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

Základy práce se soubory

```
f = fopen('nazev_soubor', 'režim');
```

fopen otevře soubor s názvem **nazev_soubor** pro přístup pro čtení nebo zápis podle zvoleného **režimu**. Řetězec **nazev_soubor** obsahuje název souboru, který má být otevřen. Pokud fopen nemůže otevřít soubor, vrátí hodnotu -1. Jetliže soubor není v aktuální složce, lze zadat i cestu k souboru, např.:

```
f = fopen('C:\\cesta_k_souboru\\soubor', 'režim');
```

kde **režim** může být:

- `'r'` – číst (read)
- `'w'` – zápis, přepis, vytvoření nového souboru (write)
- `'a'` – připsat na konec existujícího (append)
- `'r+'` – čtení nebo zápis
- `'w+'` – čtení nebo zápis, přepis, vytvoření nového souboru
- `'a+'` – čtení nebo zápis, přepis, vytvoření nového souboru a přidávat na konec souboru další data

Základy práce se soubory

Zápis do textového souboru:
`fprintf()` – jako na obrazovku, jediný rozdíl je, že je nutné uvést proměnnou, do které je otevřen soubor, např.

```
muj_soubor = fopen('C:\\cesta\\soubor.txt','w');

fprintf(muj_soubor,'format.sekvence',argumenty);
```

formátovací sekvence a argumenty používáme stejné jako při výpisu na obrazovku

Použitý soubor je třeba nakonec zavřít pomocí **`fclose()`**, jež vrací 0 v případě úspěšného zavření souboru nebo -1 v případě neúspěchu .

```
fclose(muj_soubor);
```

Pozn.: test konce souboru
`feof(muj_soubor)` – pokud je konec souboru, vrátí 1

Základy práce se soubory

'w' – zápis (write)

Zápis do textového souboru:
`fprintf()` – jako na obrazovku, jediný rozdíl je, že je nutné uvést proměnnou, do které je otevřen soubor, např.

```
muj_soubor = fopen('C:\\cesta\\soubor.txt','w');

fprintf(muj_soubor,'format.sekvence',argumenty);
```

formátovací sekvence a argumenty používáme stejné jako při výpisu na obrazovku

Použitý soubor je třeba nakonec zavřít pomocí `fclose()`, jež vrací 0 v případě úspěšného zavření souboru nebo -1 v případě neúspěchu .

```
fclose(muj_soubor);
```

Pozn.: test konce souboru
`feof(muj_soubor)` – pokud je konec souboru, vrátí 1

Základy práce se soubory

Příklad: Funkce, v níž do textového souboru uloženého v aktuální složce bude zapsán **text**, **znak** a **číslo**, které zadá uživatel z klávesnice.

```
function soubor_ulozeni
f = fopen('soubor.txt', 'w');
zadany_text = input('Zadejte text: ', 's');
zadany_znak = input('Zadejte znak: ', 's');
cislo = input('Zadejte cislo: ');
fprintf(f,'Bylo zadan text: %s\n', zadany_text);
fprintf(f,'Bylo zadan znak: %c\n', zadany_znak);
fprintf(f,'Bylo zadano cislo: %g\n', cislo);
fclose(f);
end
```

Základy práce se soubory

Příklad: Funkce, v níž do textového souboru uloženého v aktuální složce bude zapsán **text**, **znak** a **číslo**, které zadá uživatel z klávesnice.

'w' – zápis (write)

```
function soubor_ulozeni
f = fopen('soubor.txt', 'w');
zadany_text = input('Zadejte text: ', 's');
zadany_znak = input('Zadejte znak: ', 's');
cislo = input('Zadejte cislo: ');
fprintf(f,'Bylo zadan text: %s\n', zadany_text);
fprintf(f,'Bylo zadan znak: %c\n', zadany_znak);
fprintf(f,'Bylo zadano cislo: %g\n', cislo);
fclose(f);
end
```

Základy práce se soubory

Pokračování příkladu:
Volání funkce `soubor_ulozeni`:

```
Zadejte text: Ahoj!
Zadejte znak: x
Zadejte cislo: 12345
```

V textovém souboru `soubor.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

Základy práce se soubory

Příklad:

```
function soubor_cisla
p = input('Zadej, kolik chces ulozit cisel? ');
if(p > 0)
    f = fopen('data.txt','w');
    for n = 1:p
        fprintf(f,'%g\n',rand);
    end
    fclose(f);
    fprintf('Cisla byla ulozena\n');
else
    fprintf('Chybne zadani\n');
end
end
```

Základy práce se soubory

Pokračování příkladu:
Volání funkce **soubor_cisla**:

```
Zadej, kolik chces ulozit cisel? -5
Chybne zadani
```

nebo

```
Zadej, kolik chces ulozit cisel? 20
Cisla byla ulozena
```

V textovém souboru **data.txt**, který je uložen v aktuální složce, je pak zapsáno např.:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

fscanf – čtení z textového souboru uloženého na disku, soubor musí nejprve být otevřen pomocí fopen

Např.

Načtení všech čísel ze souboru **data.txt** uloženého na disku C

```
soubor = fopen('C:\\data.txt', 'r');

cisla = fscanf(soubor, '%g');
fclose(soubor);
```

'r' – číst (**r**ead)

Načtení jednoho čísla ze souboru **data.txt** uloženého na disku D (**1 řádek**, **1 sloupec**)

```
soubor = fopen('D:\\data.txt', 'r');
cisla = fscanf(soubor, '%g', [1,1]);
fclose(soubor);
```

Velikost je nepovinná, ale pomocí ní lze omezit počet prvků, které je možné číst ze souboru.

Základy práce se soubory

Velikost je tedy nepovinná, ale pomocí ní lze omezit počet prvků, které je možné číst ze souboru, je-li zadán rozměr matice, vyplní se matice o dané velikosti, např.

Čtení z textového souboru `soubor.txt` uloženého v aktuálním adresáři:

```
muj_soubor = fopen('soubor.txt', 'r');

cisla = fscanf(muj_soubor, '%g', [2,inf]);
```

fscanf má přečíst reálná čísla, organizovaná ve **2 řádcích** a **neznámém počtu sloupců** (chci **číst** až **do konce souboru**), na pozici řádek **nesmí** být `inf`.

Použitý soubor je třeba nakonec zavřít:

```
fclose(muj_soubor)
```

Základy práce se soubory

Příklad: čtení z textového souboru `soubor.txt`

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);

disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345
```

V textovém souboru `soubor.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

Základy práce se soubory

Příklad: čtení z textového souboru `soubor.txt`

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);
```

`'r'` – číst (read)

`'%c'` – znak

```
disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345
```

V textovém souboru `soubor.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

Základy práce se soubory

Příklad: čtení z textového souboru **soubor.txt**

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);
```

'r' – číst (read)

'%c' – znak

V textovém souboru **soubor.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

```
disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345

f2=fopen('soubor.txt','r');
s = fscanf(f2,'%s');
fclose(f2);

disp(s)
Bylozadantext:Ahoj!Bylozadanoznak:xBylozadanocislo:12345
```

Základy práce se soubory

Příklad: čtení z textového souboru **soubor.txt**

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);

disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345

f2=fopen('soubor.txt','r');
s = fscanf(f2,'%s');
fclose(f2);

disp(s)
Bylozadantext:Ahoj!Bylozadanoznak:xBylozadanocislo:12345
```

'r' – číst (read)

'%c' – znak

V textovém souboru **soubor.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

'r' – číst (read)

'%s' – řetězec

Bez mezer, bez konců řádek

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g');
fclose(fr);
```

```
disp(c)
  0.1067
  0.9619
  0.0046
  0.7749
  0.8173
  0.8687
  0.0844
  0.3998
  0.2599
  0.8001
  0.4314
  0.9106
  0.1818
  0.2638
  0.1455
  0.1361
  0.8693
  0.5797
  0.5499
  0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g');
fclose(fr);
```

`'r'` – číst (read)

`'%g'` – destinné číslo
příp. i v exp. tvaru, bez nevýznamných nul

```
disp(c)
```

```
0.1067
0.9619
0.0046
0.7749
0.8173
0.8687
0.0844
0.3998
0.2599
0.8001
0.4314
0.9106
0.1818
0.2638
0.1455
0.1361
0.8693
0.5797
0.5499
0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru **data.txt**

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[4,Inf]);
fclose(fr);
```

V textovém souboru **data.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

```
disp(c)
    0.1067    0.8173    0.2599    0.1818    0.8693
    0.9619    0.8687    0.8001    0.2638    0.5797
    0.0046    0.0844    0.4314    0.1455    0.5499
    0.7749    0.3998    0.9106    0.1361    0.1450
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[4,Inf]);
fclose(fr);
```


```
disp(c)
```

```
→ 0.1067   0.8173   0.2599   0.1818   0.8693
→ 0.9619   0.8687   0.8001   0.2638   0.5797
→ 0.0046   0.0844   0.4314   0.1455   0.5499
→ 0.7749   0.3998   0.9106   0.1361   0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[10,Inf]);
fclose(fr);
```

```
disp(c)
```

```
    0.1067    0.4314
    0.9619    0.9106
    0.0046    0.1818
    0.7749    0.2638
    0.8173    0.1455
    0.8687    0.1361
    0.0844    0.8693
    0.3998    0.5797
    0.2599    0.5499
    0.8001    0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[10,Inf]);
fclose(fr);
```

```
disp(c)
```

```
    0.1067    0.4314
    0.9619    0.9106
    0.0046    0.1818
    0.7749    0.2638
    0.8173    0.1455
    0.8687    0.1361
    0.0844    0.8693
    0.3998    0.5797
    0.2599    0.5499
    0.8001    0.1450
```

10 řádků a neznámý počet sloupců (chci číst až do konce souboru)

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru **data.txt**

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[6,Inf]);
fclose(fr);
```

```
disp(c)
```

```
    0.1067    0.0844    0.1818    0.5499
    0.9619    0.3998    0.2638    0.1450
    0.0046    0.2599    0.1455         0
    0.7749    0.8001    0.1361         0
    0.8173    0.4314    0.8693         0
    0.8687    0.9106    0.5797         0
```

V textovém souboru **data.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[6,Inf]);
fclose(fr);
```

6 řádků a neznámý počet sloupců (chci číst až do konce souboru)

```
disp(c)
```

```
    0.1067    0.0844    0.1818    0.5499
    0.9619    0.3998    0.2638    0.1450
    0.0046    0.2599    0.1455         0
    0.7749    0.8001    0.1361         0
    0.8173    0.4314    0.8693         0
    0.8687    0.9106    0.5797         0
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[6,Inf]);
fclose(fr);
```

6 řádků a neznámý počet sloupců (chci číst až do konce souboru)

```
disp(c)
```

```
→ 0.1067    0.0844    0.1818    0.5499
→ 0.9619    0.3998    0.2638    0.1450
→ 0.0046    0.2599    0.1455         0
→ 0.7749    0.8001    0.1361         0
→ 0.8173    0.4314    0.8693         0
→ 0.8687    0.9106    0.5797         0
```

doplněno nulami

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru **data.txt**

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[3,2]);
fclose(fr);
```

```
disp(c)
    0.1067    0.7749
    0.9619    0.8173
    0.0046    0.8687
```

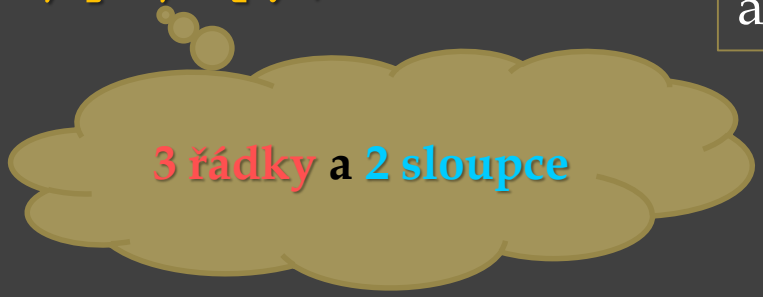
V textovém souboru **data.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[3,2]);
fclose(fr);
```




```
disp(c)
```

```
   0.1067    0.7749
   0.9619    0.8173
   0.0046    0.8687
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Grafy

Výpočetní systémy umožňují vykreslit více grafů do jednoho grafického okna:

- vedle sebe, pod sebe - rozdělení grafického okna (`subplot`)
- přes sebe – např. `plot(x1,y1,x2,y2,…,xn,yn)`
- přes sebe – `hold`

  `hold on` – přidrží aktuální graf v grafickém okně (zmrazí aktuální grafickou obrazovku) a všechny následující grafické výstupy do něho přikresluje, lze tedy nakreslit více grafů do jednoho grafického okna postupně
  `hold off` – vypnutí, konec možnosti kreslit více grafů do jednoho grafického okna (tj. opět mazání předchozích grafů)

`text(x,y,'nejaky text')` – umístí text na souřadnice x, y

`gtext('nejaky g-text')` – umístí text na souřadnice tam, kam je kliknuto myší (platí v MATLABu)

Dvojdimenzionální grafy

Soustavy souřadnic

- kartézská soustava souřadnic je taková soustava souřadnic, u které jsou souřadné osy vzájemně kolmé a protínají se v jednom bodě - počátku soustavy souřadnic.

  `plot(x,y),semilogx(x,y),semilogy(x,y), …`

- polární soustava souřadnic je taková soustava souřadnic v rovině, u které jedna souřadnice (označovaná *r*) udává vzdálenost bodu od počátku souřadnic, druhá souřadnice (označovaná *φ*) udává úhel spojnice tělesa a počátku od zvolené osy ležící v rovině.

  `polar(φ,r)`

Dvojdimenzionální grafy

`plot(x,y,S)` – rovinný graf s **lineárním** dělením na osách x, y (tj. „klasický“ graf x, y, v **kartézských souřadnicích**), jsou-li x a y vektory o stejné délce, pak **`plot(x,y)`** vykreslí x-y graf,

`semilogx(x,y,S)` – graf jako **`plot`**, ale s **logaritmickou** stupnicí (o základu 10) **na ose** x,

`semilogy(x,y,S)` – graf jako **`plot`**, ale s **logaritmickou** stupnicí (o základu 10) **na ose** y,

`loglog(x,y,S)` – graf jako **`plot`**, ale s **logaritmickou** stupnicí (o základu 10) **na obou osách** x, y.

`polar(uhel,velikost,S)` – graf v **polárních souřadnicích**

`compass(u,v,S)` – vektor se složkami u, v ve formě **šipky vycházející z počátku**

Parametr **`S`** je **řetězec**, který musí být v apostrofech (některé výpočetní systémy připouštějí i uvozovky) a specifikuje barvu, způsob vykreslení a styl křivky nebo typ značky bodu.

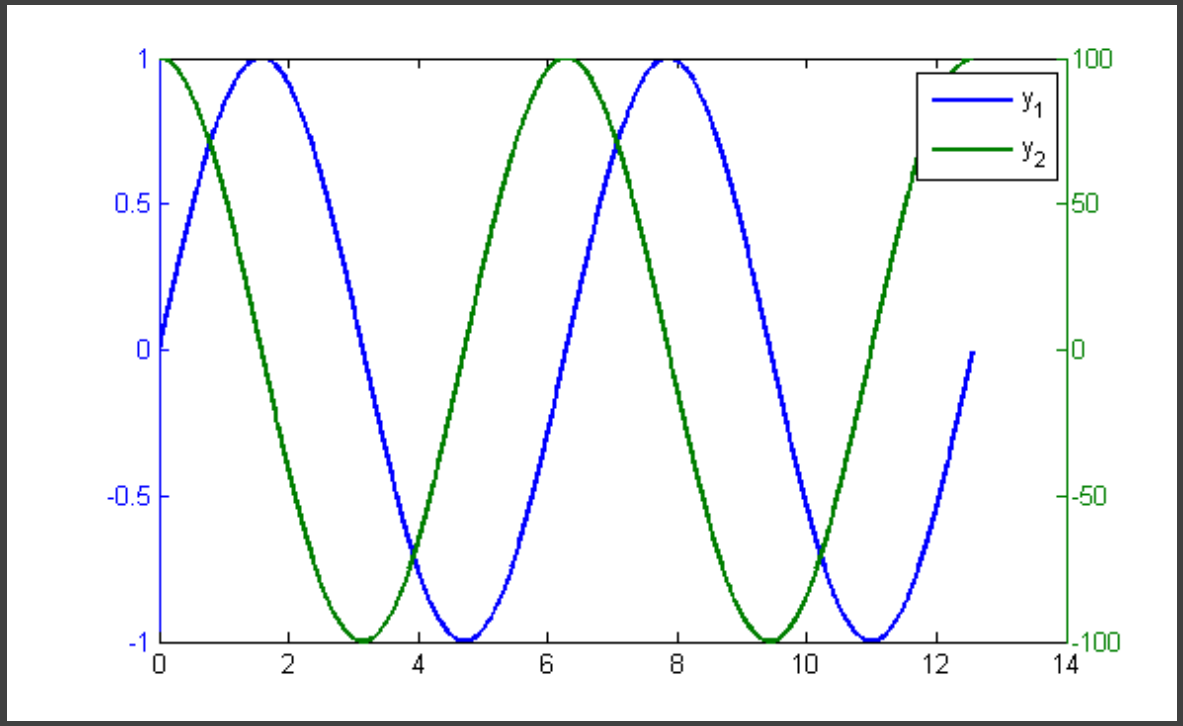
Pozn. barvy, typy značek bodů, styly čar – viz předchozí přednášky a viz **`help plot`**

Dvojdimenzionální grafy

plotyy – graf s dvěma různými osami y pro dvě různé křivky, např. **plotyy** (x_1, y_1, x_2, y_2) – osa pro graf y_1 vlevo, osa pro graf y_2 vpravo

Příklad:
Vykreslení grafů funkcí $y_1 = \sin(x)$, $y_2 = 100\cos(x)$

```
x=[0:0.01:4*pi];
y1=sin(x);
y2=100*cos(x);
plotyy(x,y1,x,y2)
legend('y_1','y_2')
```

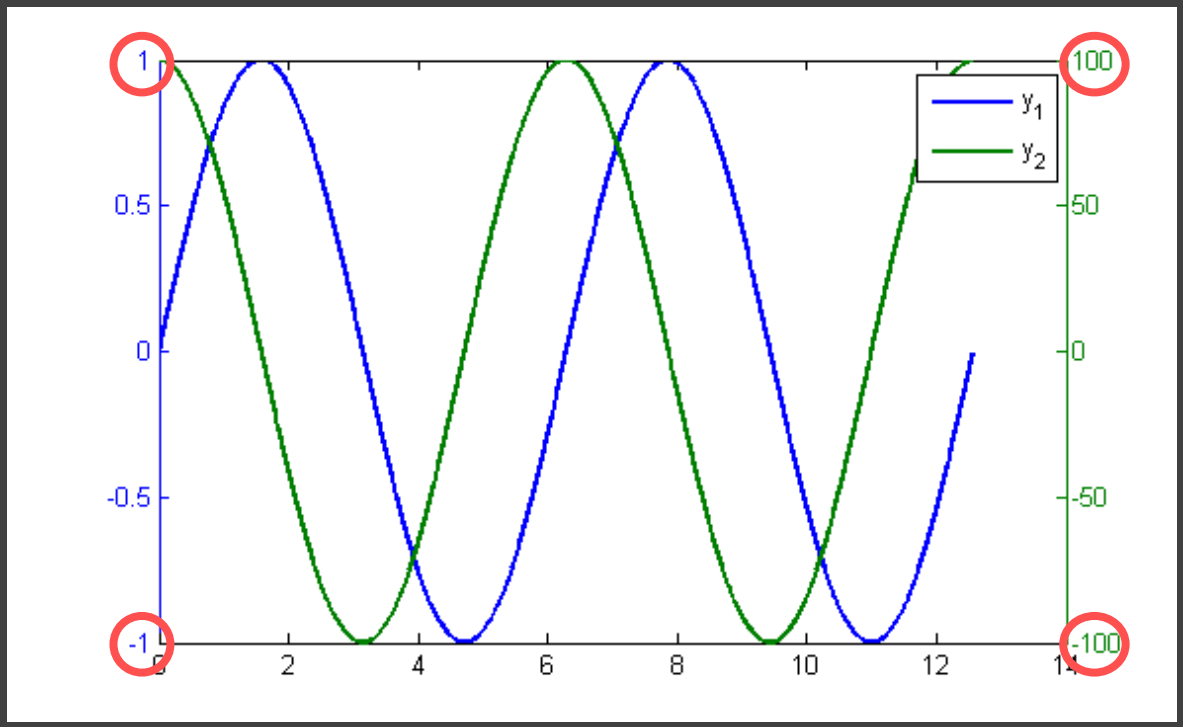


Dvojdimenzionální grafy

plotyy – graf s dvěma různými osami y pro dvě různé křivky, např. **plotyy** (x_1, y_1, x_2, y_2) – osa pro graf y_1 vlevo, osa pro graf y_2 vpravo

Příklad:
Vykreslení grafů funkcí $y_1 = \sin(x)$, $y_2 = 100\cos(x)$

```
x=[0:0.01:4*pi];
y1=sin(x);
y2=100*cos(x);
plotyy(x,y1,x,y2)
legend('y_1','y_2')
```



Dvojdimenzionální grafy

`compass(u,v,S)` – vektor se složkami *u*, *v* ve formě **šipky vycházející z počátku**

Příklady:

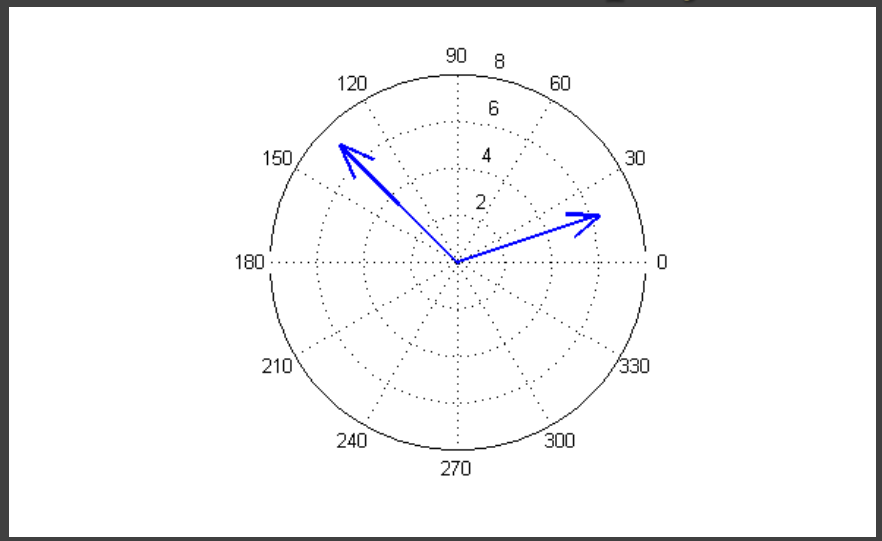
Zobrazení vektorů

a = [-5,5], *b* = [6,2]

```
compass([-5,6],[5,2])
```

nebo jiným způsobem:

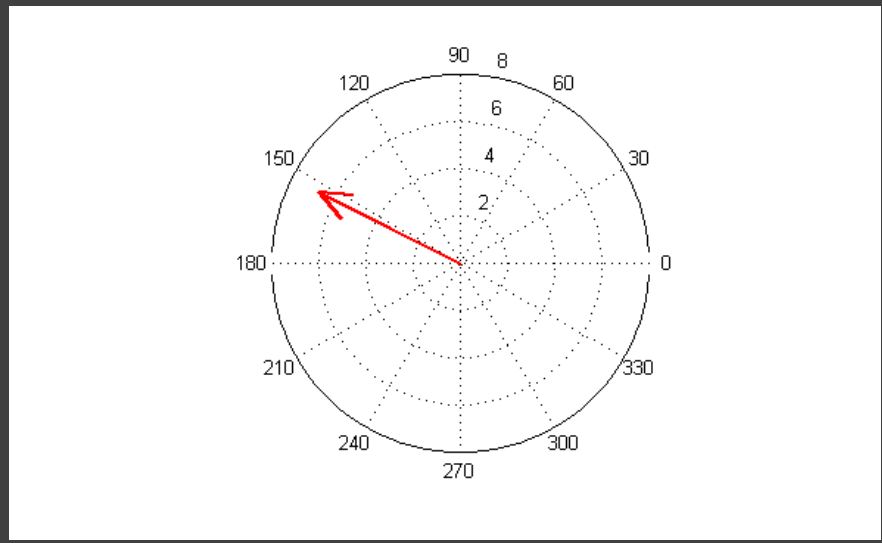
```
a = [-5,5]; b = [6,2];
compass([a(1),b(1)],[a(2),b(2)])
```



Zobrazení komplexních čísel – viz předchozí přednášky

Např.

```
compass(-6+3i,'r')
```



Dvojdimenzionální grafy

`polar(uhel,velikost,S)` – graf v **polárních souřadnicích**

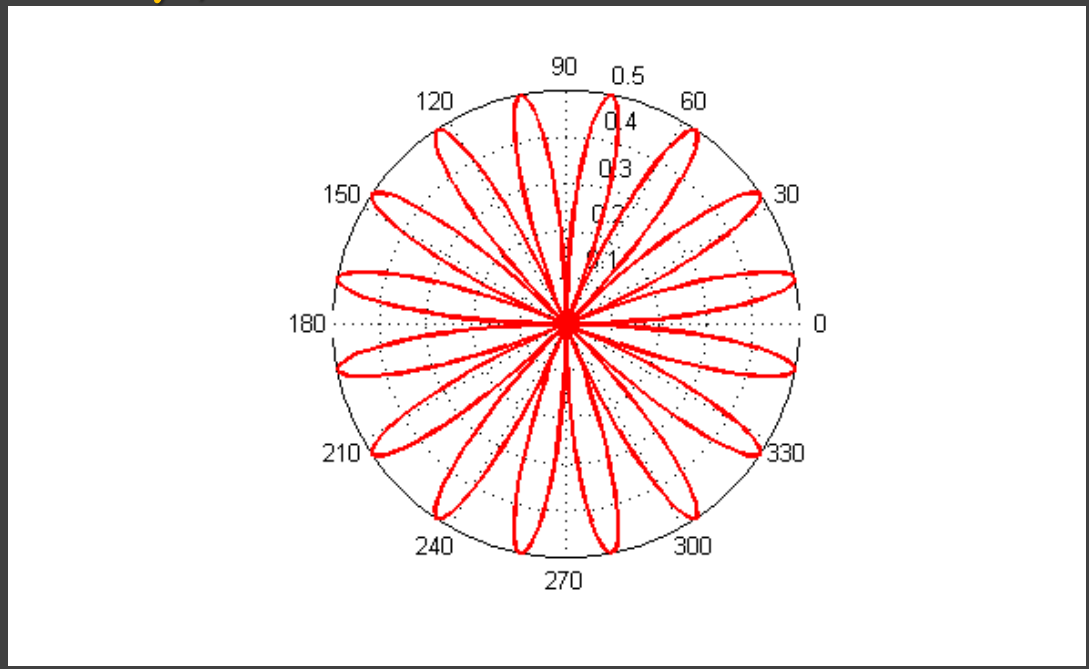
Příklad:
graf v polárních souřadnicích
$r = \sin(4\varphi)\cos(4\varphi)$,
kde φ je z intervalu od 0 do 2π.

V nových verzích MATLABu se setkáme i s příkazem
`polarplot(uhel,velikost)`

```
fi = 0:0.01:2*pi;
y = sin(4.*fi).*cos(4.*fi);
polar(fi, y, 'r')

% v nových verzích
MATLABu lze též

polarplot(fi, y, 'r')
```

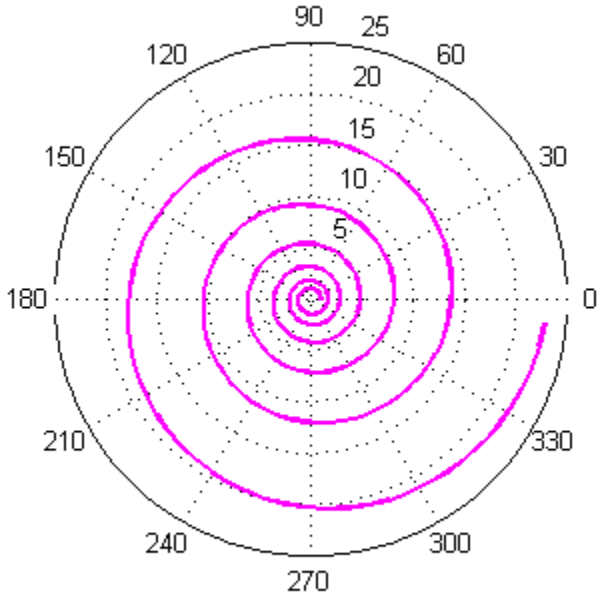


Dvojdimenzionální grafy

`polar(uhel,velikost,S)` – graf v **polárních souřadnicích**

Příklad:
graf logaritmické spirály v polárních souřadnicích
Logaritmická spirála se vyskytuje např. v kresbě ulit plžů, je dána rovnicí $r = a\, e^{m\varphi}$
Křivka vykreslena pro $a = 1$, $m = 1/12$, φ z intervalu od **0** do **12** π.

```
phi = 0:0.1:12*pi;
r = exp((1/12)*phi)
polar(phi,r,'m')
```



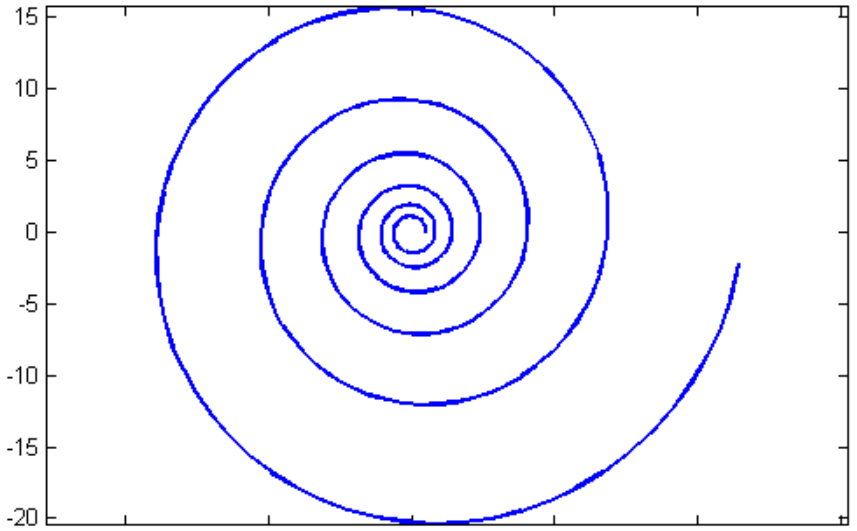
Dvojdimenzionální grafy

Pozn.: užitečná funkce **`pol2cart`** – převod souřadnic polárních na kartézské

```
[x,y] = pol2cart(uhel, vzdálenost_od_počátku);
```

Pokračování příkladu:
graf logaritmické spirály z předchozího příkladu $r = a\, e^{m\varphi}$, kde $a = 1$, $m = 1/12$, φ je z intervalu od **0** do **12** π, je převeden do kartézských souřadnic a vykreslen příkazem **`plot`**.

```
phi = 0:0.1:12*pi;
r = exp((1/12)*phi);
[x,y]=pol2cart(phi,r);
plot(x,y)
```



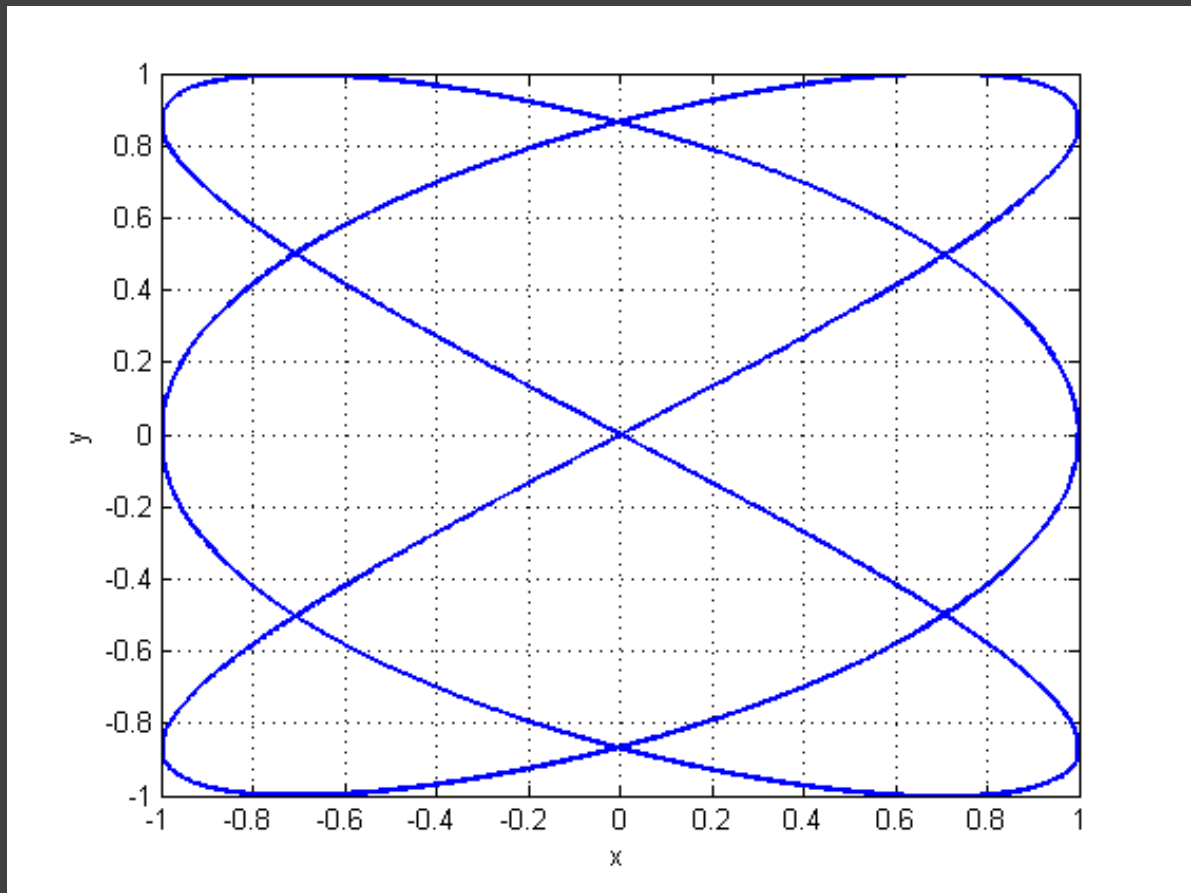
Naopak:
`cart2pol` – převod souřadnic kartézských na polární

Dvojdimenzionální grafy

Příklad:

- graf parametrizované křivky (popsán parametrickými rovnicemi)
- *t* je parametr, na kterém závisí *x*, *y*

```
t = 0:.001:2*pi;
x = cos(3.*t);
y = sin(2.*t);
plot(x,y)
grid
 % grid přikreslí do
 % grafu síť (mřížku)
xlabel('x')
ylabel('y')
 % popis os
```

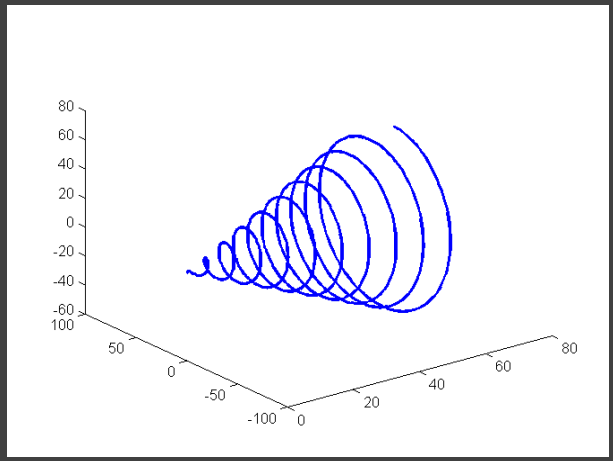


Trojdimenzionální grafy

- křivkové grafy

body, úsečky, křivky v prostoru

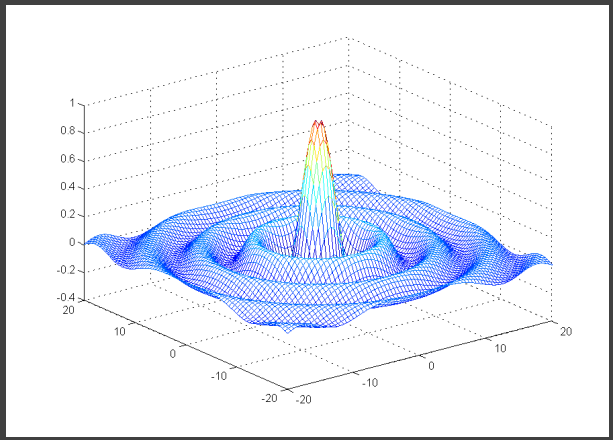
např. `plot3(x,y,z)` – vyjadřuje obvykle závislost y a z na x



- plošné grafy

síťové(drátové), povrchové grafy

např. `mesh(x,y,z)`, `surf(x,y,z)` atp. – vyjadřují obvykle závislost z na x a y



Trojdimenzionální grafy

`plot3(x,y,z,S)` – 3D graf křivkový

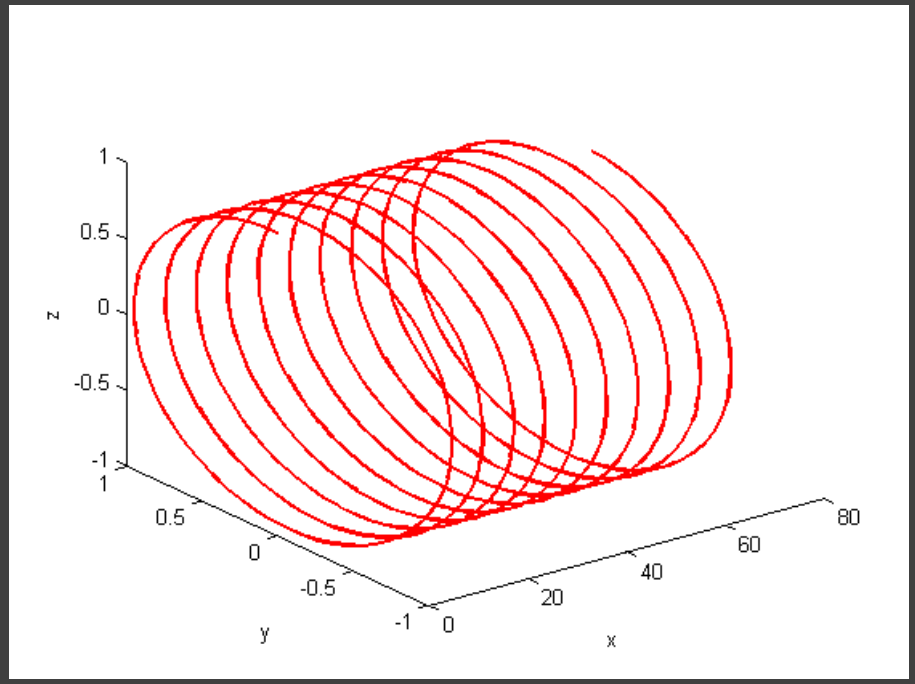
– vykreslí křivku v prostoru procházející body o souřadnicích x, y, z, které jsou prvky stejně dlouhých vektorů **`x`**, **`y`**, **`z`**,

– vyjadřuje obvykle závislost y a z na x,

– lze použít podobné příkazy a parametry jako ve 2D (**`plot`**), parametr **`S`** je **řetězec**, který specifikuje barvu, způsob vykreslení a styl křivky nebo typ značky bodu.

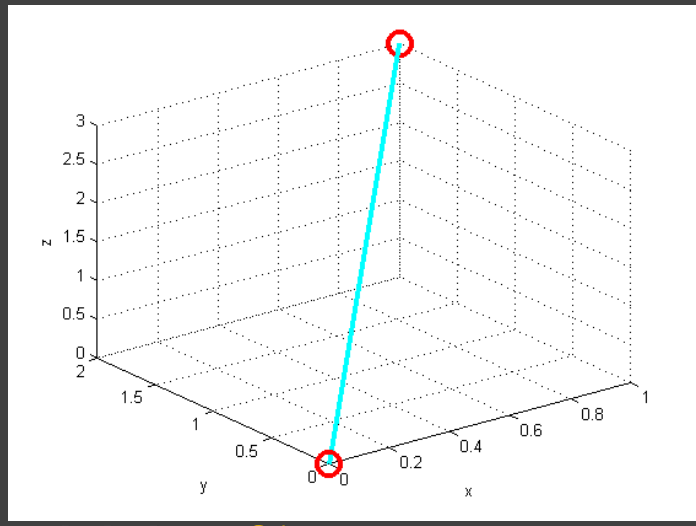
Příklad: graf křivky popsané parametrickými rovnicemi:
$x = t$, $y = \sin(t)$, $z = \cos(t)$,
pro t od 0 do 20π.

```
t = [0:0.1:20*pi]
plot3(t,sin(t),cos(t),'r')
xlabel('x')
ylabel('y')
zlabel('z')
```



Trojdimenzionální grafy

Příklad: vykreslení úsečky v prostoru z bodu [0, 0, 0] do bodu [1, 2, 3]

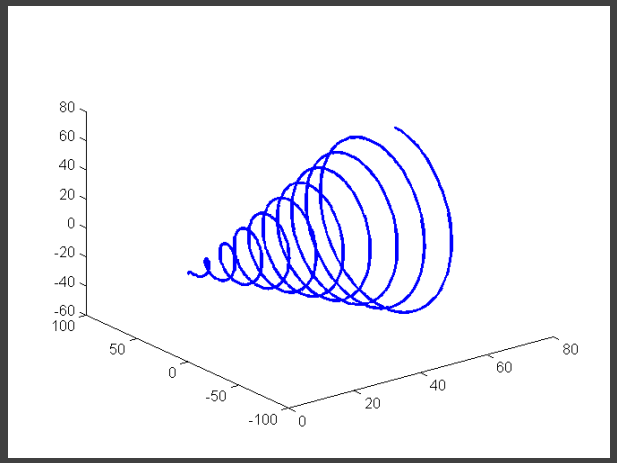



```
x=[0,1]; % x-ové souřadnice bodů
y=[0,2]; % y-ové souřadnice bodů
z=[0,3]; % z-ové souřadnice bodů
plot3(x,y,z,'c','LineWidth',3)
% zvolena barva - modrozelená (cyan)
% zvolena tloušťka čáry
                        % 'LineWidth',3
hold on
plot3(0,0,0,'ro','MarkerSize',15,'Linewidth',3)
plot3(1,2,3,'ro','MarkerSize',15,'LineWidth',3)
hold off
 % 'ro' - body jsou zobrazeny červenými kolečky
 % 'MarkerSize',15 - velikost značky
 % 'LineWidth',3 - tloušťka čáry kolečka (značky)
xlabel('x') % popis osy x
ylabel('y') % popis osy y
zlabel('z') % popis osy z
grid % přidá do grafu síť, mřížku
```

Trojdimenzionální grafy

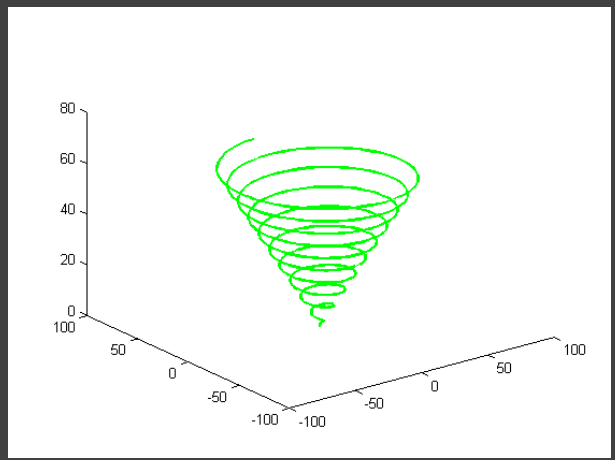
Příklady:
Graf křivky popsané parametrickými rovnicemi:
$x = t$, $y = t \sin(t)$, $z = t \cos(t)$,
pro t od **0** do 20π.

```
t = [0:0.1:20*pi];
plot3(t,t.*sin(t),t.*cos(t))
```



Graf křivky popsané parametrickými rovnicemi:
$x = t \sin(t)$, $y = t \cos(t)$, $z = t$,
pro t od **0** do 20π.

```
t = [0:0.1:20*pi];
plot3(t.*sin(t),t.*cos(t),t,'g')
```



Trojdimenzionální grafy

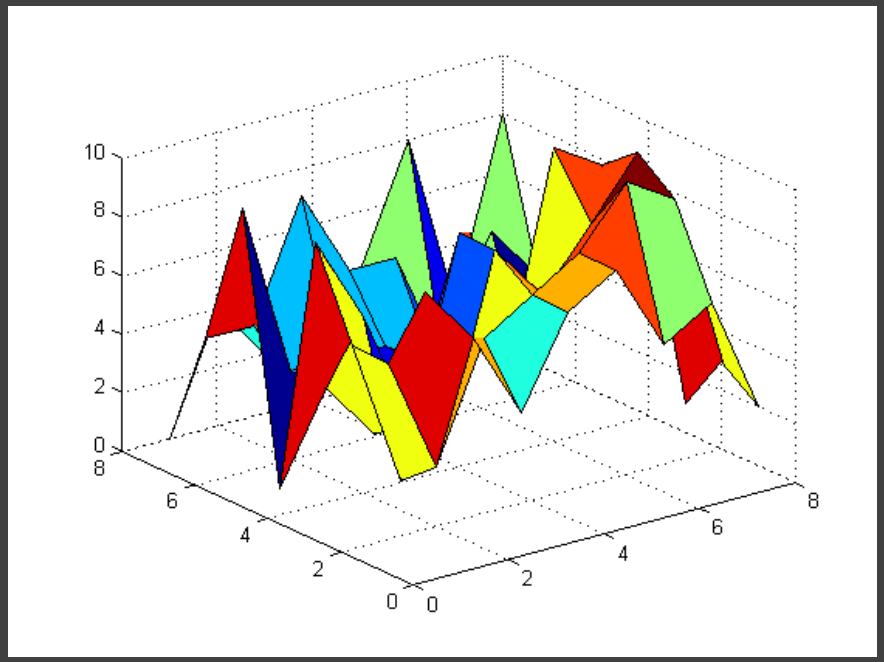
3D "plošné" grafy – 3D plochy a sítě

`mesh(X,Y,Z)` – vykreslí nad souřadnicemi *x*, *y* **síť** (**drátěný model**) tvarovanou podle *Z* (lze též uvést **`mesh(Z)`** – nemám potom regulérní hodnoty *x*, *y*)

`surf(X,Y,Z)` - vykreslí nad souřadnicemi *x*, *y* **plochu** (vystínovanou, vybarvenou) tvarovanou podle *Z* (lze též uvést **`surf(Z)`** – nemám potom regulérní hodnoty *x*, *y*)

Příklad:
Matice náhodných čísel s 8 řádky a 8 sloupci zobrazená pomocí **`surf`**. Rozsah osy *x* a *y* bude dán automaticky od 1 do 8 (indexy prvků matice).

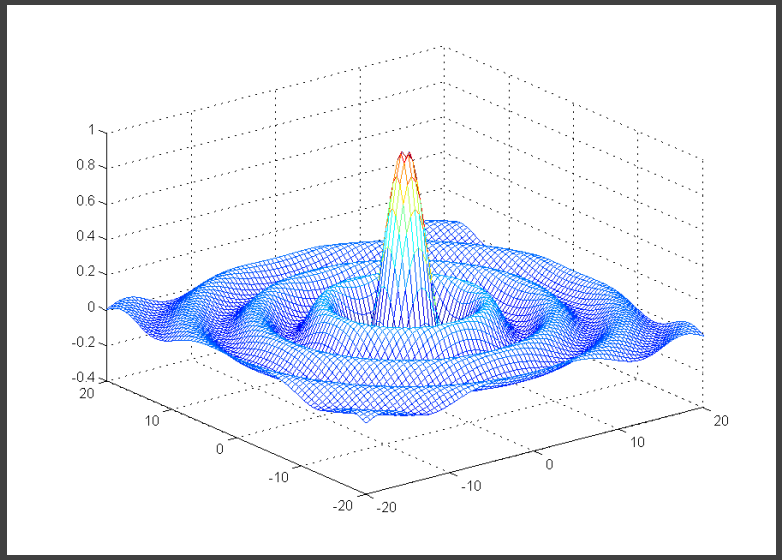
```
N = round(rand(8,8).*10);
surf(N)
```



Trojdimenzionální grafy

Příklad:
Graf funkce:

$$z(x,y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$



pro x, y od **-20** do **20** s krokem **0,5**.

```
x = -20:0.5:20;
y = x;
[X,Y] = meshgrid(x,y);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

Pokud mají x, y stejný rozsah **-20** až **20**, lze zapsat, např. takto:

```
[X,Y] = meshgrid(-20:0.5:20);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

Trojdimenzionální grafy

Proč používáme **meshgrid**?

- **[X,Y] = meshgrid(x,y)** vytvoří **pomocné matice X, Y**, jejichž prvky obsahují souřadnice bodů v rovině *xy*
- pro usnadnění zápisu výpočtu – s takto vytvořenými souřadnicemi mohu zapisovat rovnici pro výpočet **Z** "normálně" dle matematického zápisu, pouze nesmím zapomenout na operace prvek po prvku (tečka-notaci).

Pozn.: Funkce vracející jako výsledek dvě a více hodnot (mohou to být i dvě matice) bude mít hlavičku:

```
function [prvni,druha] = vraci_dve(vstupni_parametry)
...atd...
prvni = nějaký výsledek;
druha = nějaký výsledek;
```

Volání této funkce bude potom vypadat např.:

```
[a, b] = vraci_dve(x)
a = ...
b = ...
```

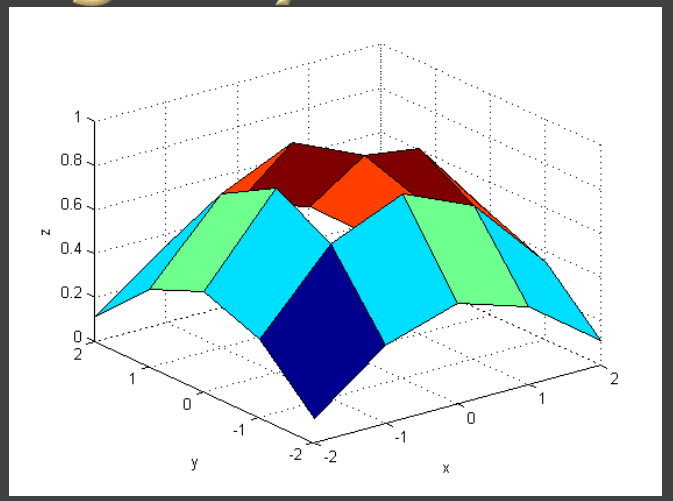
Příkladem takové funkce je **meshgrid**.

Trojdimenzionální grafy

Pokračování příkladu:

Graf funkce: $$z(x,y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$

pro x, y od **-2** do **2** s krokem **1**.




```
[X,Y] = meshgrid(-2:2)
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2)
surf(X,Y,Z)
xlabel('x')
ylabel('y')
zlabel('z')
```

Za příkazy nejsou středníky, vypíší se matice **X**, **Y**, **Z**

```
X =
    -2    -1     0     1     2
    -2    -1     0     1     2
    -2    -1     0     1     2
    -2    -1     0     1     2
    -2    -1     0     1     2
```

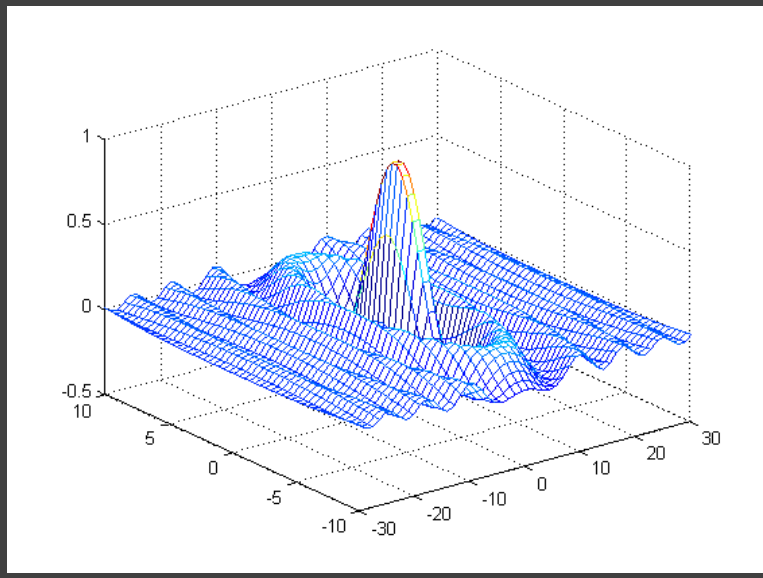
```
Y =
    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1
     0     0     0     0     0
     1     1     1     1     1
     2     2     2     2     2
```

```
Z =
    0.11    0.35    0.45    0.35    0.11
    0.35    0.70    0.84    0.70    0.35
    0.45    0.84     NaN    0.84    0.45
    0.35    0.70    0.84    0.70    0.35
    0.11    0.35    0.45    0.35    0.11
```

Trojdimenzionální grafy

Pokračování příkladu:

Graf funkce: $$z(x,y)=\frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$



pro x od -30 do 30 a y od -10 do 10
(různý rozsah os x, y)

```
x = linspace(-30,30,50);
y = linspace(-10,10,50);
[X,Y] = meshgrid(x,y);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

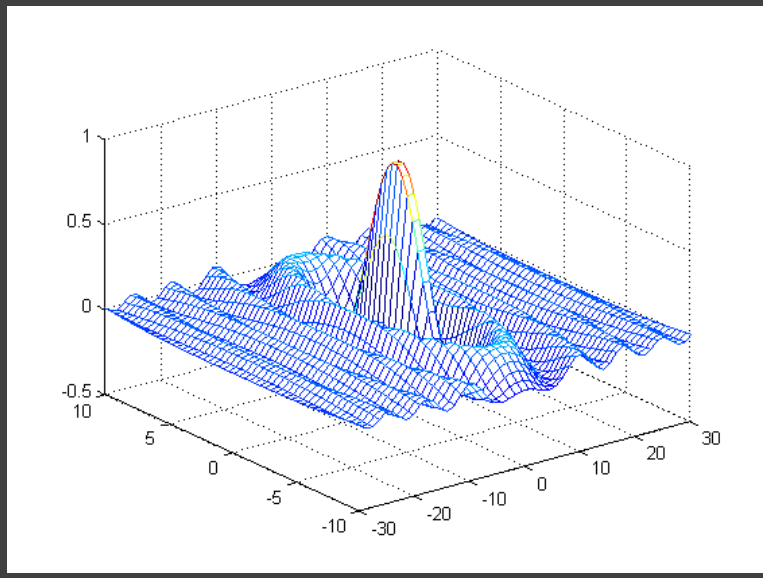
pro x od -20 do 20 a y od -40 do 40

```
[X,Y] = meshgrid(-20:0.5:20,-40:0.5:40);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

Trojdimenzionální grafy

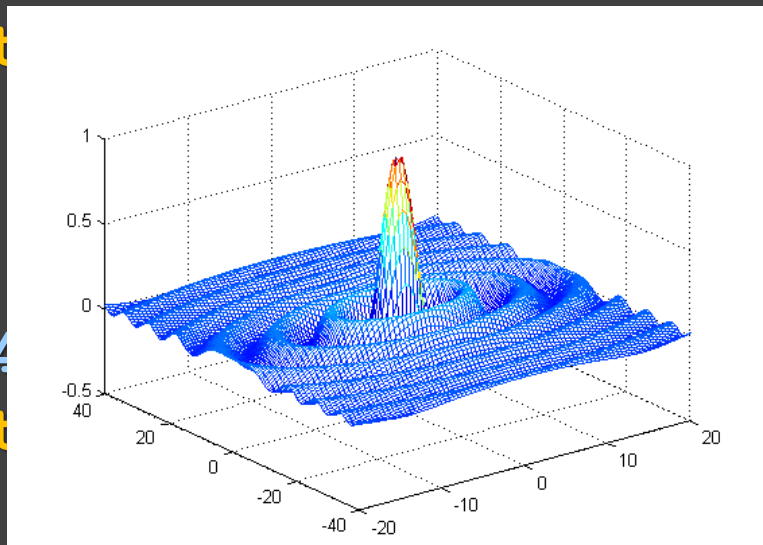
Pokračování příkladu:

Graf funkce: $$z(x,y)=\frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$



pro x od -30 do 30 a y od -10 do 10
(různý rozsah os x, y)

```
x = linspace(-30,30,50);
y = linspace(-10,10,50);
[X,Y] = meshgrid(x,y);
Z = sin(sqrt(X.^2+Y.^2))./sqrt
mesh(X,Y,Z)
```



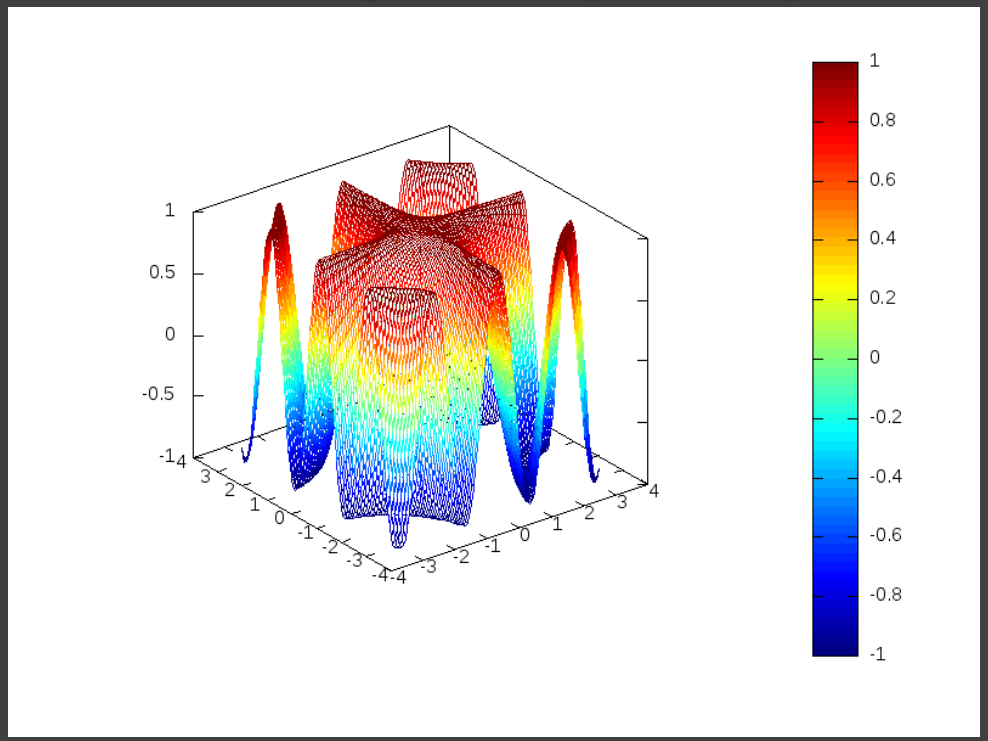
pro x od -20 do 20 a y od -40 do 40

```
[X,Y] = meshgrid(-20:0.5:20,-4
Z = sin(sqrt(X.^2+Y.^2))./sqrt
mesh(X,Y,Z)
```

Trojdimenzionální grafy

Příklad:
Graf funkce: $z = \cos(xy)$ pro x, y od $-\pi$ do π.
– funkční hodnoty vypočtené pomocí **meshgrid** a **operací prvek po prvku**

```
x = -pi:pi/50:pi;
y = x;
tic
[X,Y] = meshgrid(x,y);
Z = cos(X.*Y);
toc
mesh(X,Y,Z)
colorbar
```




```
   Elapsed time is 0.05995 seconds.
```

`colorbar` – doplní **barevnou stupnici** (škálu)

Trojdimenzionální grafy

Pokračování příkladu:
Graf funkce: $z = \cos(xy)$ pro x, y od $-\pi$ do π.
– funkční hodnoty vypočtené pomocí cyklů **for**

```
x = -pi:pi/50:pi;
y = x;
tic
for m = 1:length(x)
   for n = 1:length(y)
      Z(m,n)=cos(x(m)*y(n));
   end
end
toc
mesh(x,y,Z)
colorbar

   Elapsed time is 0.31 seconds.
```