Příklad:

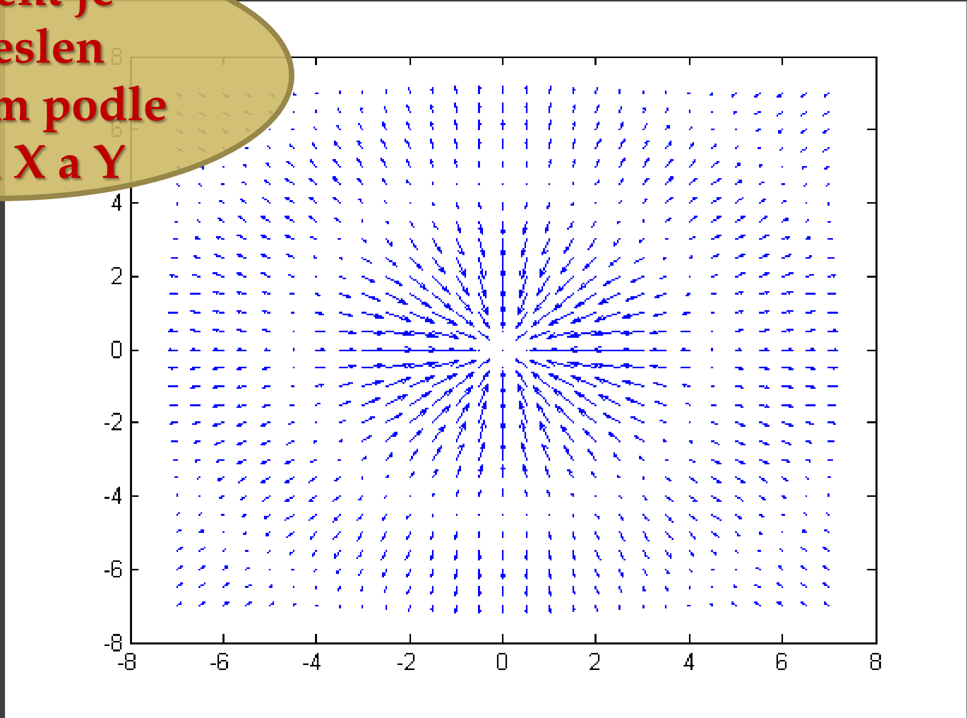
Graf funkce: $$z(x,y)=\frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$ a vykreslení jejího gradientu pro *x*, *y* od **-7** do **7** s krokem **0,5**.

```
[X,Y] = meshgrid(-7:0.5:7);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
figure % nové grafické okno
[Px,Py] = gradient(Z,.5,.5);
quiver(X,Y,Px,Py);
```

krok pro výpočet gradientu, ne pro vykresleni

gradient je vykreslen s krokem podle kroku X a Y





Trojdimenzionální grafy

Normály k povrchu (k ploše) grafu

`[U,V,W] = surfnorm(X,Y,Z)` – vytvoření pomocných matic **U**, **V**, **W** pro normály k plošnému grafu funkce **Z**

`quiver3(X,Y,Z,U,V,W)` – vykreslí graf ze šipek se složkami **U**, **V**, **W** v bodech se souřadnicemi **X**, **Y**, **Z**.
Matice **X**, **Y**, **Z**, **U**, **V**, **W** musí mít všechny stejnou velikost.

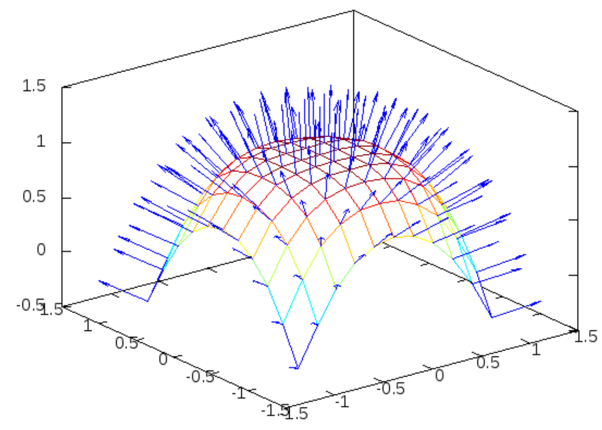
Trojdimenzionální grafy

Příklad:
Graf funkce $z = \cos(x^2 + y^2)$ pro x, y od -1 do 1 a vykreslení normál k povrchu funkce

```
[X,Y] = meshgrid(-1:0.2:1);
Z = cos(X.^2+Y.^2);
mesh(X,Y,Z)
[NX,NY,NZ] = surfnorm(X,Y,Z);
hold on
quiver3(X,Y,Z,NX,NY,NZ)
hold off
```

vytvoření pomocných matic NX,NY,NZ pro normály k plošnému grafu funkce Z

vykresleni normál k ploše grafu jako vektoru (šipek)



Trojdimenzionální grafy

- grafy lze kombinovat – využijeme příkaz **hold on / off**

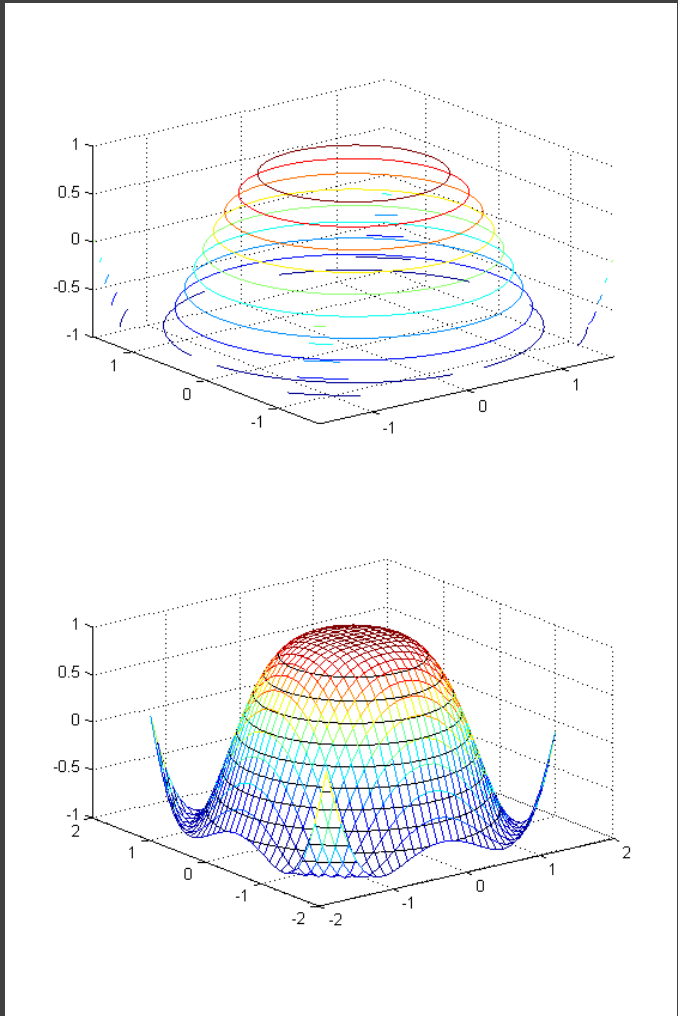
Příklad:

Graf funkce $z = \cos(x^2 + y^2)$ pro x, y od $-\pi/2$ do $\pi/2$ a jeho 3D vrstevnice.

```
[X,Y] = meshgrid(-pi/2:0.1:pi/2);
Z = cos(X.^2+Y.^2);

subplot(2,1,1)
contour3(X,Y,Z)

subplot(2,1,2)
mesh(X,Y,Z)
hold on
contour3(X,Y,Z,'k') % vrstevnice
hold off                    černě
```

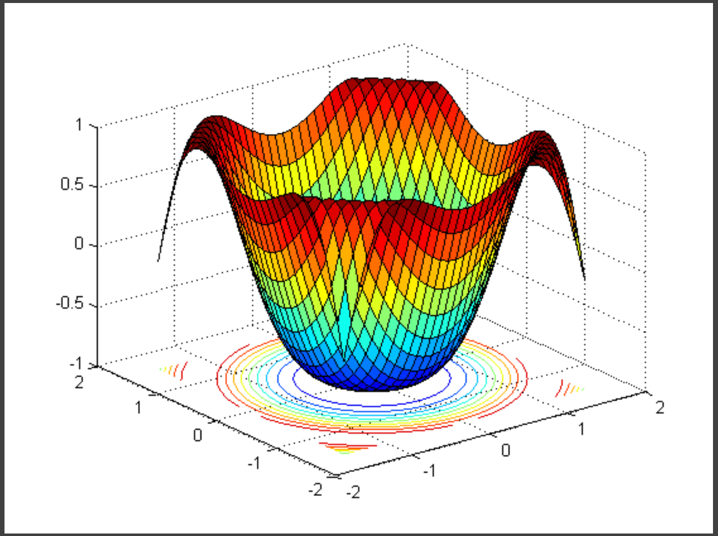


Trojdimenzionální grafy

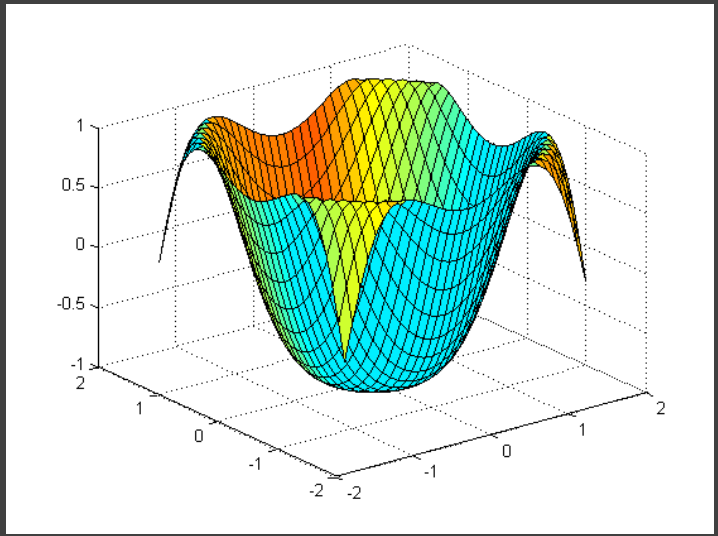
- je více variant příkazů pro zobrazení plošných grafů (odlišné v různých výpočetních systémech), v MATLABu některé kombinují vlastnosti dvou dohromady, např.:
meshc, **surfc** (**mesh** a **surf** a k tomu **contour**), **meshz** (mesh s „podstavcem“), **surfl** (**surf** s „osvětlením“)

Příklad: Graf funkce $z = -\cos(x^2 + y^2)$ pro x, y od $-\pi/2$ do $\pi/2$

```
[X,Y] = meshgrid(-pi/2:0.1:pi/2);
Z = -cos(X.^2+Y.^2);
surfc(X,Y,Z)
```




```
surfl(X,Y,Z)
```



Trojdimenzionální grafy

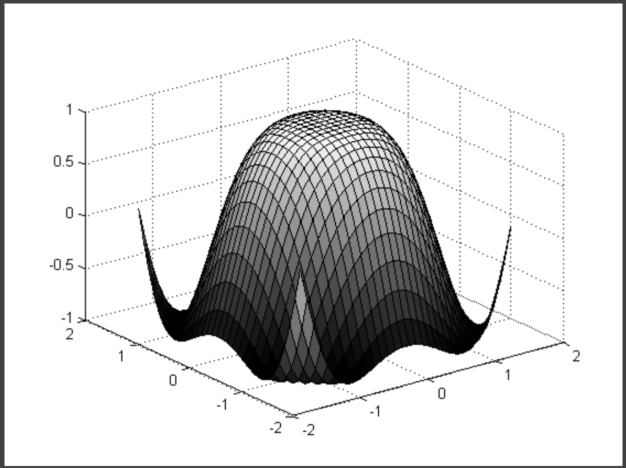
`colormap(paleta)` – nastavuje barevnou paletu pro graf
- parametr `paleta` je matice o 3 sloupcích představujících Red Green Blue, položky mají hodnoty od 0 do 1 (tj. 0 až 100%)
- předpřipravené palety `gray`, `hot`, `copper`, `hsv`, `cool`…

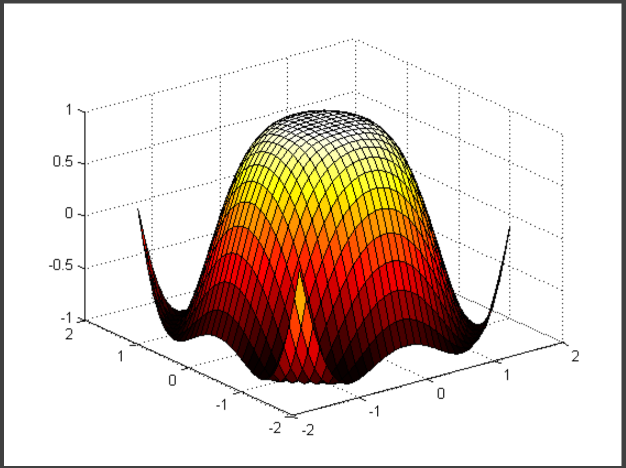
Příklad:

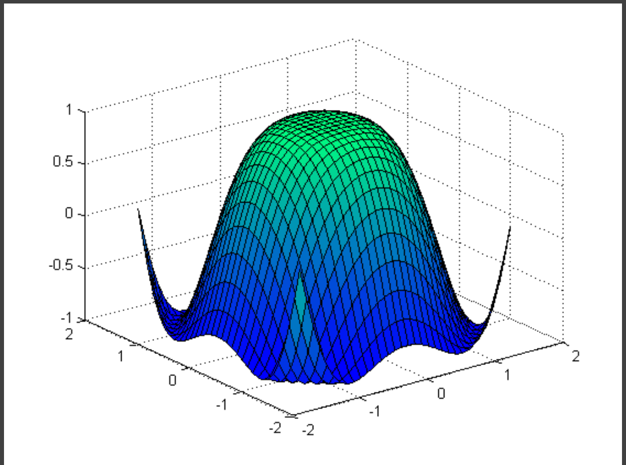
Graf funkce $z = \cos(x^2 + y^2)$ pro x, y od $-\pi/2$ do $\pi/2$

```
[X,Y] = meshgrid(-pi/2:0.1:pi/2);
Z = cos(X.^2+Y.^2);
surf(X,Y,Z)
```

`colormap(gray)` `colormap(hot)` `colormap(winter)`







Dvojdimenzionální grafy

Dodatek:

Základním je pro vykreslení grafu v kartézských souřadnicích je příkaz **`plot:`**

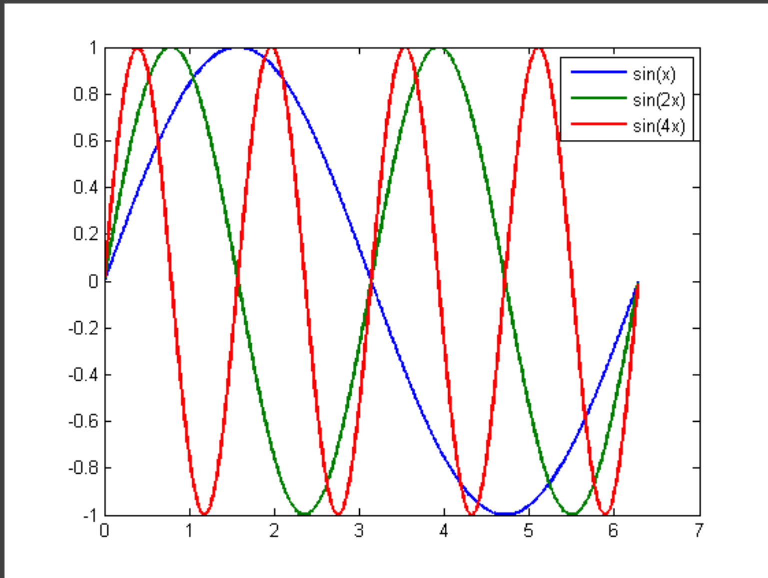
`plot(x,y,S)` – rovinný graf s **lineárním** dělením na osách *x*, *y*

- data jsou předávána ve formě sloupcových nebo řádkových vektorů
- obecně má příkaz tvar **`plot(x,y,S,…)`**,(**`x`** a **`S`** mohou být vynechány)
- skupinu parametrů **`x,y,S`** lze i několikrát opakovat
- parametr **`y`** nemusí být jen jeden vektor, více sloupcových vektorů složených do **matice** provede vykreslení několika průběhů

Dvojdimenzionální grafy

Příklad:
sestavení matice Y, která jako 3 sloupce obsahuje funkční hodnoty $y_1 = \sin(x)$, $y_2 = \sin(\mathbf{2}x)$, $y_3 = \sin(\mathbf{4}x)$, pro x od $\mathbf{0}$ do $\mathbf{2}\pi$, a vykreslení grafů $y_1(x)$, $y_2(x)$, $y_3(x)$ pomocí dat v matici **Y**.

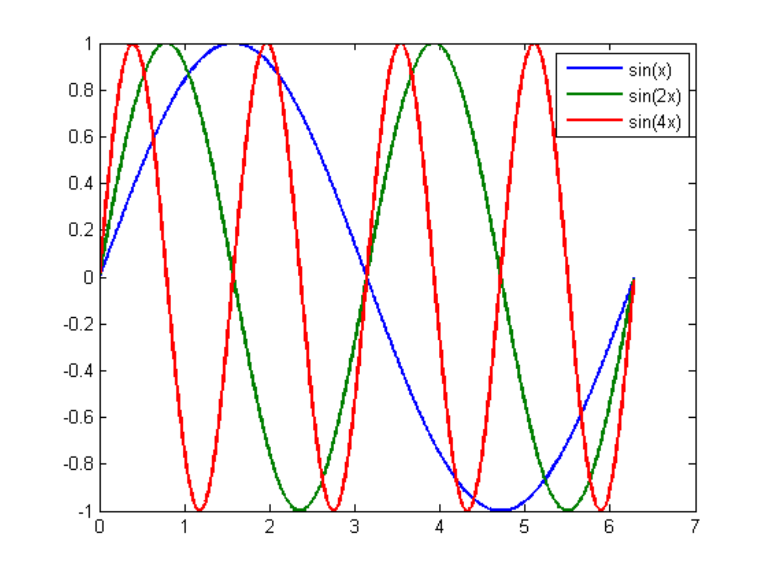
```
x = 0:.01:2*pi;
Y = [sin(x).', sin(2*x).', sin(4*x).'];
% tři sloupce
plot(x.',Y)
legend('sin(x)','sin(2x)','sin(4x)')
```



Dvojdimenzionální grafy

Pokračování příkladu:
sestavení matice Y, která jako 3 sloupce obsahuje funkční hodnoty $y_1 = \sin(x)$, $y_2 = \sin(\mathbf{2}x)$, $y_3 = \sin(\mathbf{4}x)$, pro x od $\mathbf{0}$ do $\mathbf{2}\pi$, a vykreslení grafů $y_1(x)$, $y_2(x)$, $y_3(x)$ pomocí dat v matici **Y**.

```
x = 0:.01:2*pi;
Y = [sin(x).', sin(2*x).', sin(4*x).'];
% tři sloupce
plot(x.',Y)
legend('sin(x)','sin(2x)','sin(4x)')
```



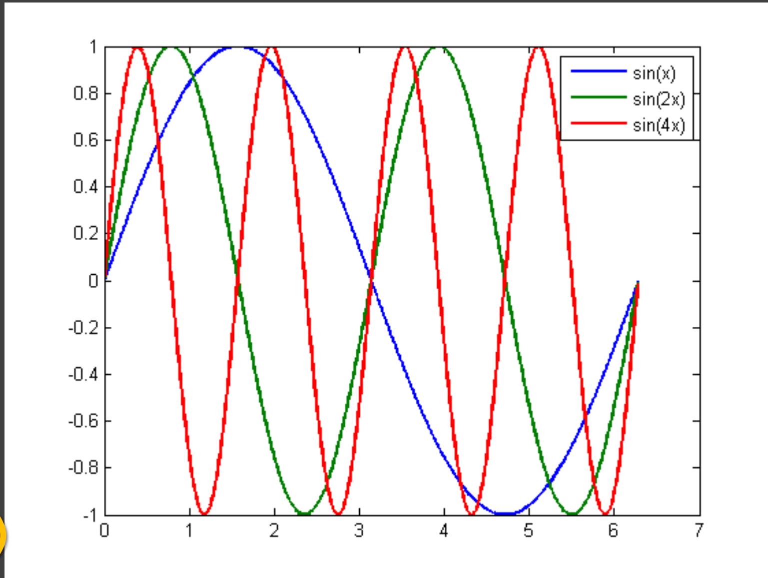
Dvojdimenzionální grafy

Pokračování příkladu:
sestavení matice Y, která jako 3 sloupce obsahuje funkční hodnoty $y_1 = \sin(x)$, $y_2 = \sin(\mathbf{2}x)$, $y_3 = \sin(\mathbf{4}x)$, pro x od $\mathbf{0}$ do $\mathbf{2\pi}$, a vykreslení grafů $y_1(x)$, $y_2(x)$, $y_3(x)$ pomocí dat v matici **Y**.

```
x = 0:.01:2*pi;
Y = [sin(x).', sin(2*x).', sin(4*x).'];
% tři sloupce
plot(x.',Y)
legend('sin(x)','sin(2x)','sin(4x)')
```

Stejný graf obdržíme, bude-li mít matice Y 3 řádky s příslušnými funkčními hodnotami

```
x = 0:.01:2*pi;
Y = [sin(x); sin(2*x); sin(4*x)];
% tři řádky
plot(x,Y)
legend('sin(x)','sin(2x)','sin(4x)')
```



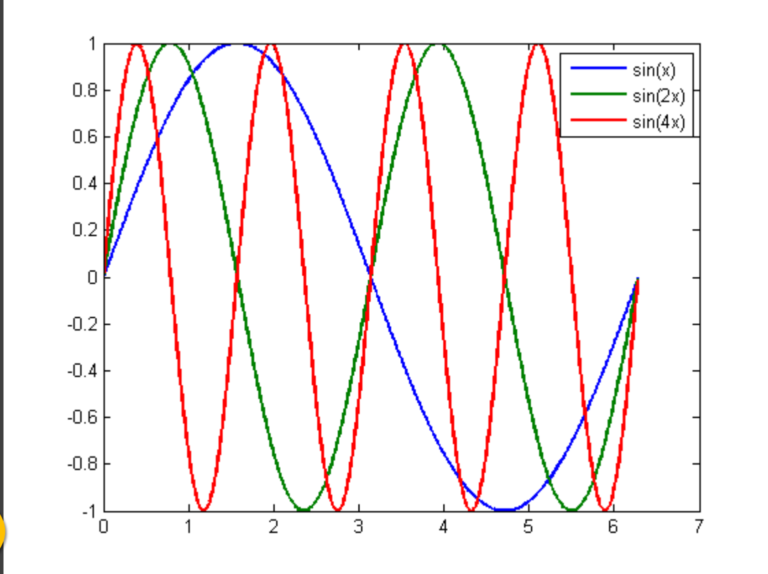
Dvojdimenzionální grafy

Pokračování příkladu:
sestavení matice Y, která jako 3 sloupce obsahuje funkční hodnoty $y_1 = \sin(x)$, $y_2 = \sin(2x)$, $y_3 = \sin(4x)$, pro x od $\mathbf{0}$ do $\mathbf{2\pi}$, a vykreslení grafů $y_1(x)$, $y_2(x)$, $y_3(x)$ pomocí dat v matici **Y**.

```
x = 0:.01:2*pi;
Y = [sin(x).', sin(2*x).', sin(4*x).'];
% tři sloupce
plot(x.',Y)
legend('sin(x)','sin(2x)','sin(4x)')
```

Stejný graf obdržíme, bude-li mít matice Y 3 řádky s příslušnými funkčními hodnotami

```
x = 0:.01:2*pi;
Y = [sin(x); sin(2*x); sin(4*x)];
% tři řádky
plot(x,Y)
legend('sin(x)','sin(2x)','sin(4x)')
```



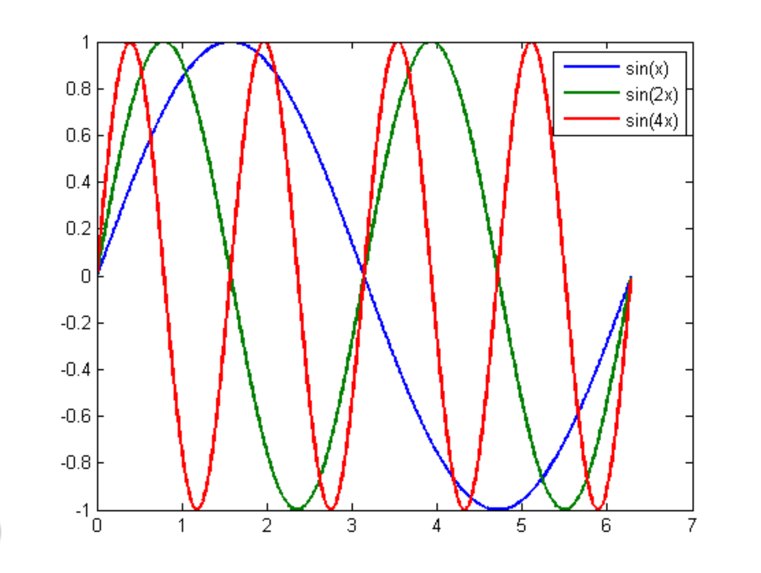
Dvojdimenzionální grafy

Pokračování příkladu:
sestavení matice Y, která jako 3 sloupce obsahuje funkční hodnoty $y_1 = \sin(x)$, $y_2 = \sin(2x)$, $y_3 = \sin(4x)$, pro x od **0** do **2π**, a vykreslení grafů $y_1(x)$, $y_2(x)$, $y_3(x)$ pomocí dat v matici **Y**.

```
x = 0:.01:2*pi;
Y = [sin(x).', sin(2*x).', sin(4*x).'];
% tři sloupce
plot(x.',Y)
legend('sin(x)','sin(2x)','sin(4x)')
```

Stejný graf obdržíme, bude-li mít matice Y 3 řádky s příslušnými funkčními hodnotami

```
x = 0:.01:2*pi;
Y = [sin(x); sin(2*x); sin(4*x)];
% tři řádky
plot(x,Y)
legend('sin(x)','sin(2x)','sin(4x)')
```



Trojdimenzionální grafy

Příklad:

3D graf křivky popsané parametrickými rovnicemi:

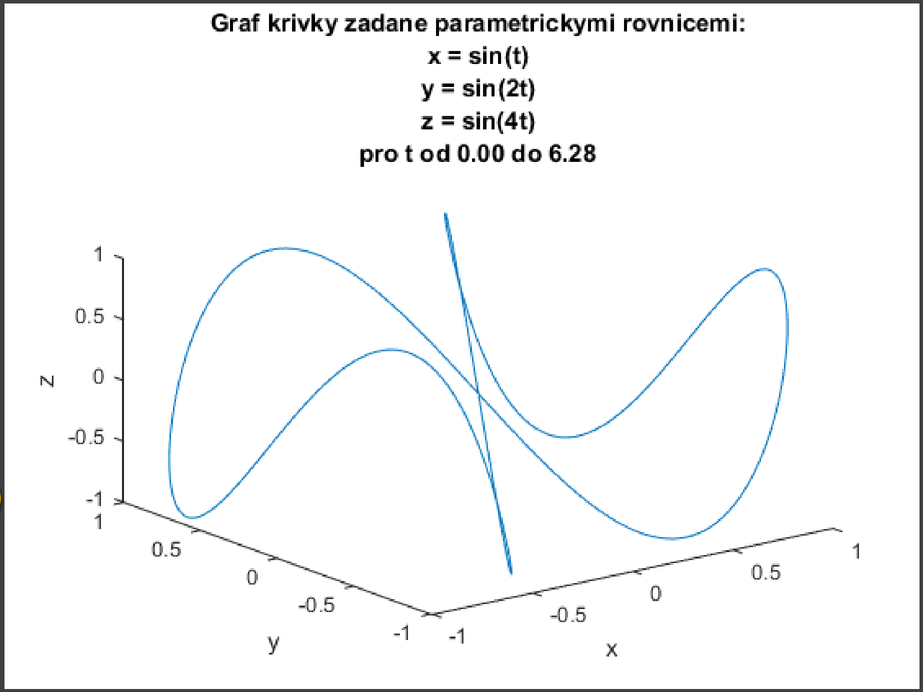
$y_1 = \sin(x)$, $y_2 = \sin(\mathbf{2}x)$, $y_3 = \sin(\mathbf{4}x)$, pro x od **0** do $\mathbf{2\pi}$.

Trojdimenzionální grafy

Pokračování příkladu:

3D graf křivky popsané parametrickými rovnicemi:

$y_1 = \sin(x)$, $y_2 = \sin(2x)$, $y_3 = \sin(4x)$, pro x od **0** do $\mathbf{2\pi}$.




```
x = 0:.01:2*pi;
plot3(sin(x),sin(2*x),sin(4*x))
xlabel('x')
ylabel('y')
zlabel('z')
r = sprintf('Graf krivky zadane parametrickymi
rovnicemi:\nx = sin(t)\ny = sin(2t)\nz = sin(4t)\npro t od
%.2f do %.2f',0,2*pi);
title(r)
```

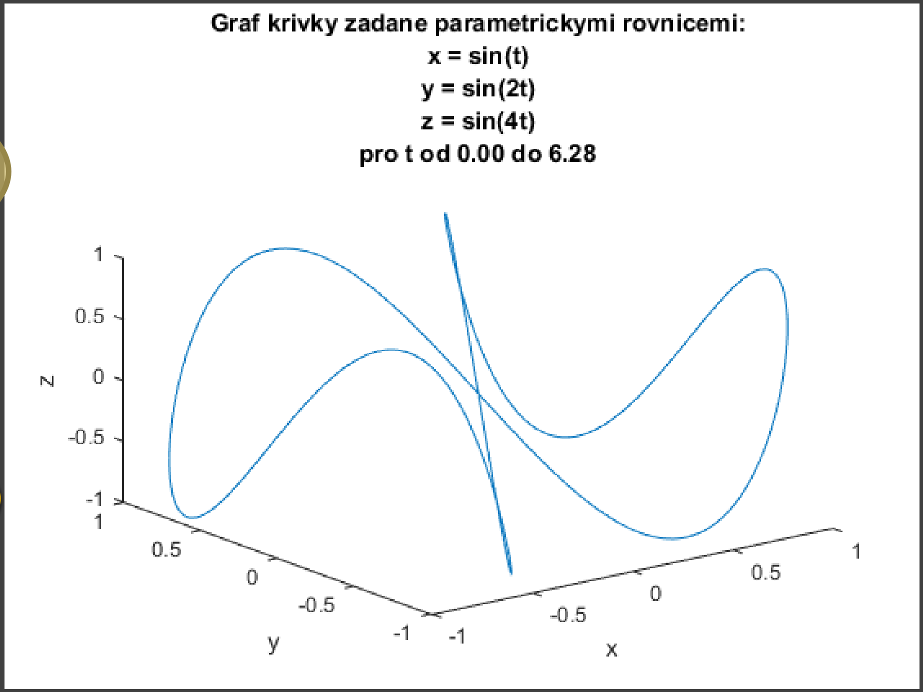
Trojdimenzionální grafy

Pokračování příkladu:

3D graf křivky popsané parametrickými rovnicemi:

$y_1 = \sin(x)$, $y_2 = \sin(2x)$, $y_3 = \sin(4x)$, pro x od $\mathbf{0}$ do $\mathbf{2\pi}$.

V předchozím příkladu řádky matice, zde parametry příkazu `plot3` oddělené „ ,“




```
x = 0:.01:2*pi;
plot3(sin(x),sin(2*x),sin(4*x))
xlabel('x')
ylabel('y')
zlabel('z')
r = sprintf('Graf krivky zadane parametrickymi
rovnicemi:\nx = sin(t)\ny = sin(2t)\nz = sin(4t)\npro t od
%.2f do %.2f',0,2*pi);
title(r)
```

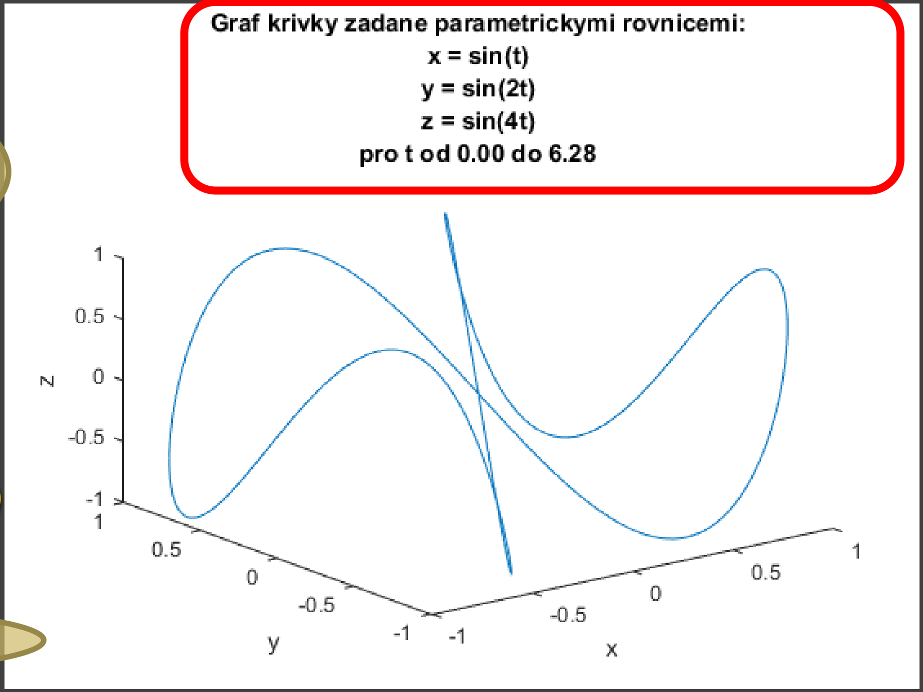
Trojdimenzionální grafy

Pokračování příkladu:

3D graf křivky popsané parametrickými rovnicemi:

$y_1 = \sin(x)$, $y_2 = \sin(\mathbf{2}x)$, $y_3 = \sin(\mathbf{4}x)$, pro x od $\mathbf{0}$ do $\mathbf{2\pi}$.

V předchozím příkladu řádky matice, zde parametry příkazu `plot3` oddělené „ , “




```
x = 0:.01:2*pi;
plot3(sin(x),sin(2*x),sin(4*x))
xlabel('x')
ylabel('y')
zlabel('z')
r = sprintf('Graf krivky zadane parametrickymi
rovnicemi:\nx = sin(t)\ny = sin(2t)\nz = sin(4t)\npro t od
%.2f do %.2f',0,2*pi);
title(r)
```

Zápis do řetězce r

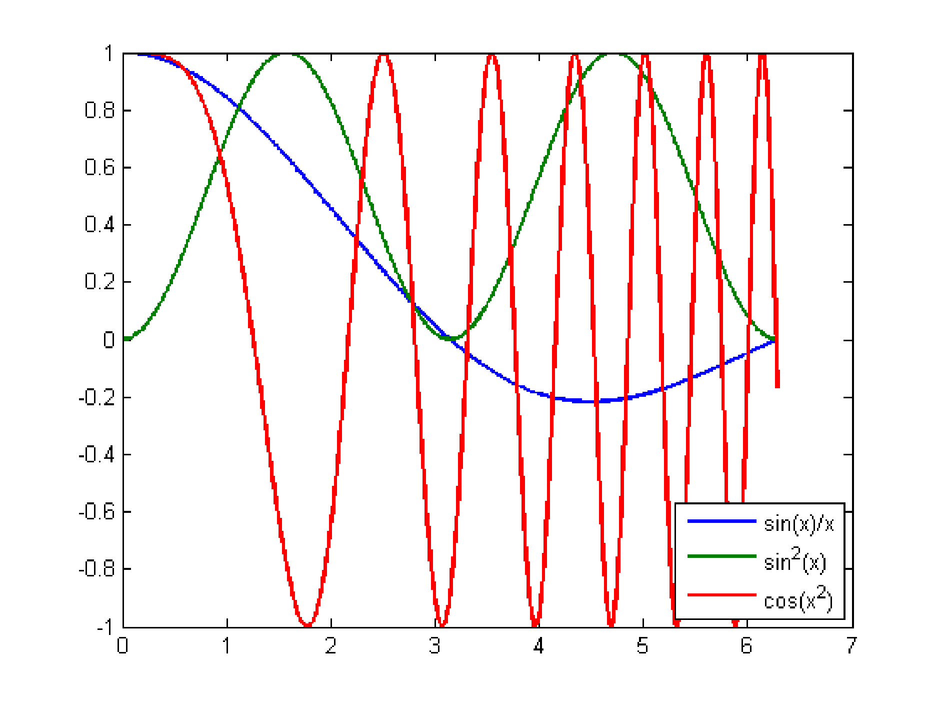
\n – nová řádka

Dvojdimenzionální grafy

Příklad:

Vykreslení grafů funkcí $y_1 = \sin(x)/x$, $y_2 = \sin^2(x)$, $y_3 = \cos(x^2)$.

```
function nekolik_grafu(a)
x = a.';  % sloupcový vektor x (transponovaný a)
y1 = sin(x)./x;                  % jeden sloupcový vektor
y23 = [sin(x).^2,cos(x.^2)]; % dva sloupce
plot(x,y1,x,y23)
legend('sin(x)/x','sin^2(x)','cos(x^2)','Location','SouthEast')
end
```



Volání funkce pro x od $\mathbf{0}$ do $\mathbf{2\pi}$:

```
x = [0:0.01:2*pi];
nekolik_grafu(x)
```

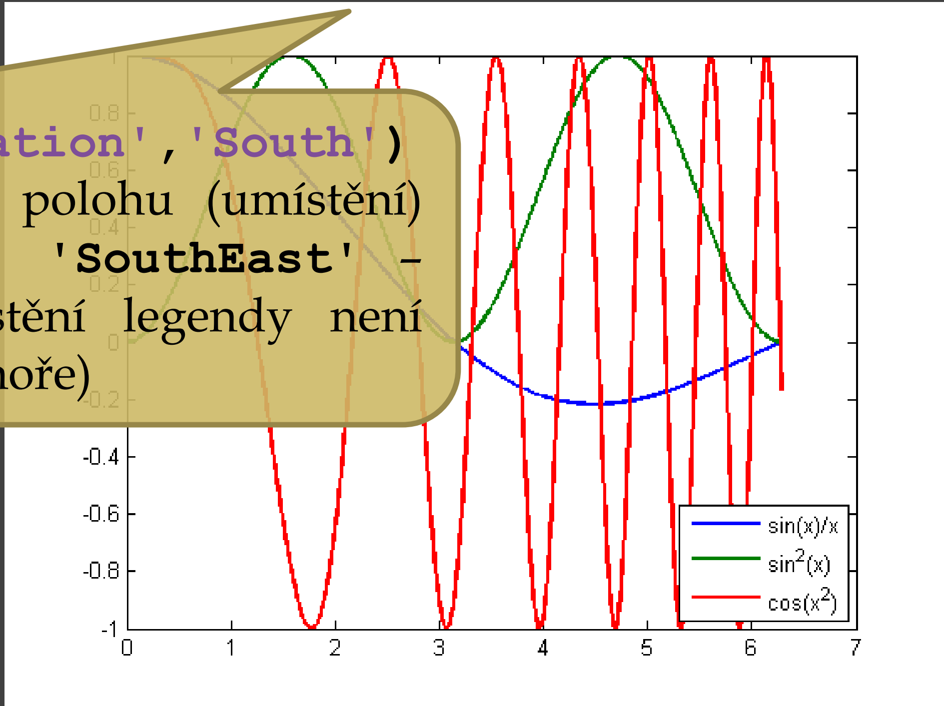
Dvojdimenzionální grafy

Pokračování příkladu:

Vykreslení grafů funkcí $y_1 = \sin(x)/x$, $y_2 = \sin^2(x)$, $y_3 = \cos(x^2)$.

```
function nekolik_grafu(a)
x = a.';  % sloupcový vektor x (transponovaný a)
y1 = sin(x)./x;                  % jeden sloupcový vektor
y23 = [sin(x).^2,cos(x.^2)]; % dva sloupce
plot(x,y1,x,y23)
legend('sin(x)/x','sin^2(x)','cos(x^2)','Location','SouthEast')
end
```

`legend('krivka1', 'krivka2','Location','South')`
– parametrem `'Location'` lze nastavit polohu (umístění) legendy podle světových stran, (např. `'SouthEast'` – jihovýchod vpravo dole) , pokud umístění legendy není nastaveno, je na severovýchodě (vpravo nahoře)



Volání funkce pro x od **0** do 4π :

```
x = [0:0.01:2*pi];
nekolik_grafu(x)
```

Grafy

Grafy zobrazující vztah částí k celku (statistika)

`bar` – sloupcový graf
`pie` – výsečový (koláčový) graf
`barh` – horizontální sloupcový graf

`bar3` – trojrozměrný sloupcový graf
`pie3` – trojrozměrný výsečový (koláčový) graf
`bar3h` – trojrozměrný horizontální sloupcový graf

Platí pro MATLAB, v jiných výpočetních systémech nemusí být trojrozměrné grafy k dispozici

Grafy

Příklad:

V bufetu bylo za 15 minut prodáno 10 rohlíků, 20 baget, 30 koblih a 5 kusů ostatního pečiva. Vytvořte výsečový graf, který ukáže podíl prodaného druhu pečiva.

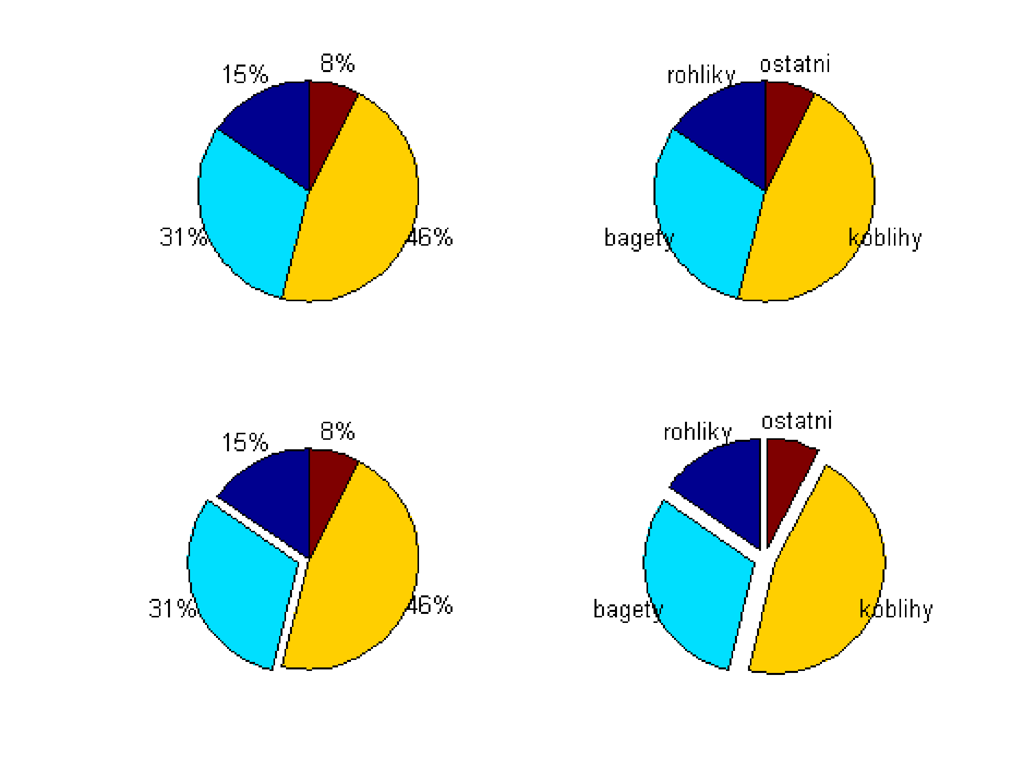
```
pecivo = [10,20,30,5];
subplot(2,2,1)
pie(pecivo)
subplot(2,2,2)
pie(pecivo,{'rohliky',...   % tři tečky – pokračování na další řádce
'bagety','koblihy','ostatni'})
subplot(2,2,3)
ex=[0,1,0,0];
pie(pecivo,ex)
subplot(2,2,4)
pie(pecivo,[1,1,1,1],...
{'rohliky','bagety','koblihy','ostatni'})
```

Grafy

Příklad:

V bufetu bylo za 15 minut prodáno 10 rohlíků, 20 baget, 30 koblih a 5 kusů ostatního pečiva. Vytvořte výsečový graf, který ukáže podíl prodaného druhu pečiva.

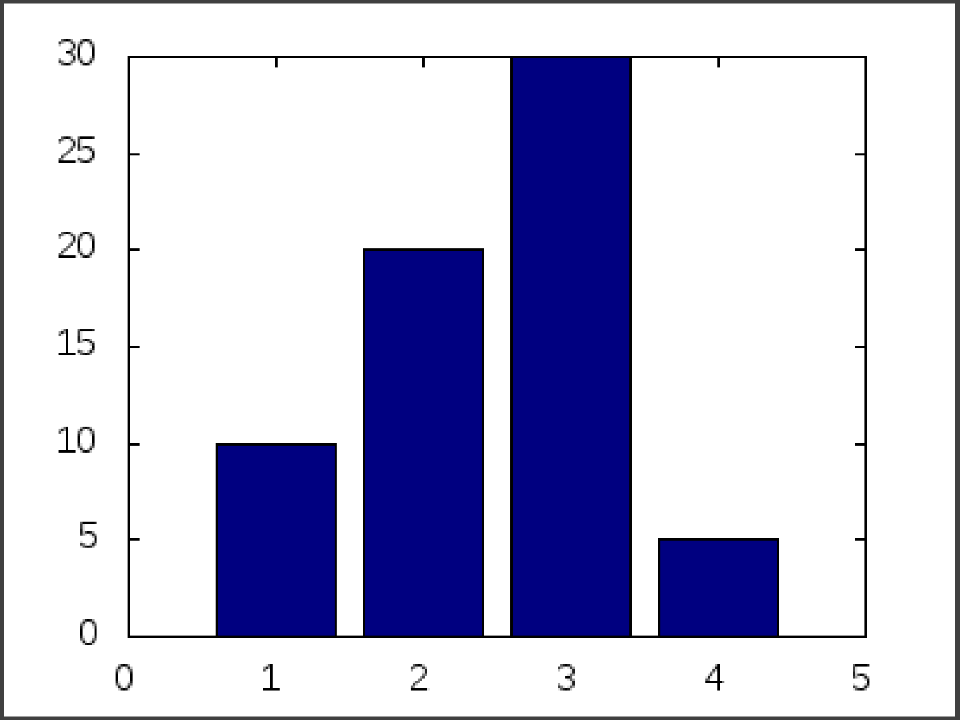
```
pecivo = [10,20,30,5];
subplot(2,2,1)
pie(pecivo)
subplot(2,2,2)
pie(pecivo,{'rohliky',
'bagety','koblihy','os
subplot(2,2,3)
ex=[0,1,0,0];
pie(pecivo,ex)
subplot(2,2,4)
pie(pecivo,[1,1,1,1],...
{'rohliky','bagety','koblihy','ostatni'})
```



Grafy

Pokračování příkladu:
V bufetu bylo za 15 minut prodáno **10** rohlíků, **20** baget, **30** koblih a **5** kusů ostatního pečiva. Vykreslete rovněž toto prodané pečivo ve formě sloupcového grafu

```
pecivo = [10,20,30,5];
bar(pecivo)
```



Grafy

Příklad:

V bufetu bylo za 15 minut prodáno 10 rohlíků, 20 baget, 30 koblih a 5 kusů ostatního pečiva, ve druhé prodejně bylo prodáno za stejnou dobu prodáno 8 rohlíků, 12 baget, 4 koblih a 15 kusů ostatního pečiva a ve třetí prodejně bylo prodáno opět za stejnou dobu prodáno 20 rohlíků, 4 bagety, 7 koblih a 5 kusů ostatního pečiva. Vykreslete poměrné zastoupení prodaného pečiva v jednotlivých prodejnách.

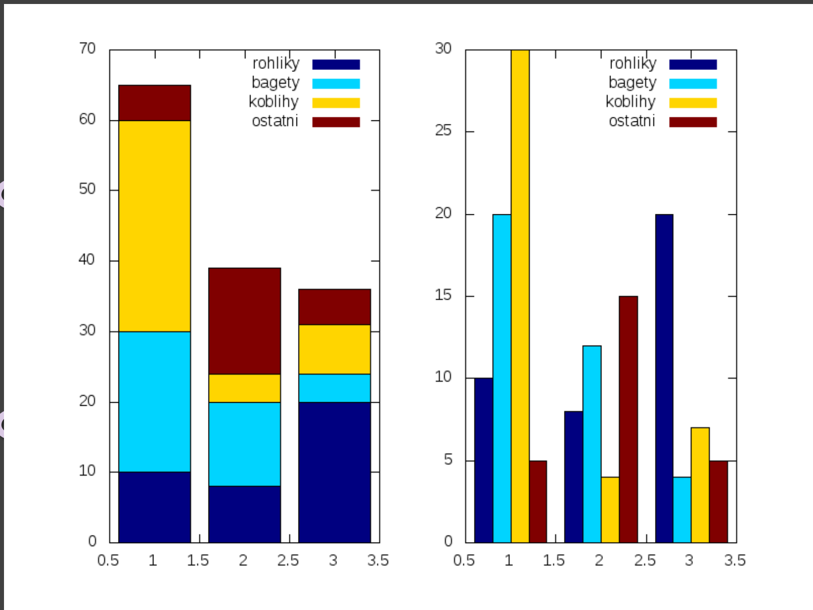
```
prodejna=[10,20,30,5; 8,12,4,15; 20,4,7,5]
subplot(1,2,1)
bar(prodejna,'stacked')
legend('rohliky','bagety','koblihy','ostatni')
subplot(1,2,2)
bar(prodejna,'grouped')
legend('rohliky','bagety','koblihy','ostatni')
```

Grafy

Příklad:

V bufetu bylo za 15 minut prodáno 10 rohlíků, 20 baget, 30 koblih a 5 kusů ostatního pečiva, ve druhé prodejně bylo prodáno za stejnou dobu prodáno 8 rohlíků, 12 baget, 4 koblih a 15 kusů ostatního pečiva a ve třetí prodejně bylo prodáno opět za stejnou dobu prodáno 20 rohlíků, 4 bagety, 7 koblih a 5 kusů ostatního pečiva. Vykreslete poměrné zastoupení prodaného pečiva v jednotlivých prodejnách.

```
prodejna=[10,20,30,5; 8,12,4,15; 20,4,7,5]
subplot(1,2,1)
bar(prodejna,'stacked')
legend('rohliky','bagety','ko
subplot(1,2,2)
bar(prodejna,'grouped')
legend('rohliky','bagety','ko
```



Grafy

Další typy grafů

`stairs` – "schodový" graf

`hist` – graf vyjadřující u spojité proměnné rozdělení četností podle intervalů hodnot. `hist(y)` rozdělí prvky *y* na 10 rovnoměrně rozložených intervalů a vrací počet prvků v každém intervalu. `hist(y,x)`, kde x je vektor, vrátí rozdělení *y* mezi intervaly s centry určenými *x*

`rose` – úhlový histogram

`errorbar` – chybové úsečky – ukazují úroveň spolehlivosti dat nebo odchylku podél křivky.

`feather` – graf zobrazující vektory vycházející z bodů rovnoměrně rozložených podél vodorovné osy, tj. z [0,0], [1,0], [2,0], …

a další … viz `help`

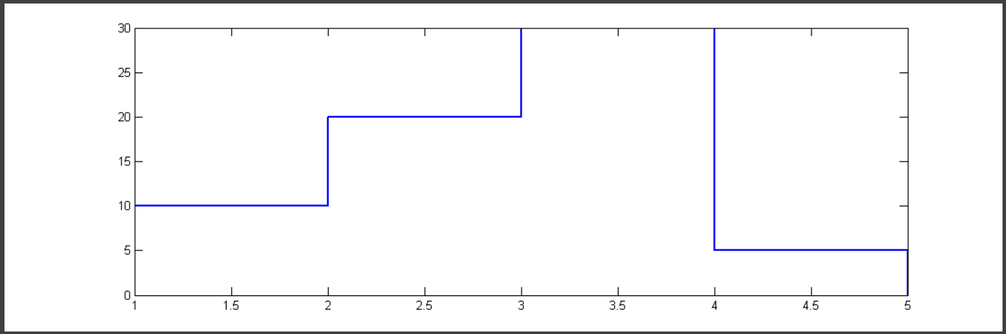
Grafy

stairs – "schodový" graf

Příklady:

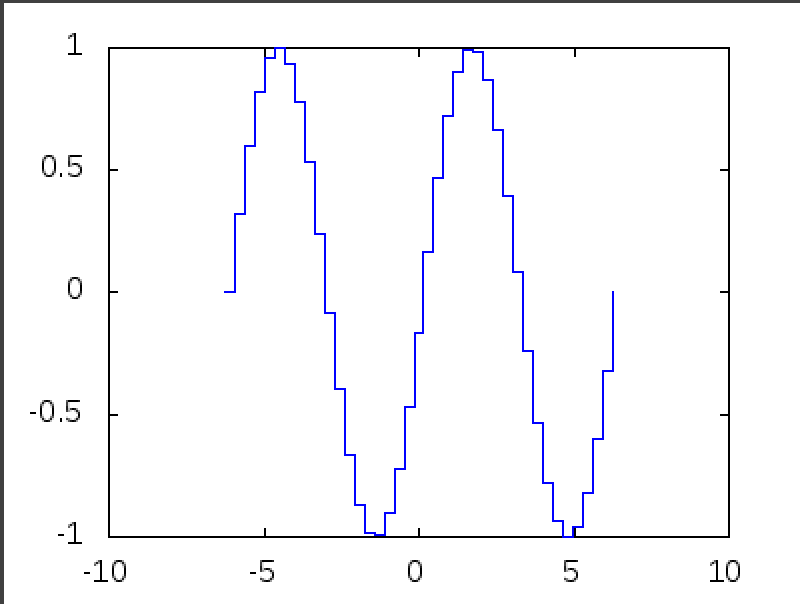
V bufetu bylo za 15 minut prodáno 10 rohlíků, 20 baget, 30 koblih a 5 kusů ostatního pečiva. Vykreslete rovněž toto prodané pečivo ve formě schodového grafu

```
pecivo = [10,20,30,5,0];
stairs(pecivo)
```



Zobrazení funkce sin(x)

```
x = linspace(-2*pi,2*pi,40);
stairs(x,sin(x))
```

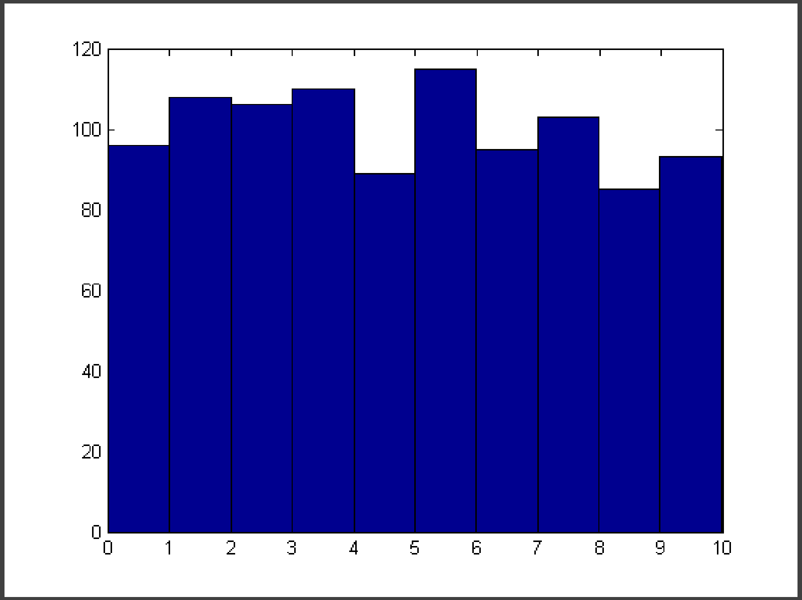


Grafy

hist – graf vyjadřující u spojité proměnné rozdělení četností podle intervalů hodnot. **hist(y)** rozdělí prvky *y* na 10 rovnoměrně rozložených intervalů a vrací počet prvků v každém intervalu. **hist(y,x)**, kde x je vektor, vrátí rozdělení *y* mezi intervaly s centry určenými *x*

Příklady:

```
a=rand(1,1000).*10;
hist(a)
```

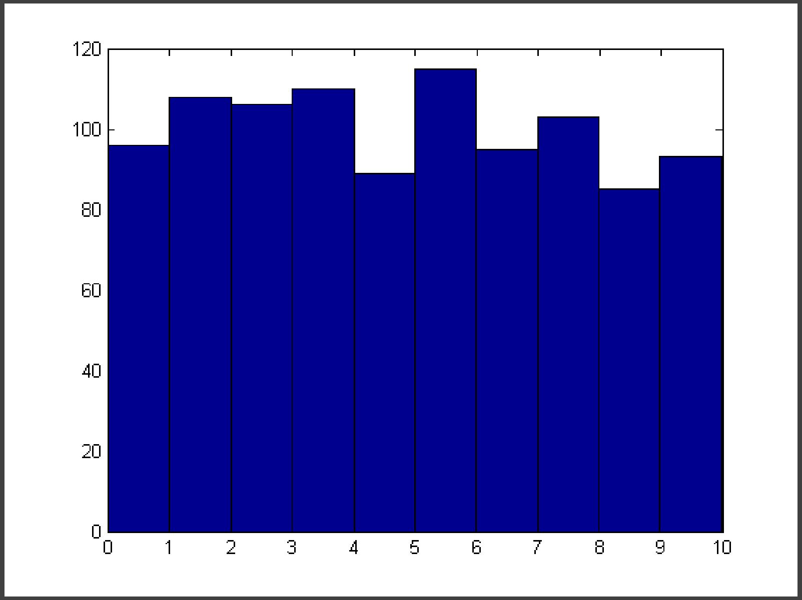


Grafy

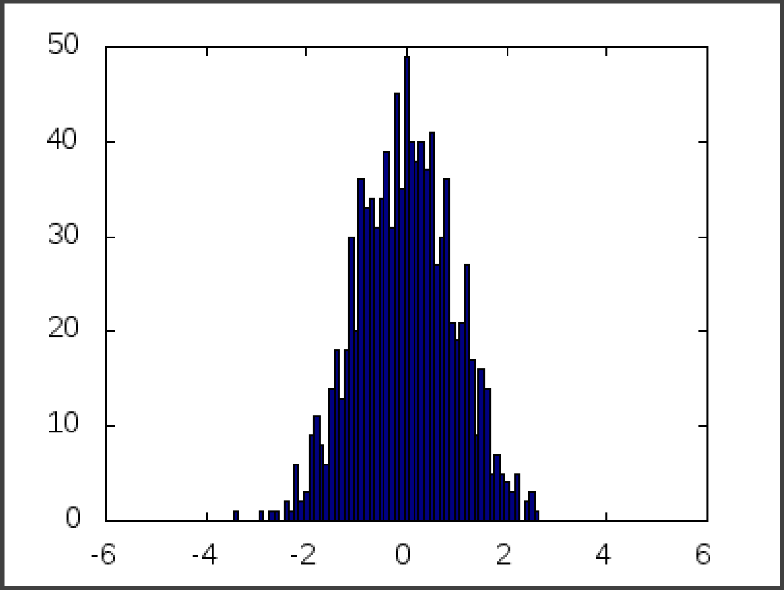
hist – graf vyjadřující u spojité proměnné rozdělení četností podle intervalů hodnot. **hist(y)** rozdělí prvky *y* na 10 rovnoměrně rozložených intervalů a vrací počet prvků v každém intervalu. **hist(y,x)**, kde x je vektor, vrátí rozdělení *y* mezi intervaly s centry určenými *x*

Příklady:

```
a=rand(1,1000).*10;
hist(a)
```




```
x = -5:0.1:5;
y = randn(1000,1);
hist(y,x)
```



Grafy

rose – úhlový histogram

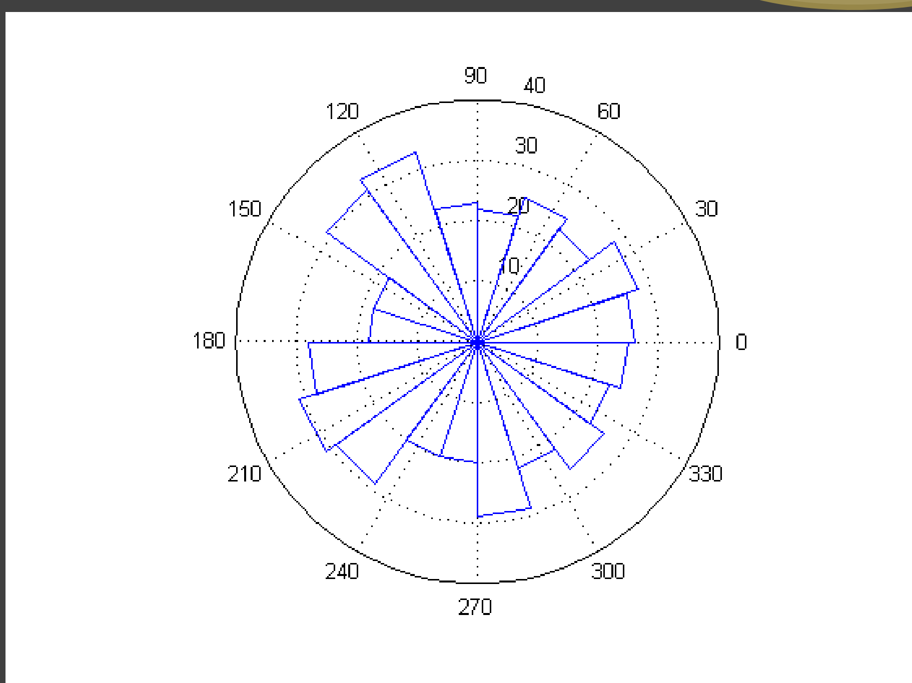
V nových verzích MATLABu se setkáme i s příkazem **polarhistogram(uhel,20)**

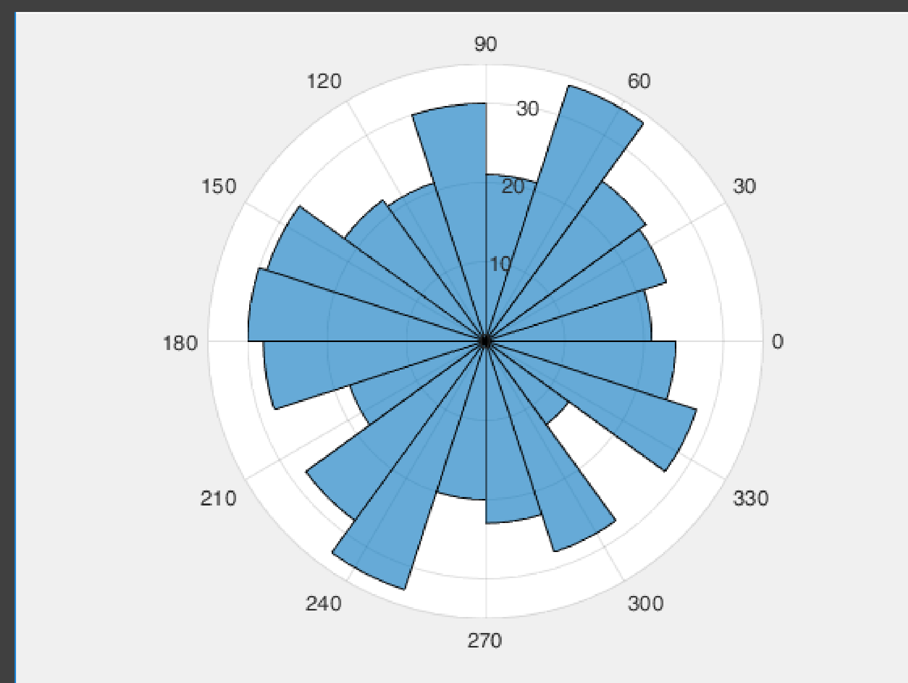
Příklad:

```
uhel = rand(1,500).*2.*pi;
rose(uhel)
```

Prvky vektoru uhel musí být zadány v radiánech.

Počet dílů – zde zvoleno 20



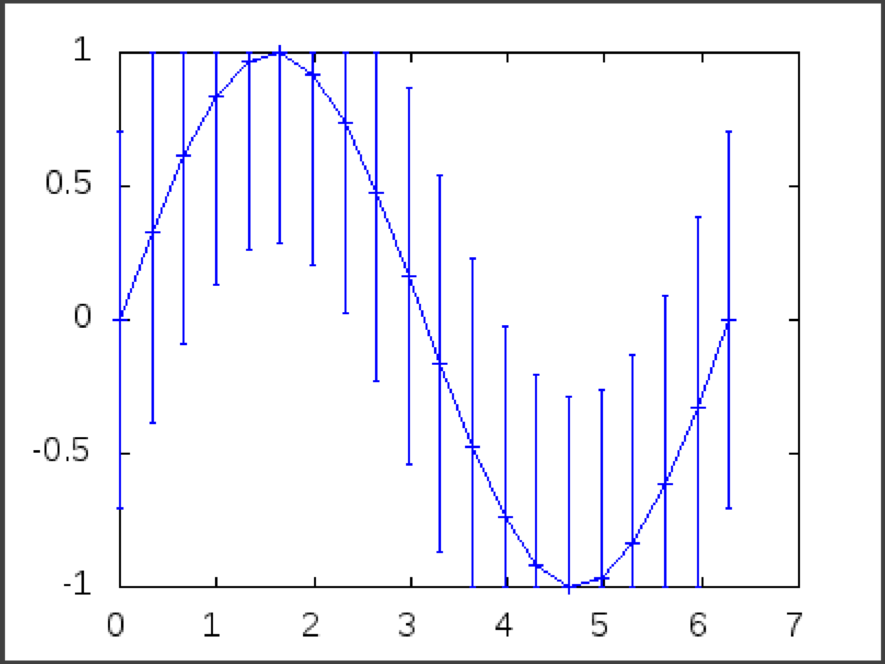


Grafy

errorbar – chybové úsečky – ukazují úroveň spolehlivosti dat nebo odchylku podél křivky.

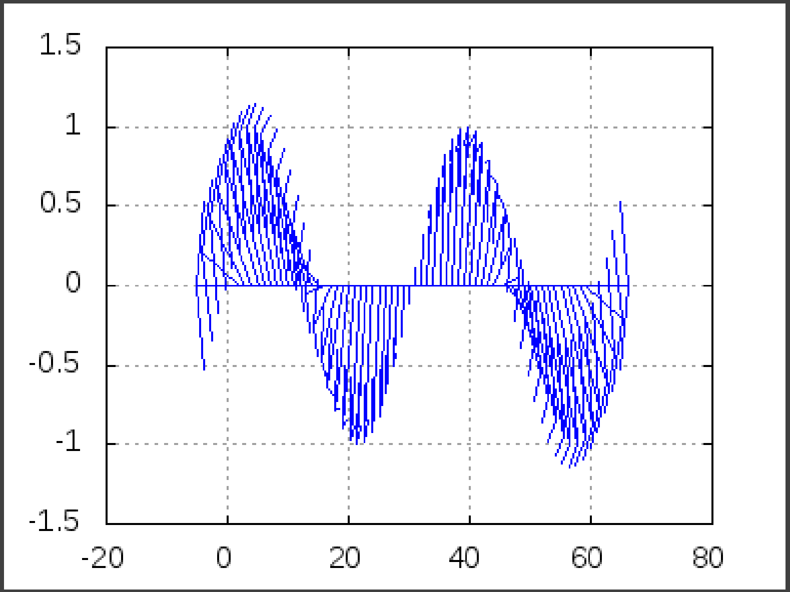
Příklad:

```
x = linspace(0,2*pi,10);
y = sin(x);
er = std(y)*ones(size(x));
errorbar(x,y,er)
```



Grafy

feather – graf zobrazující vektory vycházející z bodů rovnoměrně rozložených podél vodorovné osy, tj. z [0,0], [1,0], [2,0], [3,0], …



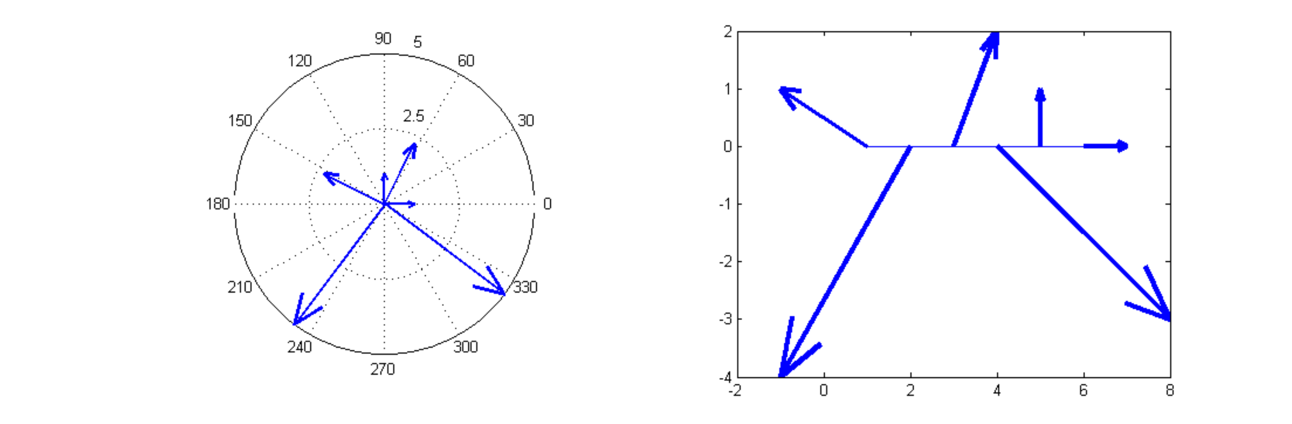
Příklady:

Zobrazení funkce sin(*x*)

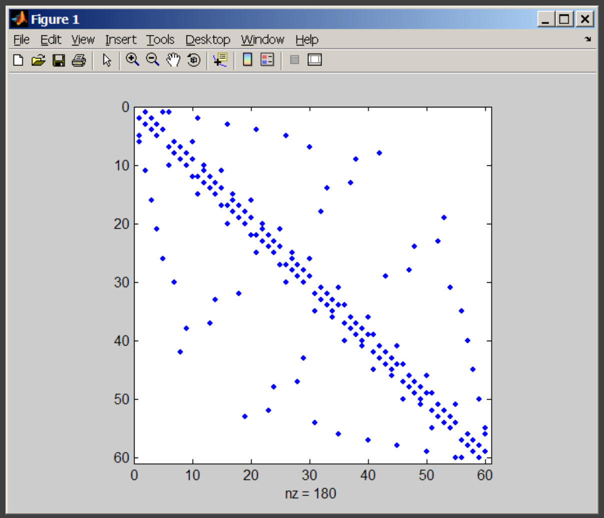
```
x = linspace(-2*pi,2*pi,40);
feather(x,sin(x))
grid
```

Zobrazení komplexních čísel

```
C = [-2+i,-3-4i,1+2i,4-3i,i,1];
subplot(1,2,1)
compass(C)
subplot(1,2,2)
feather(C)
```



Grafy



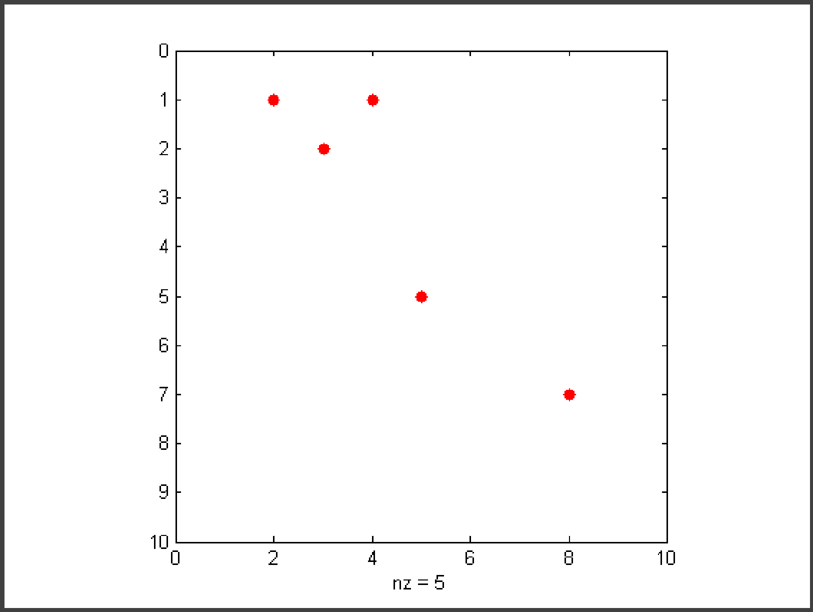
spy – grafické zobrazení řídké matice (pro analýzu řídkých matic)

Příklad:

```
V = zeros(9);
V(1,2) = 7;
V(2,3) = 6;
V(1,4) = 9;
V(5,5) = 4;
V(7,8) = 5;
spy(V,'r')
```

% body mi v grafu ukáží nenulové hodnoty

```
V =
   0   7   0   9   0   0   0   0   0
   0   0   6   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0
   0   0   0   0   4   0   0   0   0
   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   5   0
   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0
```



Grafy

Řízení vzhledu textů v grafech – příkazy LaTeXu

dolní index – t_1 se zapíše `t_1`

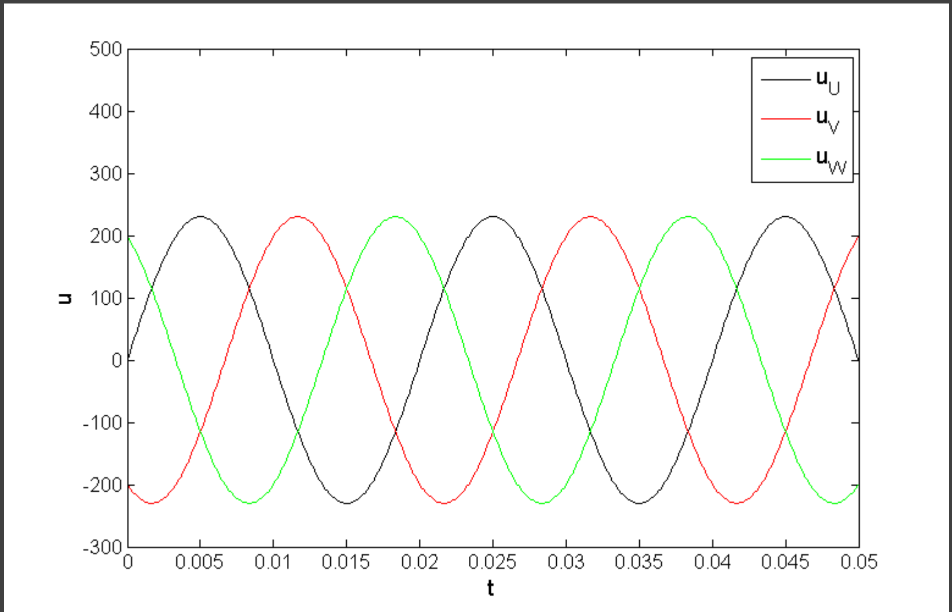
horní index – t^2 se zapíše `t^2`

- má-li platit příkaz pro více znaků – použijeme `{ }`

např. `x_{23}` vytiskne x_{23}

Příklad:

```
t=[0:1e-4:5e-2];
uU=230.*sin(2.*pi.*50.*t);
uV=230.*sin(2.*pi.*50.*t-2*pi/3);
uW=230.*sin(2.*pi.*50.*t+2*pi/3);
plot(t,uU,'k')
hold on
plot(t,uV,'r')
plot(t,uW,'g')
hold off
xlabel('t')
ylabel('u')
legend('u_U','u_V','u_W')
axis([0,0.05,-300,500])
```



Grafy

Řízení vzhledu textů v grafech – příkazy LaTeXu

dolní index – t_1 se zapíše **t_1**

horní index – t^2 se zapíše **t^2**

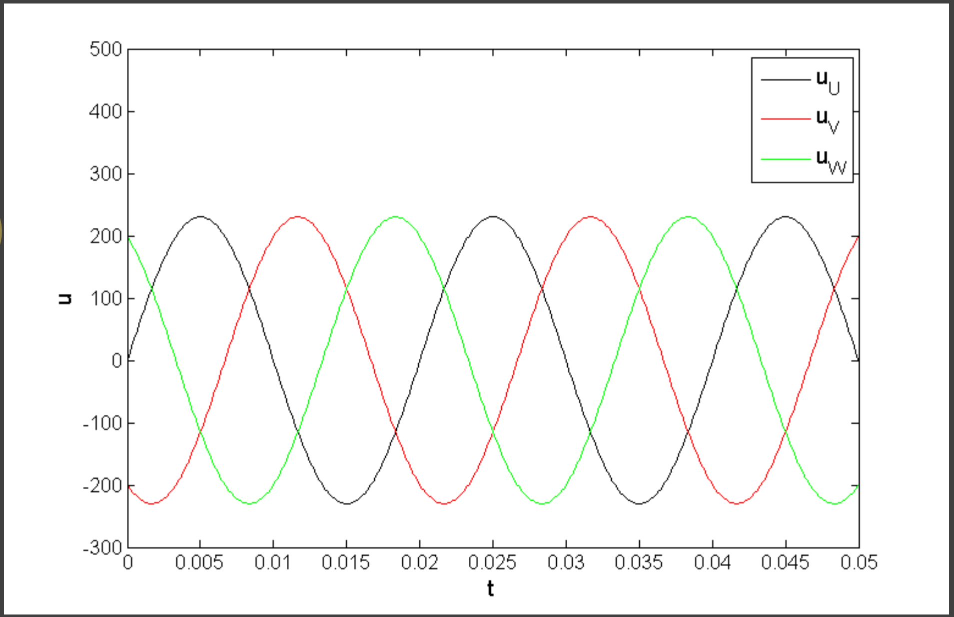
- má-li platit příkaz pro více znaků – použijeme **{ }**

např. **x_{23}** vytiskne x_{23}

Příklad:

```
t=[0:1e-4:5e-2];
uU=230.*sin(2.*pi.*50.*t);
uV=230.*sin(2.*pi.*50.*t-2*pi/3);
uW=230.*sin(2.*pi.*50.*t+2*pi/3);
plot(t,uU,'k')
hold on
plot(t,uV,'r')
plot(t,uW,'g')
hold off
xlabel('t')
ylabel('u')
legend('u_U','u_V','u_W')
axis([0,0.05,-300,500])
```

dolní index se zapíše pomocí _



Grafy

Řízení vzhledu textů v grafech – příkazy LaTeXu

dolní index – t_1 se zapíše `t_1`

horní index – t^2 se zapíše `t^2`

- má-li platit příkaz pro více znaků – použijeme `{ }`

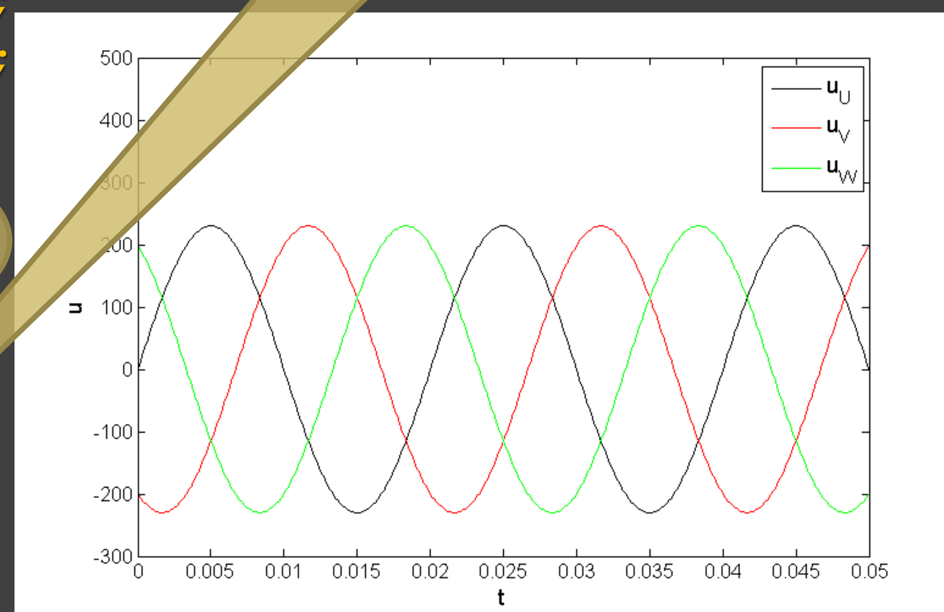
např. `x_{23}` vytiskne x_{23}

Příklad:

```
t=[0:1e-4:5e-2];
uU=230.*sin(2.*pi.*50.*t);
uV=230.*sin(2.*pi.*50.*t-2*pi/3);
uW=230.*sin(2.*pi.*50.*t+2*pi/3);
plot(t,uU,'k')
hold on
plot(t,uV,'r')
plot(t,uW,'g')
hold off
xlabel('t')
ylabel('u')
legend('u_U','u_V','u_W')
axis([0,0.05,-300,500])
```

změna rozsahu os
- viz dále

dolní index
se zapíše
pomocí _



Grafy

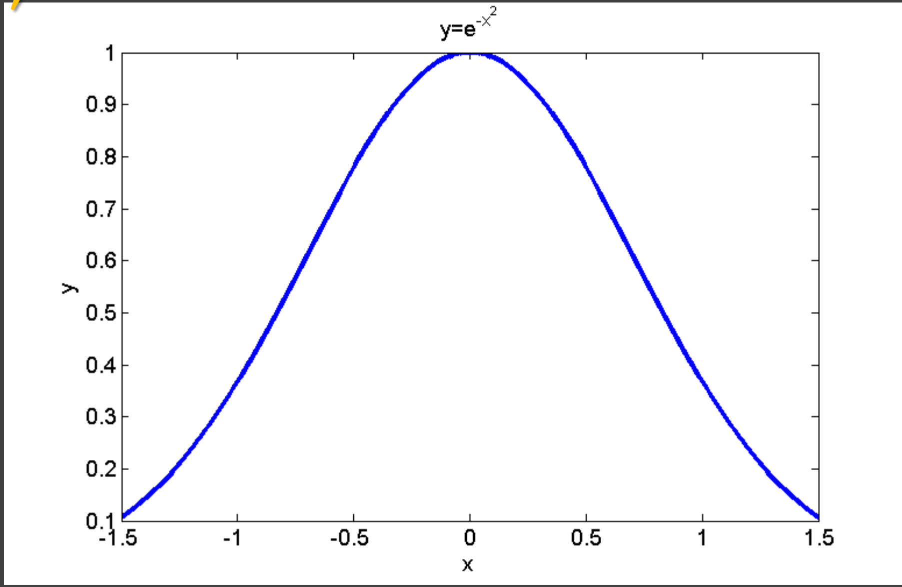
Příklad:

vykreslení grafu funkce $y = e^{-x^2}$, kde x je z intervalu od **-1.5** do **1.5**

```
function y = enaminusxna2(x)
y=exp(-x.^2);
end
```

Volání funkce a vykreslení grafu

```
x = [-1.5:.01:1.5];
vysledek = enaminusxna2(x);
plot(x,vysledek)
title('y=e^{-x^2}')
xlabel('x')
ylabel('y')
```



Grafy

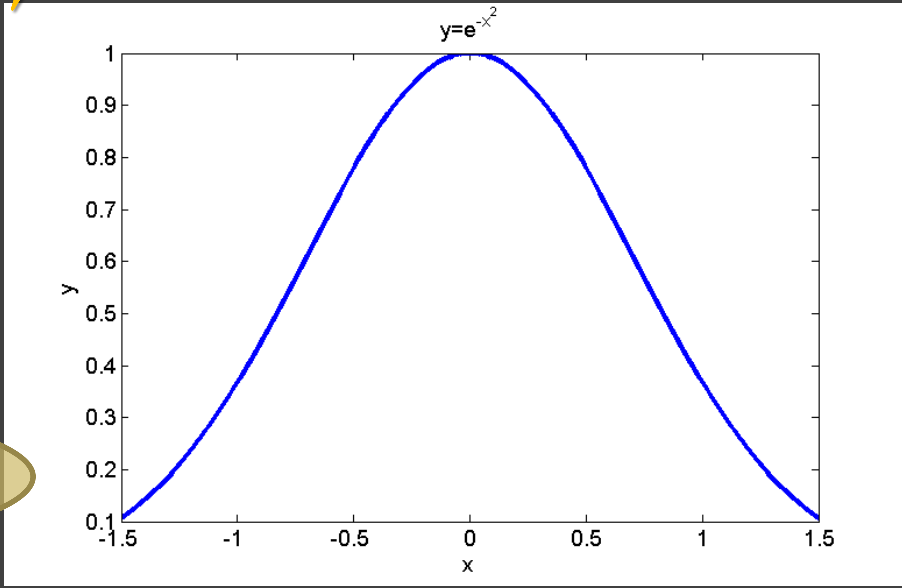
Příklad:
vykreslení grafu funkce $y = e^{-x^2}$, kde x je z intervalu od **-1.5** do **1.5**

```
function y = enaminusxna2(x)
y=exp(-x.^2);
end
```

před znakem ^ tečka – operace . ^ , umocnění proběhne prvek po prvku

Volání funkce a vykreslení grafu

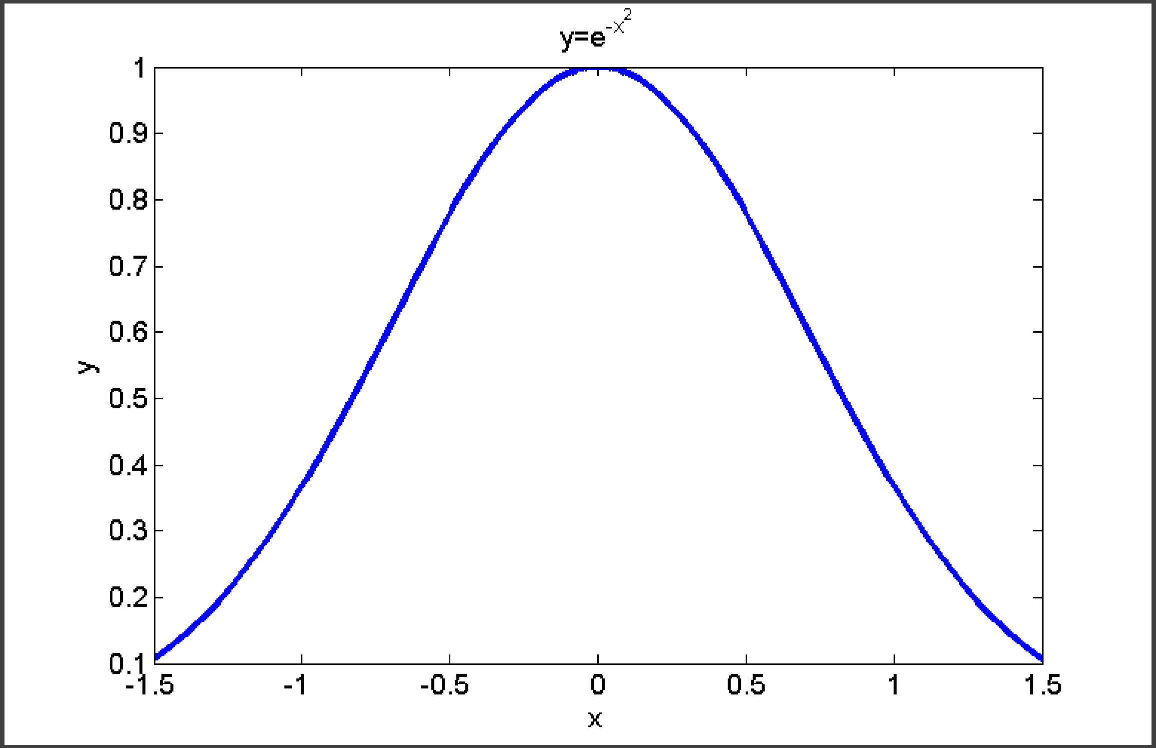
```
x = [-1.5:.01:1.5];
vysledek = enaminusxna2(x);
plot(x,vysledek)
title('y=e^{-x^2}')
xlabel('x')
ylabel('y')
```



horní index se zapíše pomocí ^ (řízení vzhledu textů)

Grafy

Pokračování příkladu:
– měřítko pro osy se volí automaticky



axis

- změna měřítka os,
- bez parametrů vrátí vektor s rozsahy os

např.: pro předchozí graf

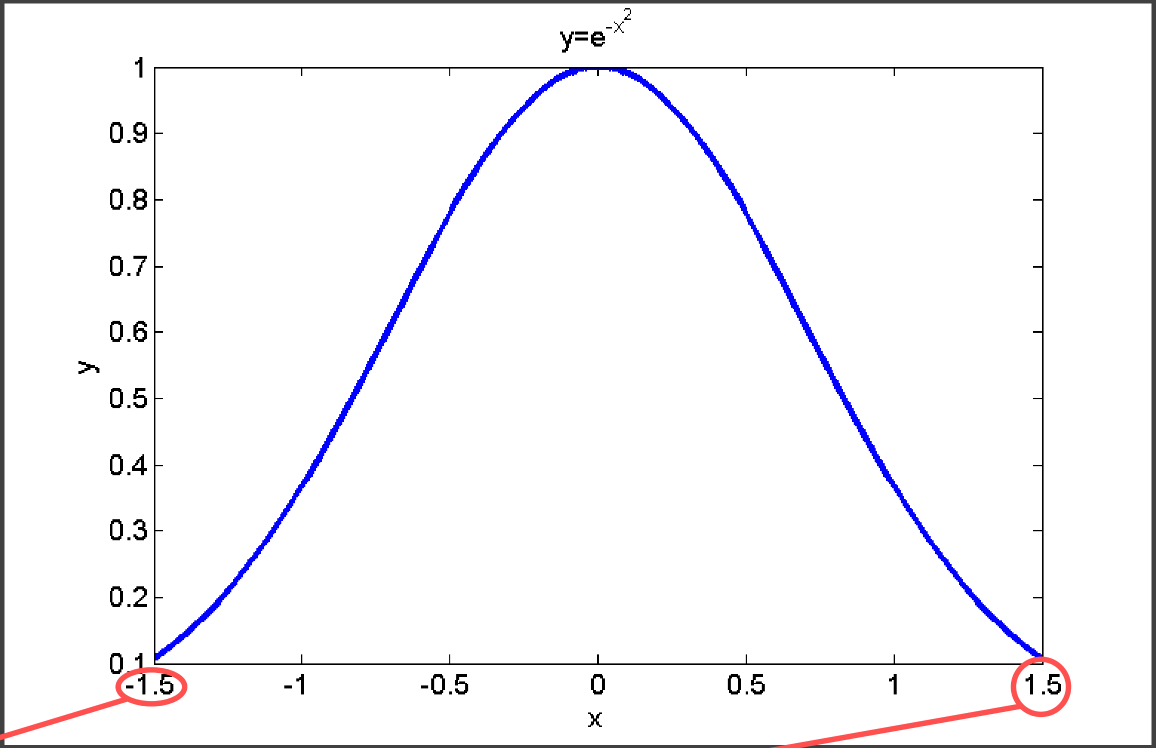
```
axis
ans =
   -1.5000   1.5000   0.1000   1.0000
```

Grafy

Pokračování příkladu:
– měřítko pro osy se volí automaticky

axis

- změna měřítka os,
- bez parametrů vrátí vektor s rozsahy os

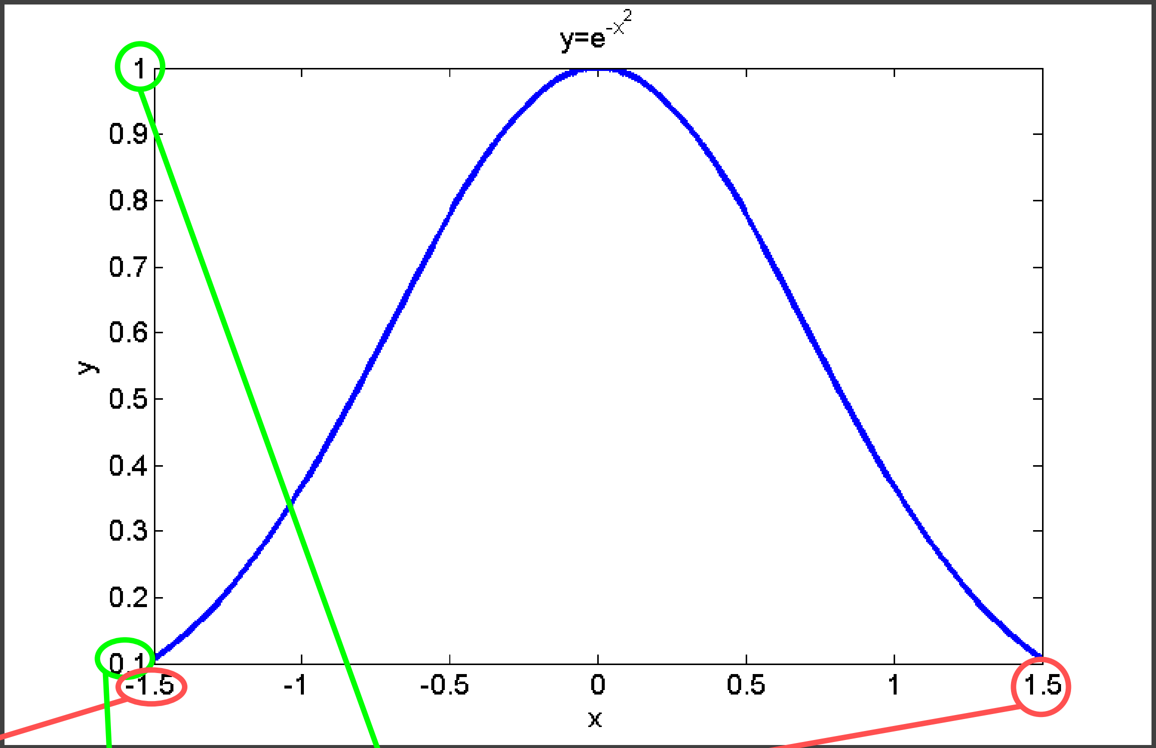


např.: pro předchozí graf

```
axis
ans =
  -1.5000  1.5000  0.1000  1.0000
```

Grafy

Pokračování příkladu:
– měřítko pro osy se volí automaticky



axis

- změna měřítka os,
- bez parametrů vrátí vektor s rozsahy os

např.: pro předchozí graf

```
axis
ans =
  -1.5000  1.5000  0.1000  1.0000
```

Grafy

Je-li zadán čtyřprvkový vektor, např.

`c = [xmin, xmax, ymin, ymax]`

`axis(c);` – nastaví měřítko podle předpisu ve vektoru `c`

Lze též zapsat pro dvourozměrné grafy takto:

`axis([xmin, xmax, ymin, ymax]);`

nebo pro trojrozměrné grafy pak takto:

`axis([xmin, xmax, ymin, ymax, zmin, zmax]);`

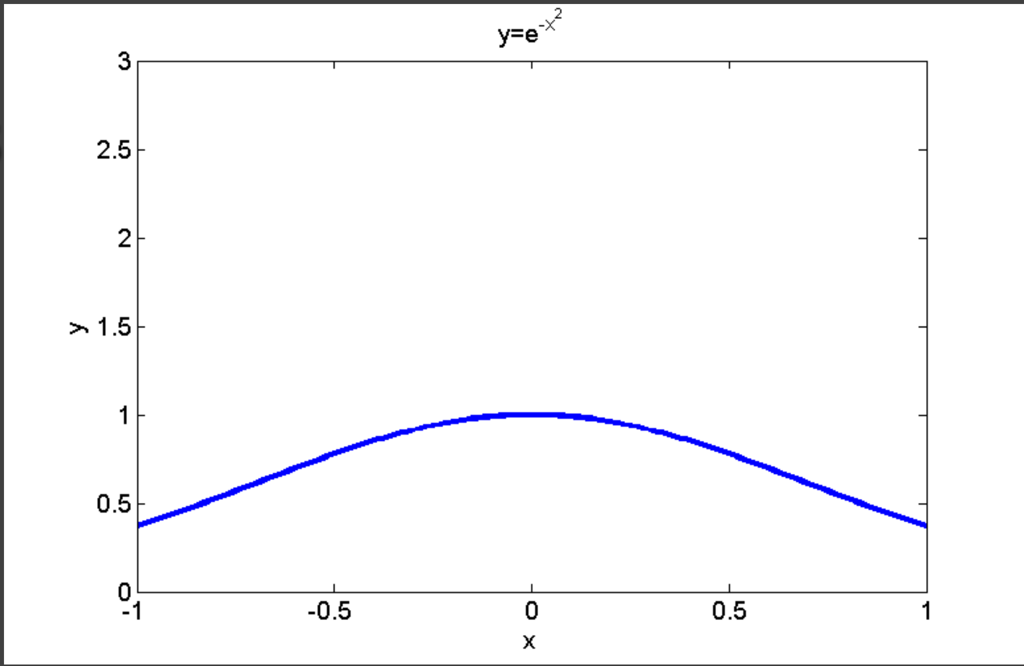
Pokračování příkladu:

graf $y = e^{-x^2}$ pro x od -1.5 do 1.5 se změnou měřítka os x a y

$x_{min} = -1$, $x_{max} = 1$,

$y_{min} = 0$, $y_{max} = 3$

`axis([-1,1,0,3])`



Grafy

Je-li zadán čtyřprvkový vektor, např.

```
c = [xmin, xmax, ymin, ymax]
```

`axis(c);` – nastaví měřítko podle předpisu ve vektoru `c`

Lze též zapsat pro dvourozměrné grafy takto:

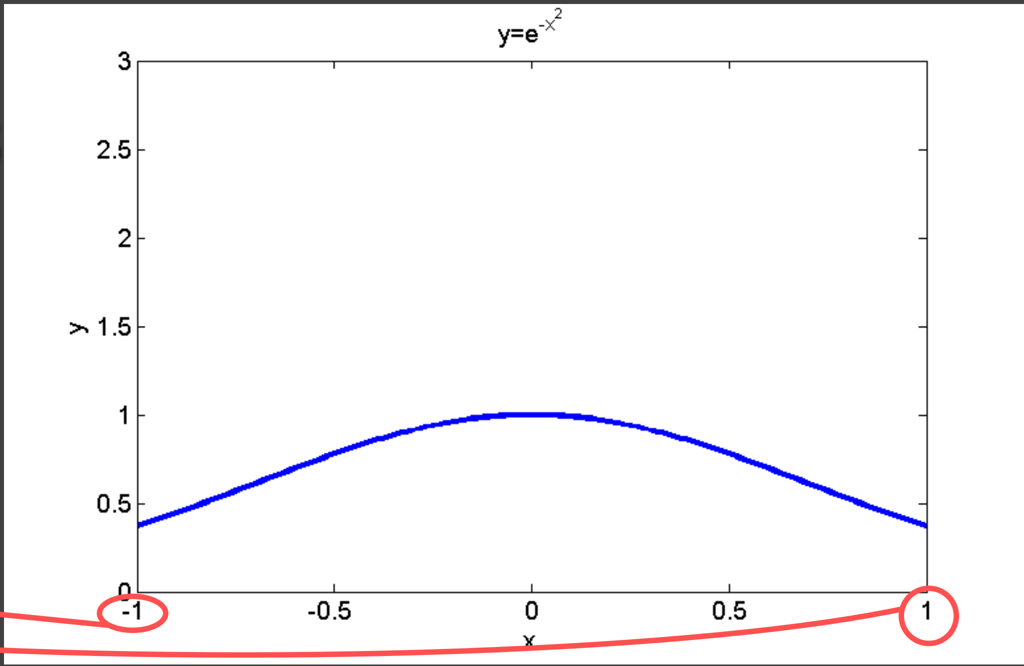
```
axis([xmin, xmax, ymin, ymax]);
```

nebo pro trojrozměrné grafy pak takto:

```
axis([xmin, xmax, ymin, ymax, zmin, zmax]);
```

Pokračování příkladu:
graf $y = e^{-x^2}$ pro x od -1.5 do 1.5
se změnou měřítka os x a y
$x_{min} = -1$, $x_{max} = 1$,
$y_{min} = 0$, $y_{max} = 3$

```
axis([-1,1,0,3])
```



Grafy

Je-li zadán čtyřprvkový vektor, např.

`c = [xmin, xmax, ymin, ymax]`

`axis(c);` – nastaví měřítko podle předpisu ve vektoru **`c`**

Lze též zapsat pro dvourozměrné grafy takto:

`axis([xmin, xmax, ymin, ymax]);`

nebo pro trojrozměrné grafy pak takto:

`axis([xmin, xmax, ymin, ymax, zmin, zmax]);`

Pokračování příkladu:

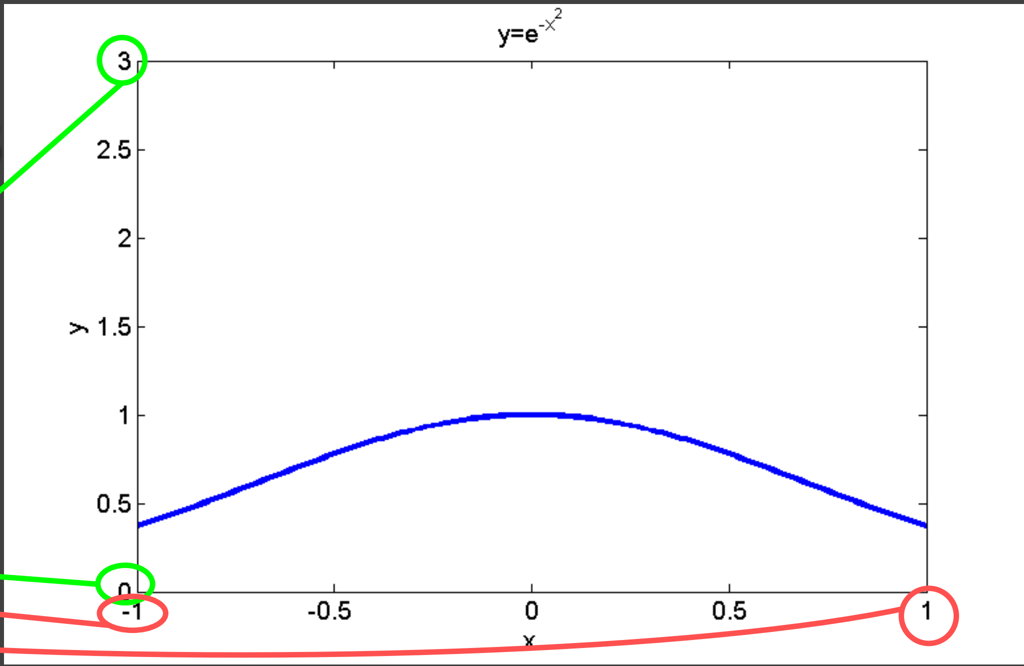
graf $y = e^{-x^2}$ pro x od -1.5 do 1.5

se změnou měřítka os x a y

$x_{min} = -1$, $x_{max} = 1$,

$y_{min} = 0$, $y_{max} = 3$

`axis([-1,1,0,3])`



Grafy

- samotný příkaz **axis** zmrazí do odvolání pro všechny grafy aktuální nastavení

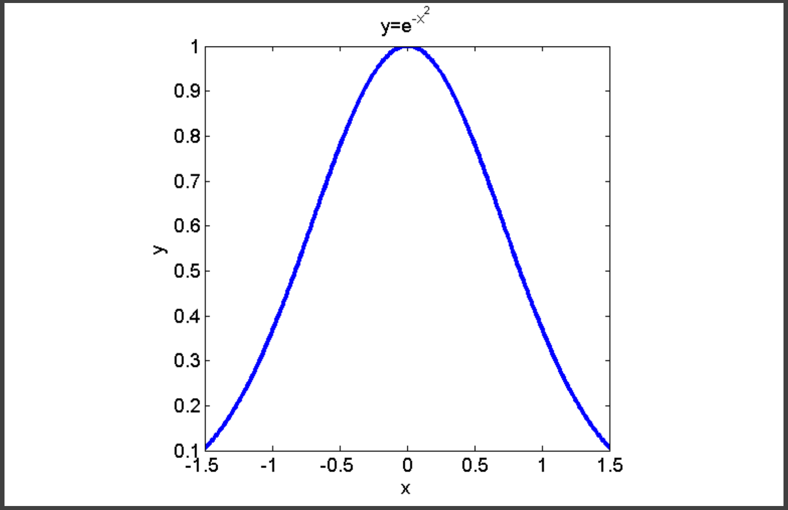
- zadá-li se příkaz **axis** ještě jednou, vrátí se nastavení na automatické měřítko

axis('square') - zajistí, že obě osy vytvoří čtvercovou oblast pro vykreslení grafu

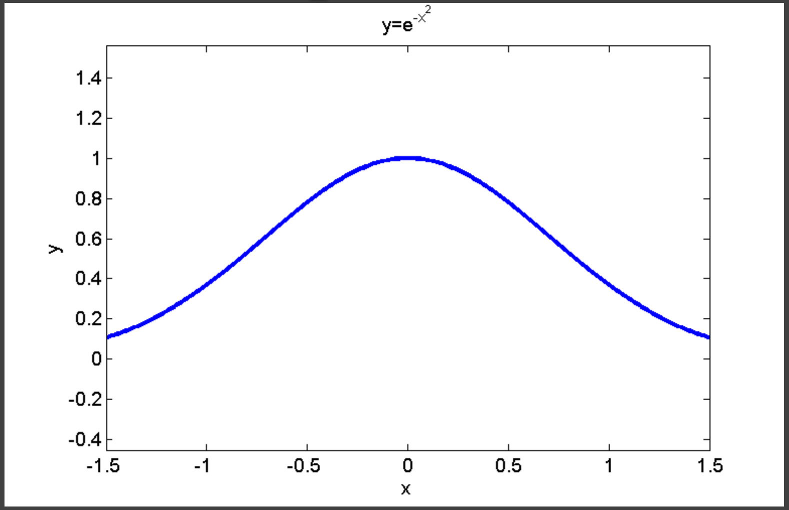
axis('equal') – stejný krok na obou osách, rovnost měřítek – tělesa ve 3D, např. koule, nebudou zmáčknutá, deformovaná

např.: pro předchozí graf $y = e^{-x^2}$ pro x od **-1.5** do **1.5**

axis('square')



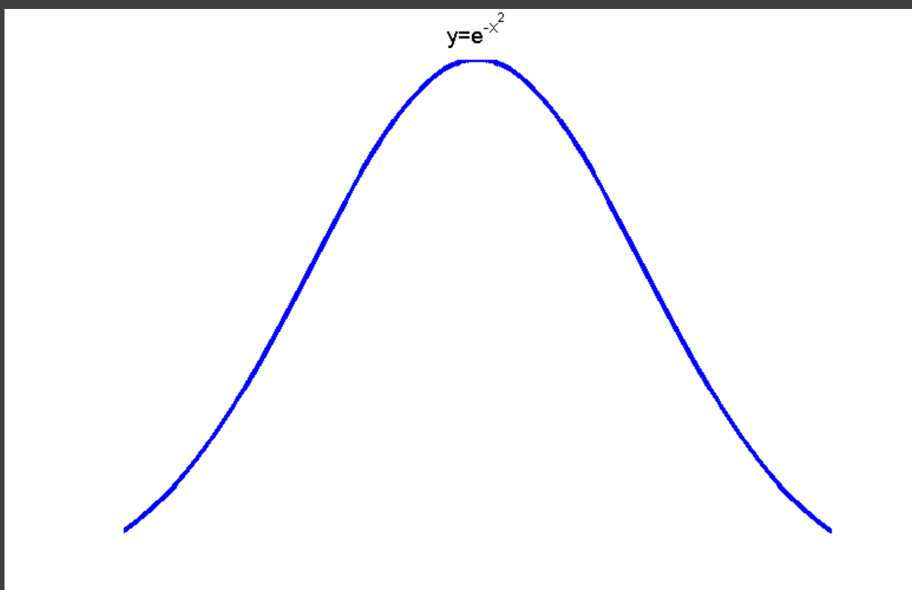
axis('equal')



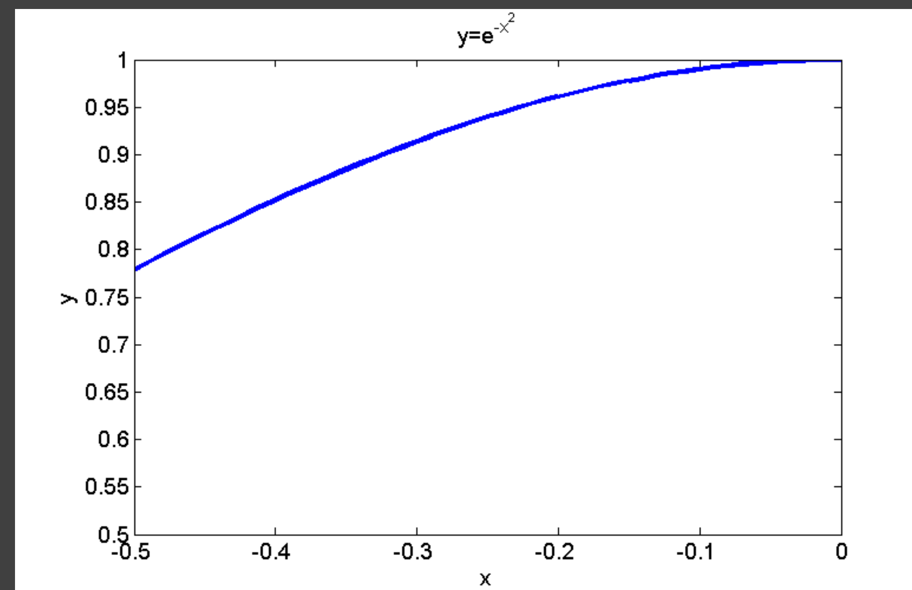
Grafy

- `axis off` – vypne osy v grafu
- `axis on` – opět zapne vypnuté osy

např.: pro předchozí graf $y = e^{-x^2}$
pro x od -1.5 do 1.5
`axis off`



`axis on`
a změna měřítka os x a y
`axis([-0.5,0,0.5,1]);`



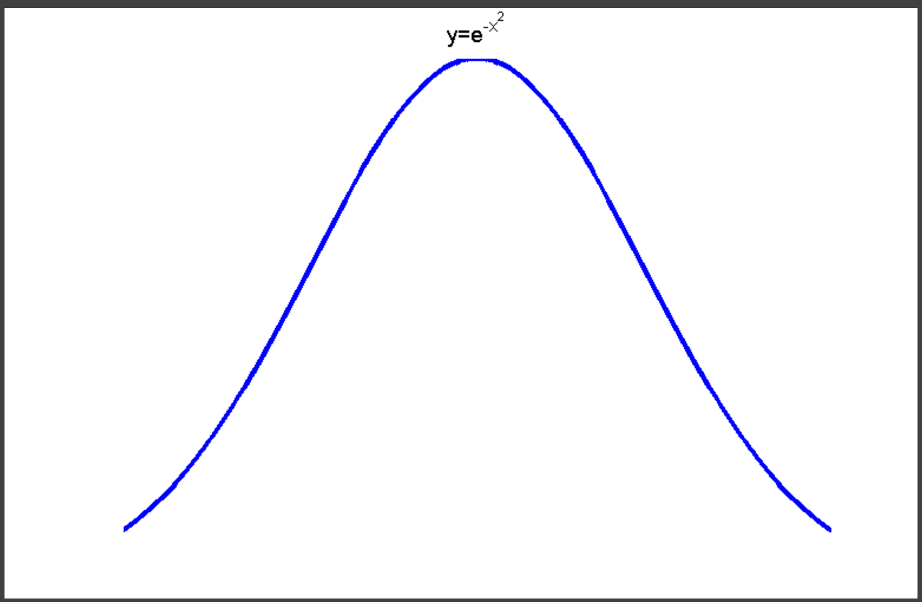
`axis('normal')` – návrat k výchozímu stavu co se týče "tvaru"
`axis('auto')` – návrat k výchozímu stavu co se týče mezí os

Grafy

- **axis off** – vypne osy v grafu
- **axis on** – opět zapne vypnuté osy

např.: pro předchozí graf $y = e^{-x^2}$
pro x od -1.5 do 1.5

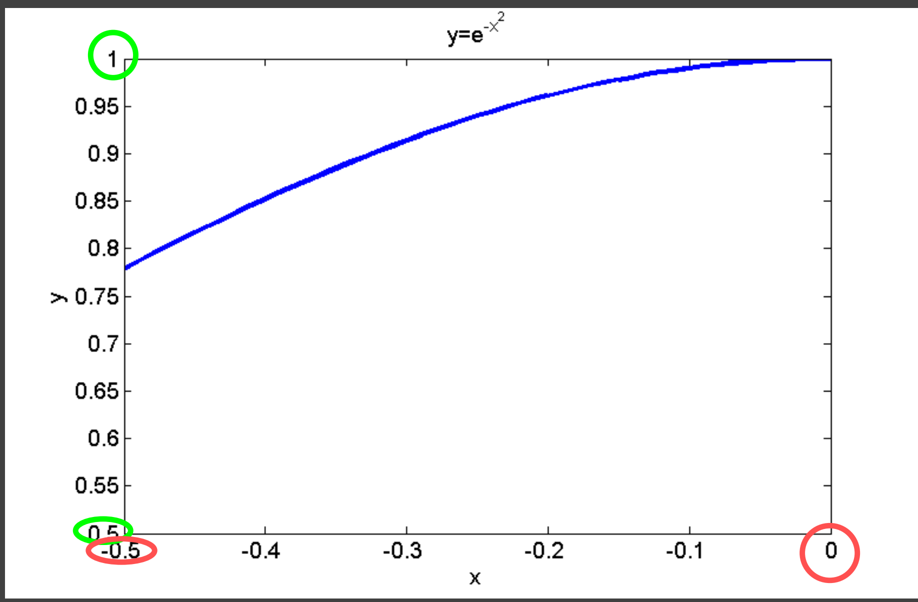
```
axis off
```




```
axis on
```

a změna měřítka os x a y

```
axis([-0.5,0,0.5,1]);
```



`axis('normal')` – návrat k výchozímu stavu co se týče "tvaru"
`axis('auto')` – návrat k výchozímu stavu co se týče mezí os

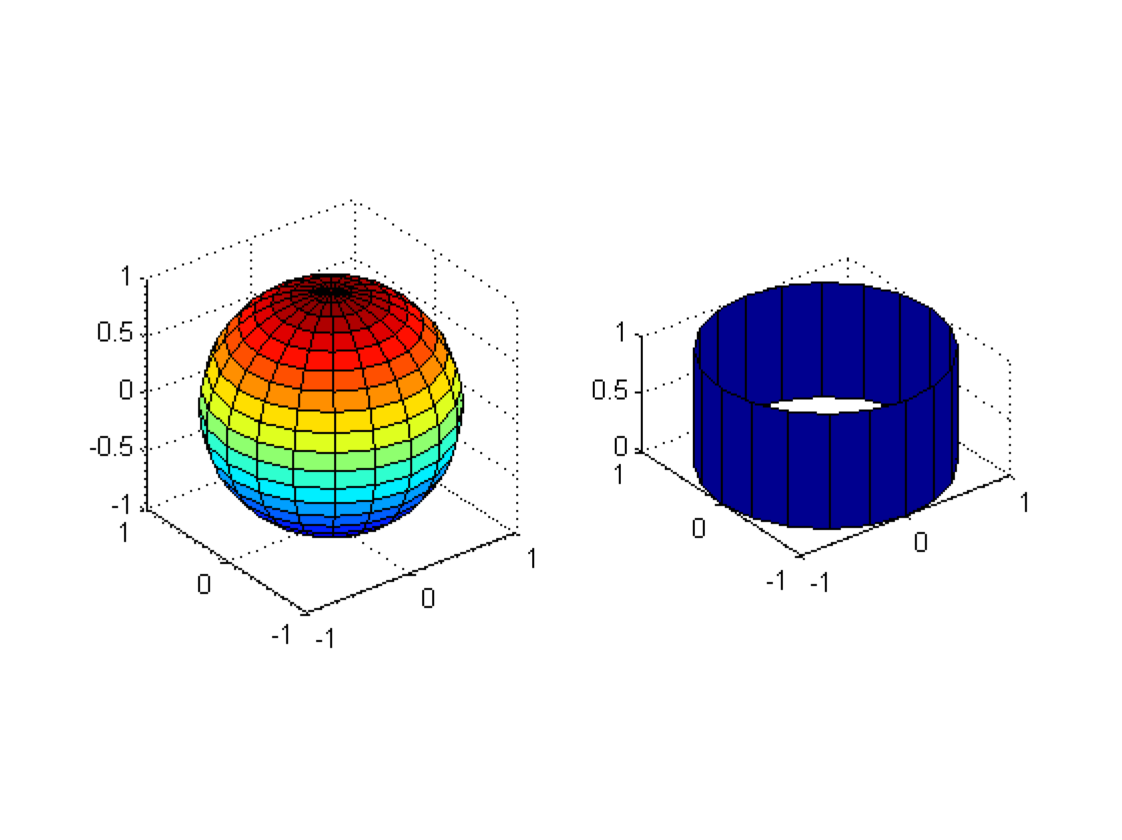
Grafy

- **axis image** – podobně jako **equal** – zajistí stejný krok na obou osách a k tomu nastaví osy přesně dle rozsahu dat

Příklad:
koule a válec pomocí generátoru křivek

```
subplot(1,2,1)
sphere % koule
axis image

subplot(1,2,2)
cylinder % valec
axis image
```



Další parametry viz **help axis**.

Grafy

Připomenutí:
vytváření **vektorů** – hodnot pro vodorovnou osu *x*, resp. *t* (možné i jiné značení) pro grafy a výpočty:

- vektorem – **výčtem** prvků v **hranatých závorkách** (při velkém počtu prvků nevhodné), např.

```
x = [0,1,2,5,6]
```

- vektorem – **výčtem** pomocí dvojtečky `[od : krok : do]` – použijeme tehdy, známe-li **meze** (`od` , `do`) a zvolenou **velikost kroku**, např.

```
x2 = 0 : 0.1 : 2*pi;
```

používáme-li dvojtečku, hranaté závorky nejsou nutné.

Grafy

Pokračování připomenutí:

- pomocí **linspace** – `linspace(od, do, počet_prvků)` – použijeme tehdy, známe-li **meze** (**od** , **do**) a **počet prvků** ve vektoru – vytvoří vektor s lineárním dělením s počtem prvků `počet_prvků` – pozor pokud uvedeme jen parametry **od** a **do**, počet prvků je automaticky **100**, např.

`x3 = linspace(3,15,45)` – vektor od 3 do 15 s 45 prvky.

- pomocí **logspace** – `logspace(od,do,počet_prvků)` – vytvoří vektor s logaritmickým dělením s **počtem prvků** `počet_prvků` – hodnoty jsou 10^{od} až 10^{do} – pozor pokud uvedeme jen parametry **od** a **do**, počet prvků je automaticky **50**, např.

`x4 = logspace(2,8,10)` – vektor od 10^2 až 10^8 s 10 prvky.

(vhodné např. bude-li potom graf vykreslován stejně pomocí `semilogx`, pak lineární dělení osy je někdy nevhodné)

Grafy

Příklad:

Amplitudová a fázová frekvenční charakteristika filtru

Filtr je obvod přenášející pouze vybrané frekvence a tlumící ostatní. Pasivní filtry se skládají z pasivních součástek (rezistor, cívka a kondenzátor). Poměr výstupního a vstupního napětí se nazývá napěťový přenos . $K_{\mathrm{U}}(\omega) = \dfrac{U_2}{U_1}$

Amplitudová frekvenční charakteristika je závislost absolutní hodnoty přenosu K_U na frekvenci ω a fázová frekvenční charakteristika je závislost úhlu $\varphi = \operatorname{arctg} \dfrac{\operatorname{Im}[K_U(\omega)]}{\operatorname{Re}[K_U(\omega)]}$ *na frekvenci ω.*

Vykreslete amplitudovou a fázovou frekvenční charakteristiku filtru – RC integračního členu v rozsahu 1 rad/s až $1 \cdot 10^5$ rad/s. Charakteristiky vykreslete do grafického okna rozděleného na dvě části v **semilogaritmických** souřadnicích **s osou *x* s logaritmickým dělením**.

Grafy

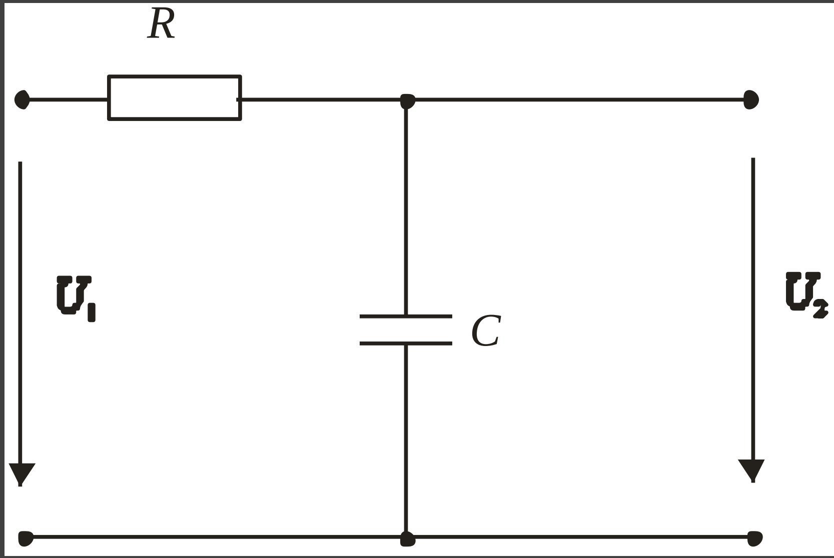
Pokračování příkladu:

RC integrační člen – dolní propusť, dáno *R* = 10 Ω, *C* = 1 mF:

```
w = logspace(0, 5, 100);
R = 10;
C = 1e-3;
Ku = 1./(j*w*C)./(R+1./(j*w*C));
Ku_abs = abs(Ku);
Ku_uhel = angle(Ku)/pi*180;
subplot(2,1,1);
semilogx(w, Ku_abs);
grid
xlabel('\omega (rad/s)');
ylabel('|K_u|');
subplot(2,1,2);
semilogx(w, Ku_uhel);
grid
xlabel('\omega (rad/s)');
ylabel('uhel K_u');
```

Pomocí vztahu pro napěťový dělič:

$$K_U(\omega) = \frac{U_2}{U_1} = \frac{\frac{1}{j\omega C}}{R + \frac{1}{j\omega C}}$$



Grafy

Pokračování příkladu:

RC integrační člen – dolní propusť, dáno $R = 10\ \Omega$, $C = 1\ \text{mF}$:

10^0 10^5

```
w = logspace(0, 5, 100);
R = 10;
C = 1e-3;
Ku = 1./(j*w*C)./(R+1./(j*w*C));
Ku_abs = abs(Ku);
Ku_uhel = angle(Ku)/pi*180;
subplot(2,1,1);
semilogx(w, Ku_abs);
grid
xlabel('\omega (rad/s)');
ylabel('|K_u|');
subplot(2,1,2);
semilogx(w, Ku_uhel);
grid
xlabel('\omega (rad/s)');
ylabel('uhel K_u');
```

operace ./ nutná

velikost K_U

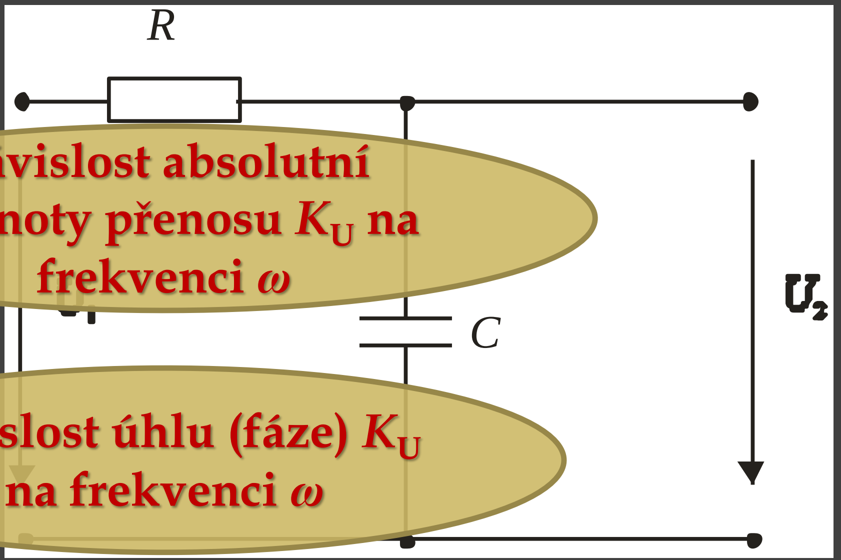
úhel ve stupních

závislost absolutní hodnoty přenosu K_U na frekvenci ω

závislost úhlu (fáze) K_U na frekvenci ω

Pomocí vztahu pro napěťový dělič:

$$K_U(\omega) = \frac{U_2}{U_1} = \frac{\frac{1}{j\omega C}}{R + \frac{1}{j\omega C}}$$



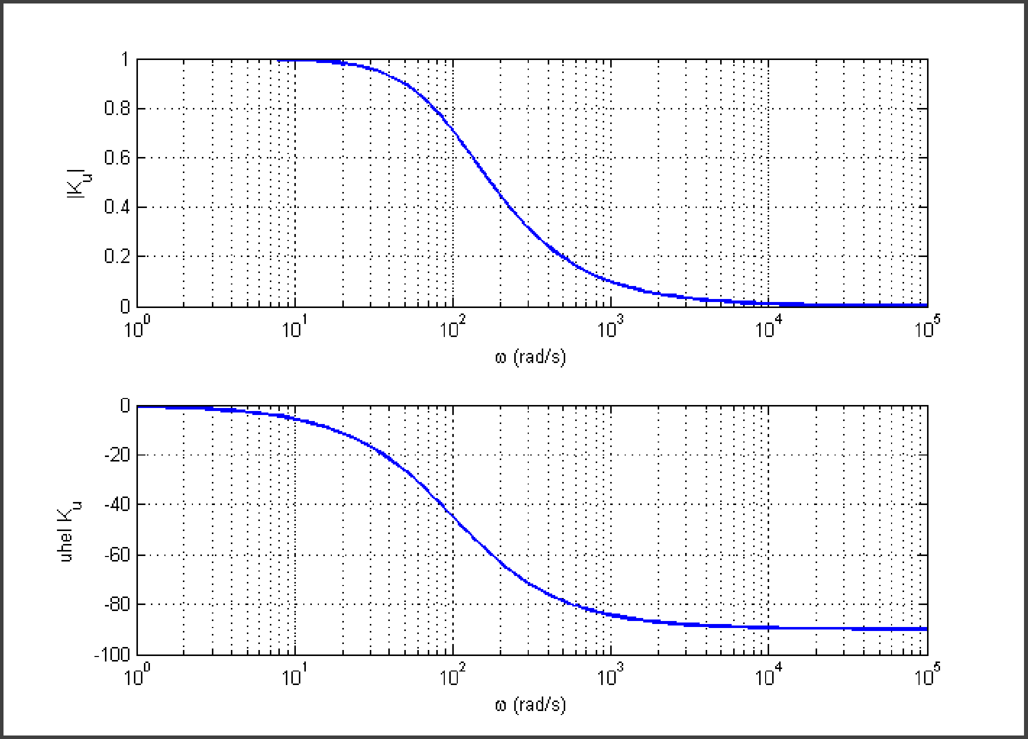
Grafy

Pokračování příkladu:

RC integrační člen – dolní propusť, dáno R = 10 Ω, C = 1 mF:

Pomocí vztahu pro napěťový dělič:

```
w = logspace(0, 5, 100);
R = 10;
C = 1e-3;
Ku = 1./(j*w*C)./(R+1./(j*w*C));
Ku_abs = abs(Ku);
Ku_uhel = angle(Ku)/pi*180;
subplot(2,1,1);
semilogx(w, Ku_abs);
grid
xlabel('\omega (rad/s)');
ylabel('|K_u|');
subplot(2,1,2);
semilogx(w, Ku_uhel);
grid
xlabel('\omega (rad/s)');
ylabel('uhel K_u');
```



Grafy

Řízení **vzhledu textů** v popisech grafů – příkazy LaTeXu

dolní index: `_`   horní index: `^`

- speciální symboly (např. řecká písmena atp.)

Ω `\Omega`	ω `\omega`
α `\alpha`	β `\beta`
Φ `\Phi`	° `\circ`
π `\pi`	∫ `\int`, atp.

viz nápověda MATLABu pod heslem Text Properties (platí tedy pro MATLAB)

- často užívané značky pro tok textu (platí opět pro MATLAB):

`\bf` – tučné písmo (bold)

`\it` – italika, kurzíva

`\sl` – oblique font (jen zřídka k dispozici)

`\rm` – normální font (tj. návrat k výchozímu fontu – ruší příkazy `\it`, `\bf` atd.)

`'FontName','jmeno_fontu'` – nastavení jiného fontu

`'FontSize',velikost_fontu` – změna velikosti fontu

Grafy

Příklad:

Grafické okno je rozděleno na dvě části,

- vlevo jsou vykresleny grafy funkcí

$$y_{\mathbf{sin}} = \mathbf{0{,}5}\ \sin(\alpha^{\mathbf{2}}) \quad \text{a} \quad y_{\mathbf{cos}} = \cos^{\mathbf{2}}(\alpha)$$

v kartézských souřadnicích pro α od **0** do **60°**,
dále je vyznačen kolečkem bod o souřadnicích [**0**, sin(**0**)] a popsán textem.

- vpravo je zobrazen graf funkce

$$r = \sin(\mathbf{2}\zeta)\ \cos(\mathbf{2}\zeta)$$

v polárních souřadnicích pro úhel ζ od **0** do **360°**.

Jsou popsány osy grafu, jsou uvedeny titulky grafů a u prvního grafu legenda.

- celý kód

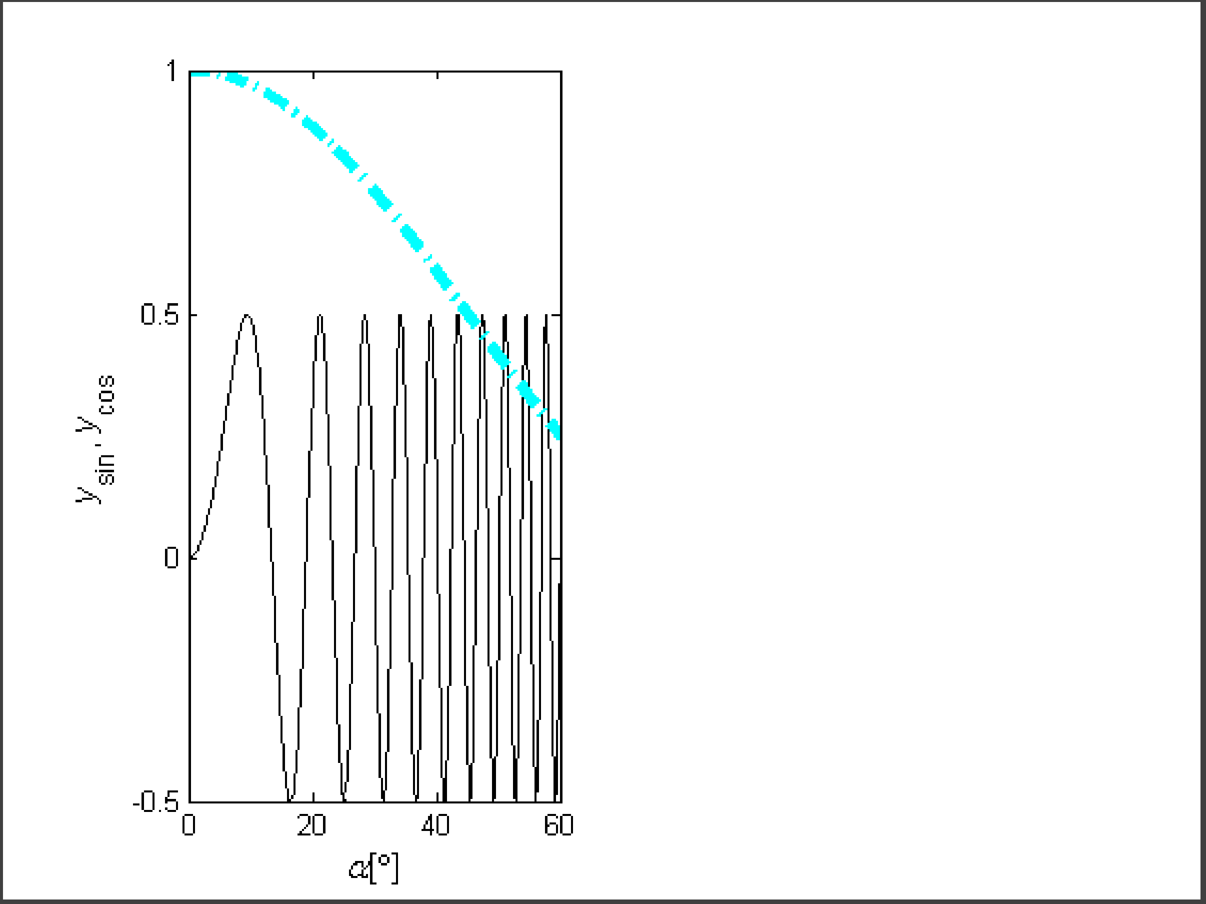
```
subplot(1,2,1)
alpha = 0:0.1:60;
ys = 0.5.*sind(alpha.^2);
yc = cosd(alpha).^2;
plot(alpha,ys,'k','Linewidth',1)
hold on
plot(alpha,yc,'c-.','Linewidth',4)
xlabel('{\it\alpha}[\circ]','FontSize',12)
ylabel('{\ity}_{sin},{\ity}_{cos}','FontSize',12)
title('Graf: {\ity}_1 = 0,5sin(\it\alpha\rm^2) a {\ity}_2 =
cos^2({\it\alpha})',...
'FontName','Times New Roman','FontSize',15)
legend({'y_{sin} = 0,5sin(\alpha^2)',...
'y_{cos} = cos^2\alpha'},'FontAngle','italic',...
'FontName','Arial','FontSize',10,'FontWeight','bold',...
'Location','South');
plot(0,sin(0),'ro')
text(0,0.6,'sin(0) = 0','color','red')
axis([-5,65,-1,1])
hold off
subplot(1,2,2)
xi = 0:.01:2*pi;
polar(xi,sin(2*xi).*cos(2*xi),'m--')
title('Graf: {\itr} = sin(2*{\it\xi}).*cos(2*{\it\xi})',...
'FontName','Times New Roman','FontSize',15)
```

```
subplot(1,2,1)
alpha = 0:0.1:60;
ys = 0.5.*sind(alpha.^2);
yc = cosd(alpha).^2;
plot(alpha,ys,'k','Linewidth',1)
hold on
plot(alpha,yc,'c-.','Linewidth',4)
xlabel('{\it\alpha}[\circ]','FontSize',12)
ylabel('{\ity}_{sin}, {\ity}_{cos}','FontSize',12)
```

- postupně

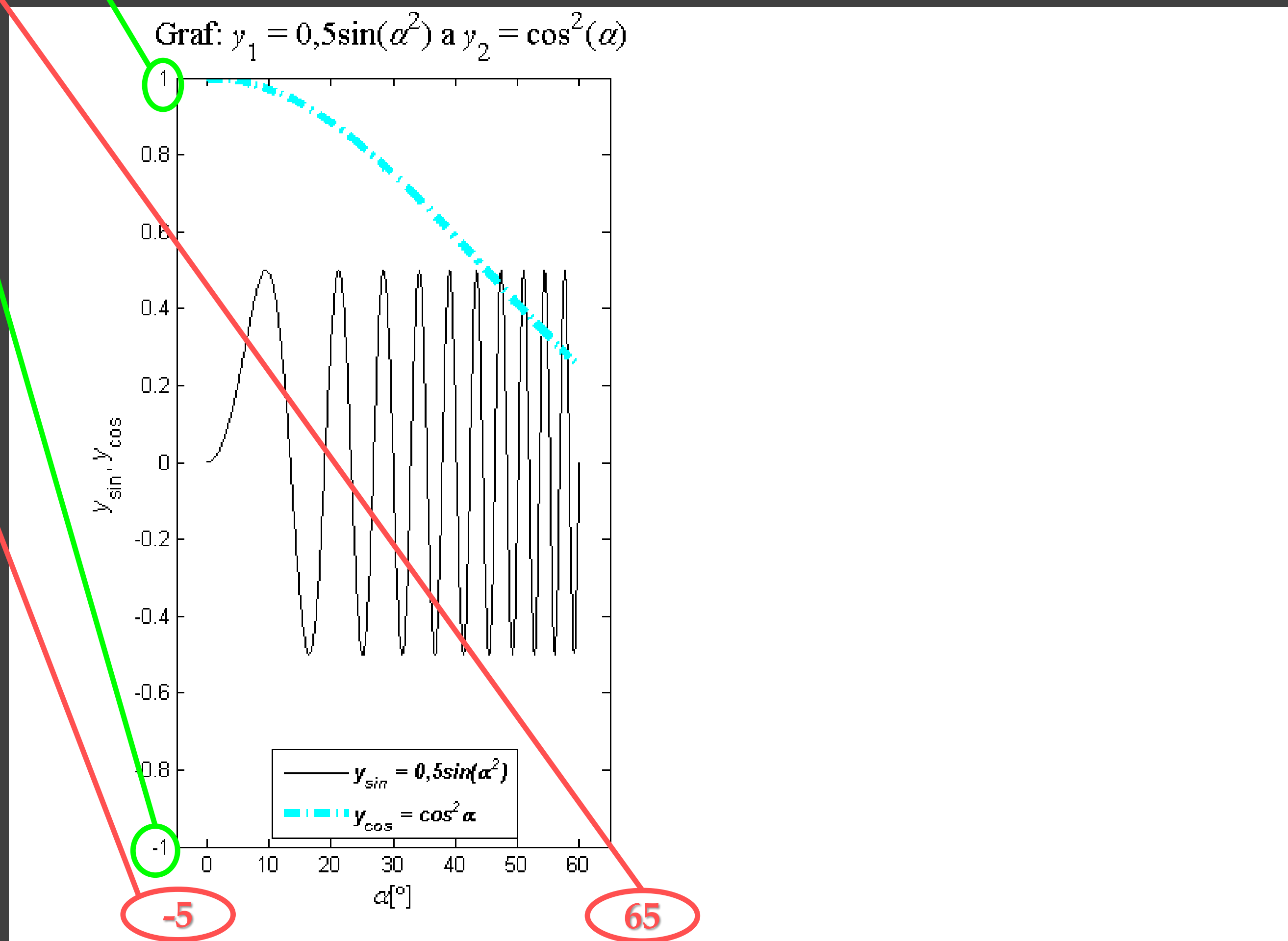
$y_{sin} = 0{,}5\,\sin(\alpha^2)$

$y_{cos} = \cos^2(\alpha)$



- pokračování

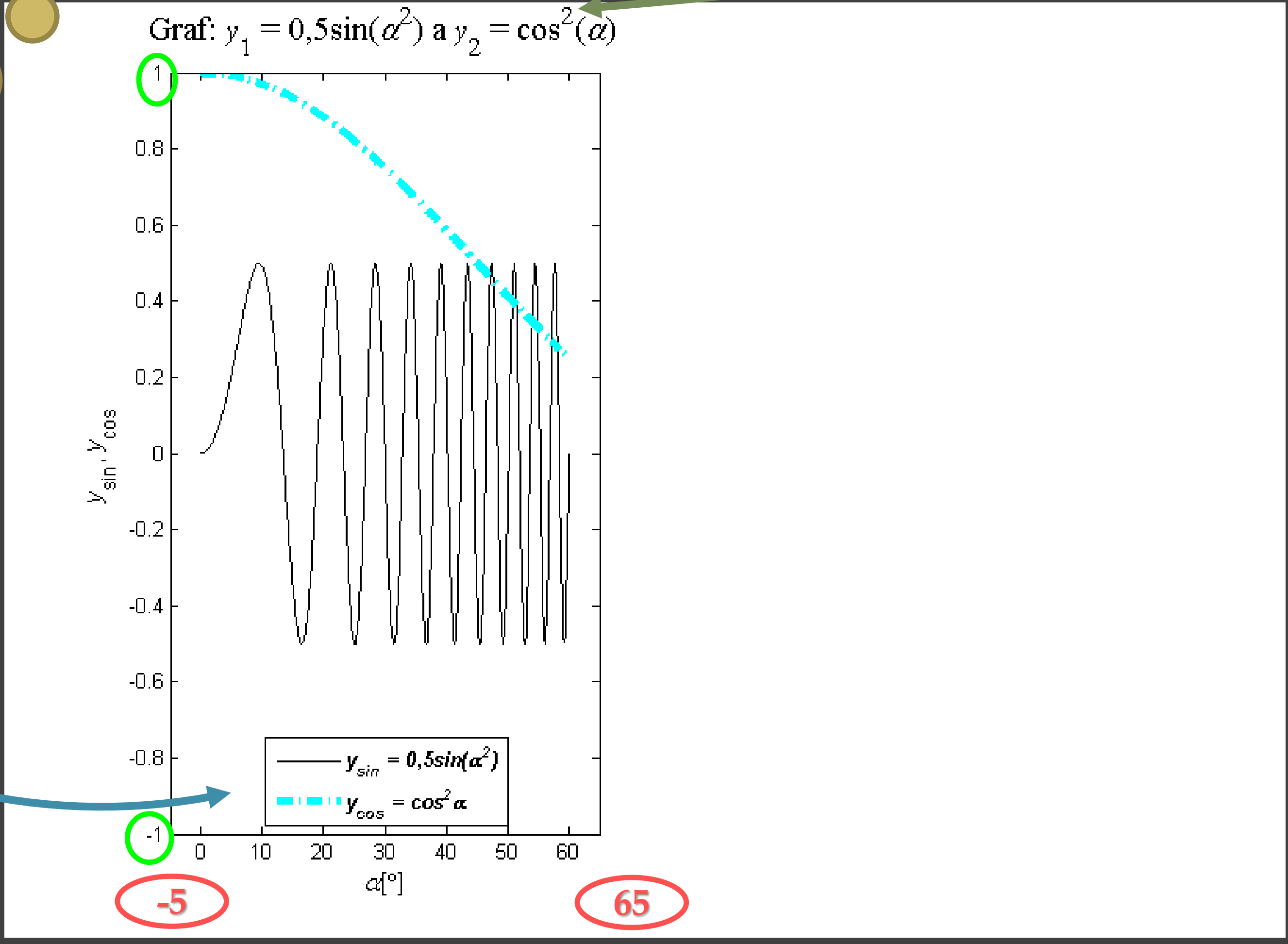
```
axis([-5,65,-1,1])
title('Graf: {\ity}_1 = 0,5sin(\it\alpha\rm^2) a {\ity}_2 =
cos^2({\it\alpha})',...
'FontName','Times New Roman','FontSize',15)
legend({'y_{sin} = 0,5sin(\alpha^2)',...
'y_{cos} = cos^2\alpha'},'FontAngle','italic',...
'FontName','Arial','FontSize',10,'FontWeight','bold',...
'Location','South');
```



```
axis([-5,65,-1,1])
title('Graf: {\ity} 1 = 0,5sin(\it\alpha\rm^2) a {\ity}_2 =
cos^2({\it\alpha})',...
'FontName','Times New Roman','FontSize',15)
legend({'y_{sin} = 0,5sin(\alpha^2)',...
'y_{cos} = cos^2\alpha'},'FontAngle','italic',...
'FontName','Arial','FontSize',10,'FontWeight','bold',...
'Location','South');
```

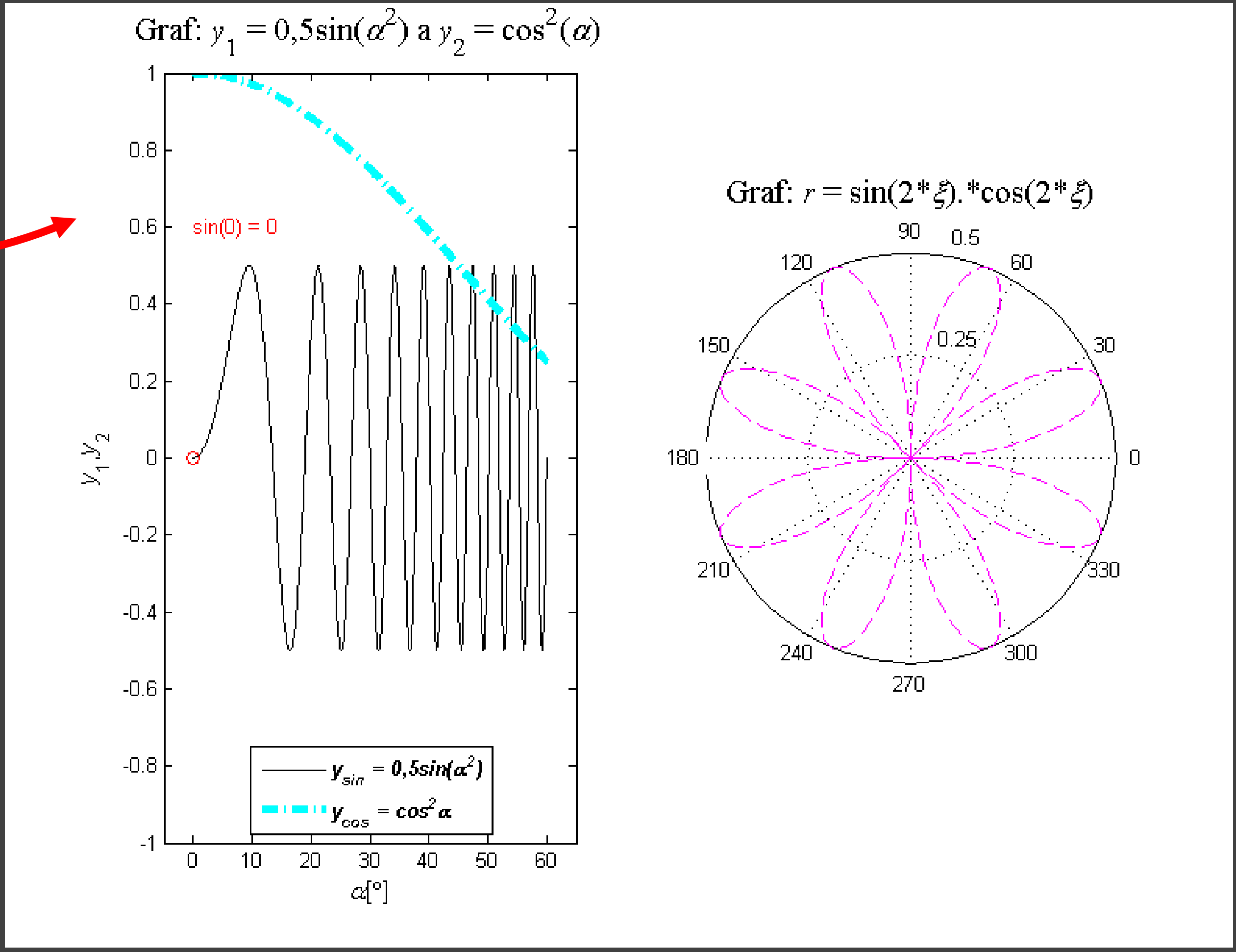
- pokračování

tři tečky – pokračování příkazu na další řádce



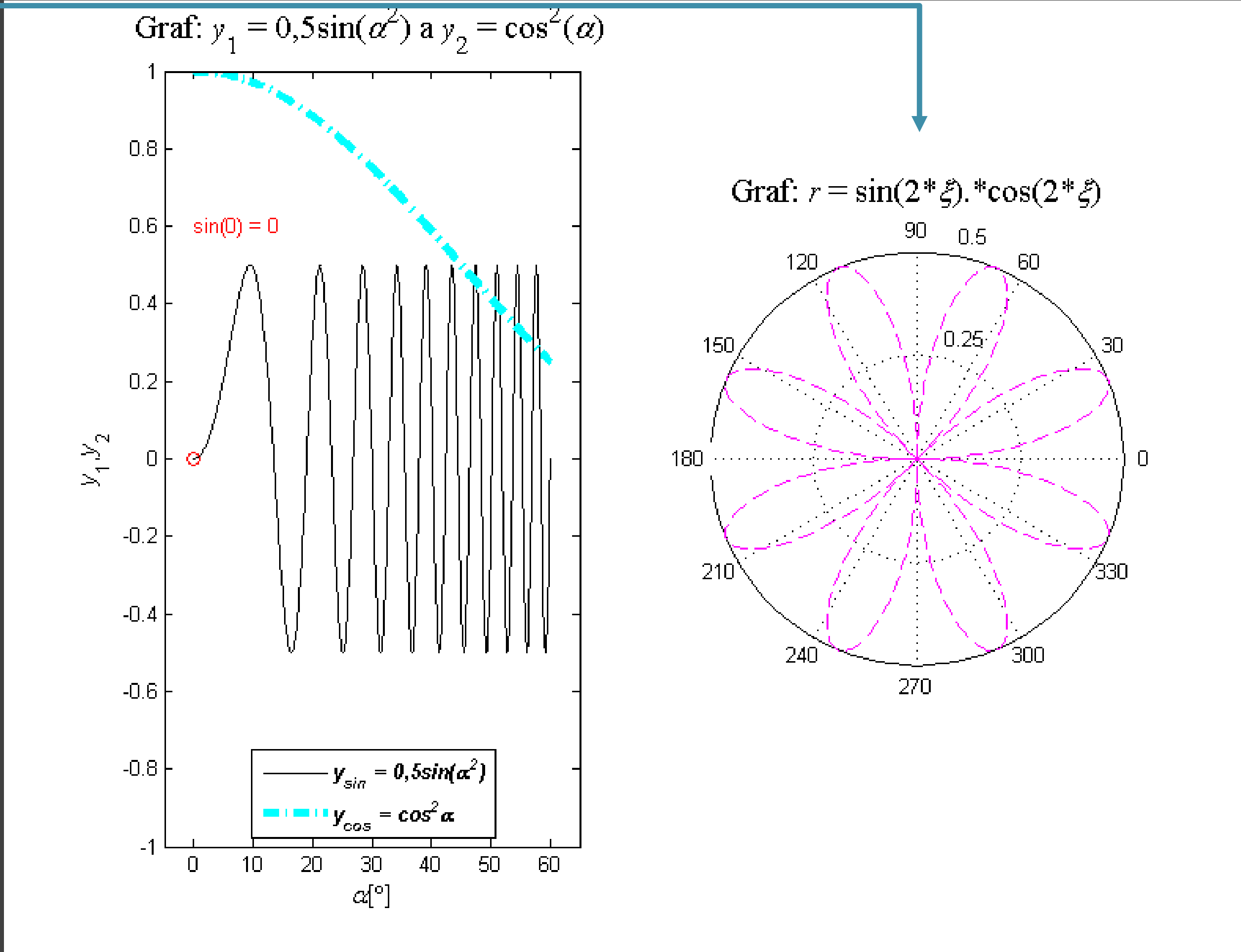
```
plot(0,sin(0),'ro')
text(0,0.6,'sin(0) = 0','color','red')
hold off
subplot(1,2,2)
xi = 0:.01:2*pi;
polar(xi,sin(2*xi).*cos(2*xi),'m--')
title('Graf: {\itr} = sin(2*{\it\xi}).*cos(2*{\it\xi})',...
'FontName','Times New Roman','FontSize',15)
```

- pokračování



```
plot(0,sin(0),'ro')
text(0,0.6,'sin(0) = 0','color','red')
hold off
subplot(1,2,2)
xi = 0:.01:2*pi;
polar(xi,sin(2*xi).*cos(2*xi),'m--')
title('Graf: {\itr} = sin(2*{\it\xi}).*cos(2*{\it\xi})',...
'FontName','Times New Roman','FontSize',15)
```

$r = \sin(2\zeta)\cos(2\zeta)$



- konec

Grafy

- **vyhlazení hran** barevných grafů u plošného grafu **`surf`**.
`shading faceted` – výchozí stav, nevyhlazené, vykreslené hrany
`shading flat` – nevyhlazené, bez vykreslených hran
`shading interp` – vyhlazené s barevnými přechody, bez vykreslených hran

Příklad:
vykreslení průběhu funkce $z = \cos(x^2 + y^2)$
pro x, y z intervalu od **-1** do **1** s krokem, který zvolí uživatel. Krok je volen v mezích od **5** do **50** .
Jedná se o třírozměrný graf (plošný), jsou použity různé typy vyhlazení hran u plošného grafu.

Grafy

Pokračování příkladu:

```
function plosny_graf_stinovani
while(1)
    p = input('Zadej pocet prvku na osach: ');
    if((p >= 5) && (p <= 50))
        break;
    end
end
x = linspace(-1,1,p); % p - počet bodů na ose x
y = linspace(-1,1,p); % p - počet bodů na ose y
[X,Y] = meshgrid(x,y);
Z = cos(X.^2+Y.^2);

subplot(2,2,1);
mesh(x,y,Z)
```

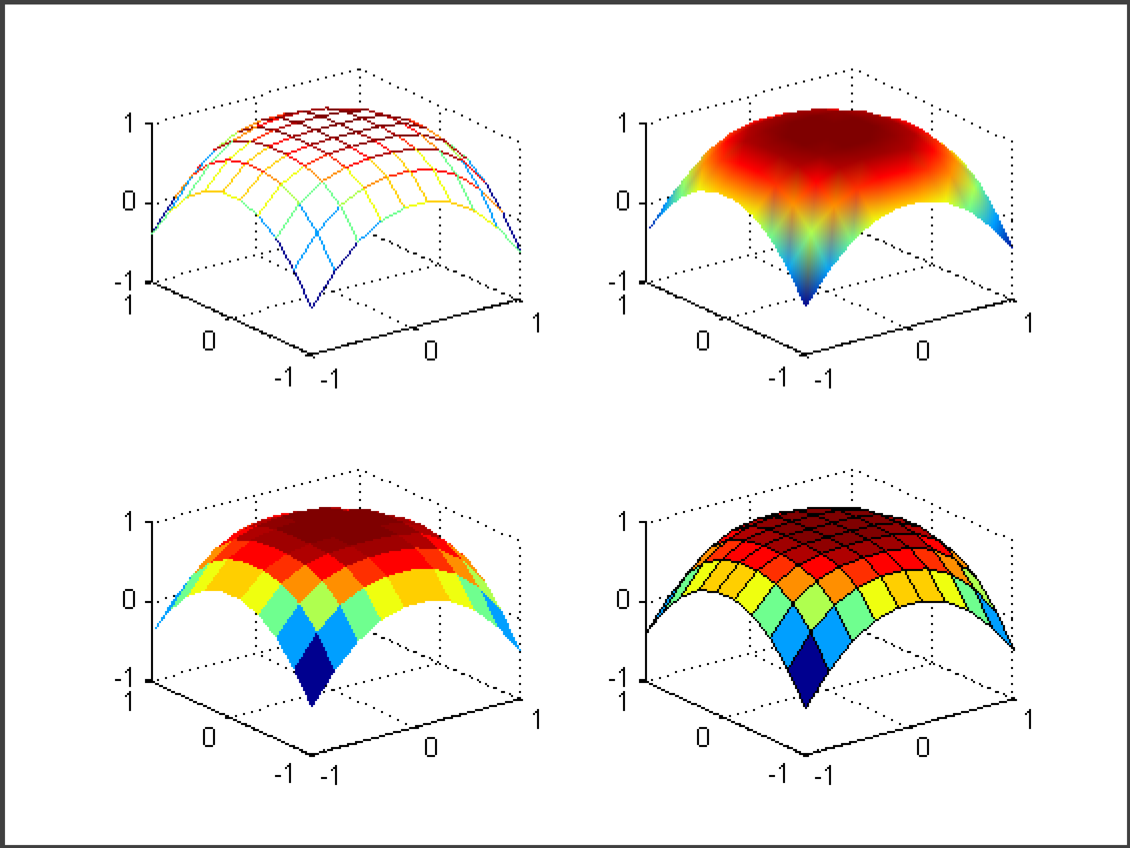
Grafy

Pokračování příkladu:

```
subplot(2,2,2);
surf(x,y,Z)
shading interp

subplot(2,2,3);
surf(x,y,Z)
shading flat

subplot(2,2,4);
surf(x,y,Z)
shading faceted
end % konec funkce
```



Volání funkce:

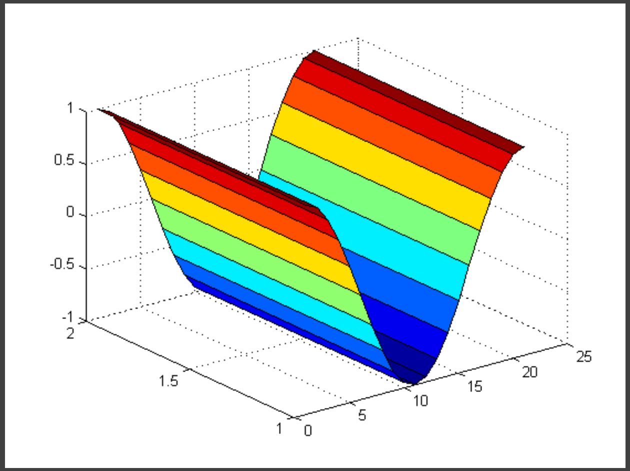
```
plosny_graf_stinovani
Zadej pocet prvku na osach: 10
```

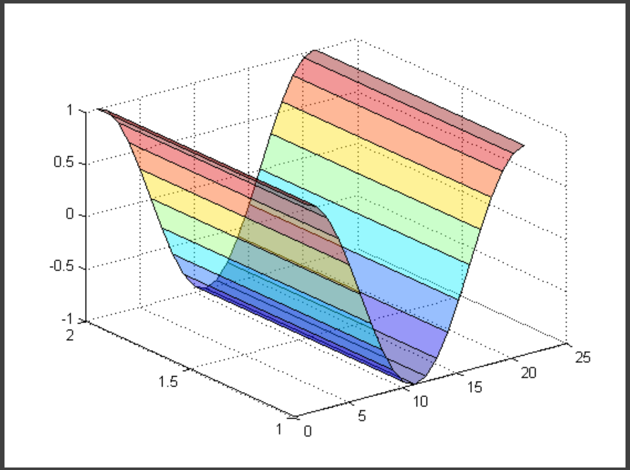
Grafy

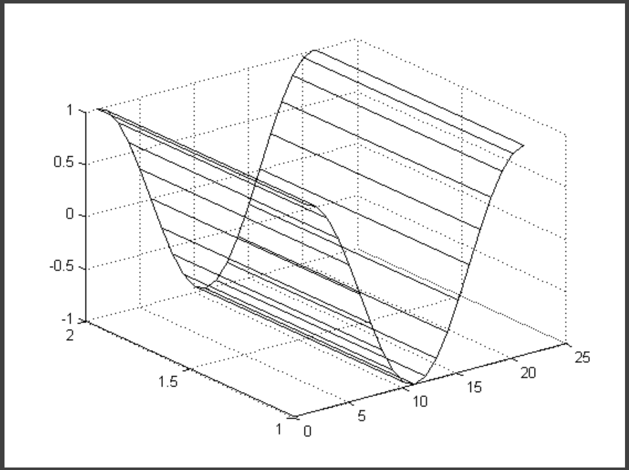
alpha(*n*) – **průhlednost** grafu – **0** je úplně průhledný a **1** je neprůhledný (viz výchozí stav)
(pouze v některých výpočetních systémech)

Příklad:

surf(cylinder) neprůhledný	**surf(cylinder)** **alpha(0.4)** průhlednost 40 %	**surf(cylinder)** **alpha(0)** průhledný







Řešení soustavy rovnic

Příklad: Vytvořte funkci s názvem **vypocet** bez parametrů, která bude řešit soustavu lineárních algebraických rovnic **A***x* = **b**, kde **A** je matice koeficientů soustavy, *x* je sloupcový vektor řešení a **b** je sloupcový vektor pravých stran. V této funkci bude zavolána podle volby uživatele buď funkce **soust_vstup** s dvěma výstupními parametry **A**, **b** pro zadání příslušných dat z klávesnice, a nebo funkce **soust_vstup_soubor** s dvěma výstupními parametry **A**, **b** pro vstup ze souboru. Bude ověřeno, zdali matice A není pázdná. V případě, že ano, funkce bude ukončena. Dále bude ověřeno, zdali hodnost matice **A** je stejná jako počet prvů ve sloupci pravých stran **b** a zdali je matice A dobře podmíněná. Pokud ne, uživatel bude upozorněn, že matice **A** je singulární a bude tázán, chce-li program ukončit. Pokud ano, funkce bude ukončena. V jiném případě bude zavolána funkce **soustava** s dvěma vstupními parametry **A**, **b** a jedním výstupem *x* pro výpočet soustavy lineárních algebraických rovnic.

Řešení soustavy rovnic

Pokračování příkladu: Potom bude zavolána funkce **soustava_vystup** se třemi vstupními parametry **A**, **b**, *x* pro výpis řešení na obrazovku. Pokud si uživatel bude přát, bude proveden výpis do souboru zavoláním funkce **soustava_vystup_soubor** se třemi vstupními parametry **A**, **b**, *x*.

Při řešení předpokládejme, že vektor b má stejný počet prvků jako má matice A řádků (tj. stejný jako je počet rovnic).

Ve funkci **soust_vstup** zadá uživatel z klávesnice počet rovnic. Bude ověřeno, zdali je počet rovnic kladný. Pokud ano, po výzvě programu (např.: „`Zadej prvek matice na pozici 1, 2:` “, kde čísla 1, 2 udávají příslušný řádek a sloupec v matici **A**) zadá uživatel hodnotu prvku, který se uloží do matice **A** na příslušnou pozici. Jednotlivé prvky sloupcového vektoru pravých stran **b** zadá uživatel také z klávesnice po výzvě programu. Pokud je počet rovnic záporný nebo rovný nule, matice **A** i vektor **b** budou prázdné.

Řešení soustavy rovnic

Pokračování příkladu: Funkce **soust_vstup** bude mít 2 výstupní parametry matici **A** a sloupcový vektor **b**.

Ve funkci **soust_vstup_soubor** budou načtena data ze souboru, jehož název zadá uživatel z klávesnice, přičemž první číslo v souboru bude představovat počet rovnic. Předpokládejme, že v souboru budou jen reálná čísla.

Funkce **soustava** s dvěma výstupními parametry **A**, **b** a jedním výstupem ***x*** bude řešit soustavu lineárních algebraických rovnic.

Funkce **soustava_vystup** s třemi vstupními parametry **A**, **b**, ***x*** bez návratové hodnoty vypíše na obrazovku matici koeficientů soustavy **A**, sloupec pravých stran **b** a výsledné hodnoty ***x***, tj. řešení soustavy lineárních algebraických rovnic.

Funkce **soustava_vystup_soubor** s třemi vstupními parametry **A**, **b**, ***x*** bez návratové hodnoty uloží do textového souboru, jehož název zvolí uživatel, matici koeficientů soustavy **A**, sloupec pravých stran **b** a výsledné hodnoty.

```
function vypocet
v = menu('Vstup z klavesnice nebo ze souboru?','klavesnice','soubor');
switch v
    case 1
        [A,b]=soust_vstup;      % volani funkce soust_vstup.m
    case 2
        [A,b]=soust_vstup_soubor; % volani funkce soust_vstup_soubor.m
    otherwise % tato vetev je zbytecna, menu umozni vyber ze 2 moznosti
        return
end % konec switch
if(min(size(A))==0)
    disp('Konec programu')
    return
end % konec if
if((rank(A) ~= length(b)) & (rcond(A) < 1e-12))
    disp('Matice je singularni')
    d = input('Pocitat dal A/N?','s');
    d = lower(d);
        while((d ~= 'a')&(d ~= 'n'))
            d = input('Pocitat dal A/N?','s');
            d = lower(d);
        end % konec while
        if(d == 'n')
            disp('Konec programu')
            return
        end % konec if
end % konec if
```

```
x = soustava(A,b); % volani funkce soustava.m
soustava_vystup(A,b,x)      % volani funkce soustava_vystup.m
s = input('Vystup do souboru A/N ? ','s');
s = upper(s);
if(s == 'A')
    soustava_vystup_soubor(A,b,x) % volani funkce
                                  % soustava_vystup_soubor.m
end % konec if
end % konec funkce vypocet
```

```
function [A,b] = soust_vstup_soubor
t = input('Zadej nazev vstupniho souboru: ','s');
f = fopen(t,'r');
r = fscanf(f,'%d',[1,1]);
A = fscanf(f,'%g',[r,r]);
b = fscanf(f,'%g',[1,r]);
b = b.';
fclose(f);
end
```

```
function [A,b] = soust_vstup
p = input('Zadej pocet rovnic: ');
if(p<=0)
    msgbox('zadna rovnice','Hlaseni','error')
    A=[];b=[];
    return
elseif(p==1)
    msgbox('1 rovnice','Pozor','warn')
else
    msgbox('Ok','Hlaseni','help')
end
A=zeros(p);
for m=1:p % radky
    for n=1:p % sloupce
        fprintf('Zadej prvek matice na pozici %d,%d: ',m,n);
        A(m,n)=input('');
    end
end
b=zeros(p,1);
for m=1:p
   fprintf('Zadej prvek vektoru pravych stran na pozici %d: ',m)
   b(m)=input('');
end
end
```

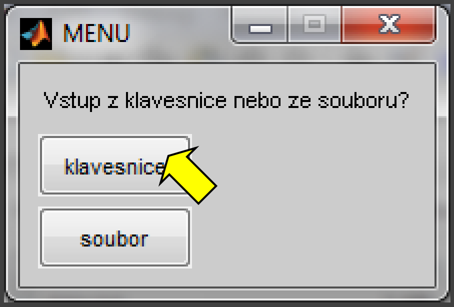
```
function x = soustava(A,b)
x=A\b;
end
```

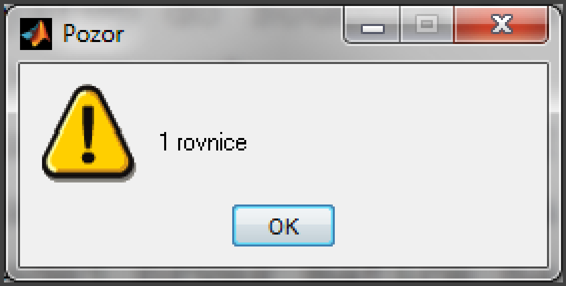
```
function soustava_vystup(A,b,x)
disp('Matice koeficientu soustavy:')
for m=1:length(A)
    for n=1:length(A)
        fprintf('%6.2f %+6.2fi\t',real(A(m,n)),imag(A(m,n)));
    end
    fprintf('\n');
end
disp('Vektor pravych stran')
for m=1:length(b)
    fprintf('%6.2f %+6.2fi\n',real(b(m)),imag(b(m)))
end
disp('Reseni')
for m=1:length(x)
    fprintf('%6.2f %+6.2fi\n',real(x(m)),imag(x(m)))
end
end
```

```
function soustava_vystup_soubor(A,b,x)
n=input('Zadej nazev souboru: ','s');
f=fopen(n,'w');
fprintf(f,'Matice koeficientu soustavy:\n');
for m=1:length(A)
    for n=1:length(A)
        fprintf(f,'%6.2f %+6.2fi\t',real(A(m,n)),imag(A(m,n)));
    end
    fprintf(f,'\n');
end
fprintf(f,'Vektor pravych stran\n');
for m=1:length(b)
    fprintf(f,'%6.2f %+6.2fi\n',real(b(m)),imag(b(m)));
end
fprintf(f,'Reseni\n');
for m=1:length(x)
    fprintf(f,'%6.2f %+6.2fi\n',real(x(m)),imag(x(m)));
end
fclose(f);
msgbox('Vysledek byl ulozen do souboru','Zprava','help')
end
```

Řešení soustavy rovnic

Pokračování příkladu: volání funkce **vypočet**






```
vypocet

Zadej pocet rovnic: 1

Zadej prvek matice na pozici 1,1: 5+i
Zadej prvek vektoru pravych stran na pozici 1: 10+4i

Matice koeficientu soustavy:
   5.00   +1.00i
Vektor pravych stran
  10.00   +4.00i
Reseni
   2.08   +0.38i
Vystup do souboru A/N ? N
```