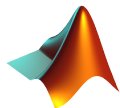


GNU Octave a Matlab – struktury, polynomy, zpracování souborů

Programování F1400 + F1400a
doc. RNDr. Petr Mikulík, Ph.D.

podzimní semestr 2020



```
fajls = '*.m'  
files = dir(fajls)  
fnames = { files.name }  
length(files), length(fnames)  
  
for k=1:length(files)  
    [cesta, jmeno, pripona] = fileparts(files(k).name);  
    fprintf('Soubor: %18s nebo %18s, ', fnames{k}, files(k).name)  
    fprintf('velikost %5g B, ', files(k).bytes)  
    fprintf('datum %s, pripona %s\n', files(k).date, pripona)  
end
```

Struktury, třídy a zapouzdření dat

- **Zapouzdření dat:** místo mnoha proměnných popisujících vlastnosti nějakého objektu použijí jedinou proměnnou, která má tyto vlastnosti jako položky.
- Terminologie v počítačových jazycích: **proměnná typu struktura (structure)** obsahuje data, **třída (class)** obsahuje data i metody nad daty pracujícími.
- Struktura – příklad pro GNU Octave a Matlab:

Příkazy:

```
clear

a_pocet_kol = 4;
a_palivo = 'benzin';
a_motor = 1600;

auto.pocet_kol = 4;
auto.palivo = 'benzin';
auto.motor = 1600;

auto2=auto; auto2.palivo='nafta';

auto
auto2
whos
```

Výstup:

```
auto =
  pocet_kol = 4
  palivo = benzin
  motor = 1600
auto2 =
  pocet_kol = 4
  palivo = nafta
  motor = 1600
Variables in the current scope:
  Name          Size  Bytes  Class
  a_motor       1x1    8     double
  a_palivo      1x6    6     char
  a_pocet_kol   1x1    8     double
  auto          1x1   22     struct
  auto2        1x1   21     struct
```

Zpracování více souborů

● Informace o jednom souboru:

```
1 fajl = 'a.txt'
2 [cesta, jmeno, pripona] = fileparts(fajl) % rozděl na cestu, jméno a příponu
3 finfo = dir(fajl) % informace z příkazu dir()
4 finfo.name % položka struktury: jméno souboru
5 finfo.date % položka struktury: datum změny souboru
6 datestr(finfo.datenum) % položka struktury: jiný formát data

7 fprintf('Informace o souboru "%s": velikost %g B, datum %s, pripona %s\n', ...
8 finfo.name, finfo.bytes, finfo.date, pripona)
```

● Zpracování více souborů:

```
1 fajls = '*.m' % filtr pro výběr souborů
2 files = dir(fajls) % příkaz dir() získá jména souborů a info o nich
3 fnames = { files.name } % do proměnné fnames dej pouze jména souborů

4 fprintf('*****\n') % oddělíme si výstup na obrazovku
5 for k=1:length(files)
6 [a,b,c] = fileparts(files(k).name);
7 fprintf('File: %18s: size %4g, date %s, extension %s\n', ...
8 files(k).name, files(k).bytes, files(k).date, c)
9 zpracuj_jeden_soubor(files(k).name); % a teď třeba zpracuj tento soubor
10 end
```

Polynomy

- **Polynom** $p(x) = p_1x^n + p_2x^{n-1} + \dots + p_n$ lze zadat jako **pole (vektor) koeficientů** $[p_1, p_2, \dots, p_n]$.
- Po zadání pole koeficientů můžeme následně využít funkce pro **nalezení kořenů** polynomu, spočtení **hodnot polynomu** pro zadaná x , derivaci či integraci polynomu, dělení polynomů, ...

```
octave> p = [1 6 5];           % x^2+6x+5, zadání polynomu pomocí koeficientů
octave> polyout(p, 'x');      % vypiš polynom symbolicky
1*x^2 + 6*x^1 + 5
```

```
octave> roots(p)              % vypiš kořeny x_i polynomu; pak p=product_i (x-x_i)
ans =
   -5
   -1
```

```
octave> x = -6:0.5:6; y = polyval(p,x); % spočti hodnoty polynomu pro daná x
```

```
octave> pderiv = polyder(p)    % spočti koeficienty derivovaného polynomu
pderiv =
     2     6
```

```
octave> yderiv = polyval(pderiv, x) % spočti hodnoty polynomu pro daná x
octave> plot(x,y, x,yderiv);      % vykresli graf polynomu a jeho derivace
octave> z = [x; y; yderiv]'      % případně vyrob tabulku hodnot
```

Polynomy – fitování dat

- **Proložení polynomu** daného stupně naměřenou křivkou **metodou nejmenších čtverců** se provádí funkcí `polyfit(x,y,n)`, kde `x` a `y` jsou naměřená data a `n` je stupeň požadovaného polynomu.

- Příklad „numerického experimentu“:

```
1 x = linspace(-5,5,30);           % vygenerujeme nějaká data: interval x,
2 y = x.*x + 2 + 5*rand(size(x));  % a parabola s náhodným šumem
3 figure(1); plot(x,y)            % vykreslíme nasimulovaná data
4 a = [x',y']; save -ascii rand_poly.dat a % data lze uložit do souboru

5 p = polyfit(x,y,2);             % data proložíme polynomem stupně 2
6 yfit = polyval(p,x);           % spočteme odpovídající hodnoty y
7 figure(2); plot(x,y,'r-*', x,yfit, 'b-', 'LineWidth', 2) % graf simulace i fitu
```

- Pro získání nejistot použijte (viz help polyfit):

```
1 n = 2;                          % polynomem kterého stupně chci data prokládat
2 [p,s] = polyfit(x,y,n);          % nafituj (prolož) data polynomem
3 stderrors = sqrt(diag(s.C)/s.df)*s.normr % získej hodnoty nejistot
4 fprintf('Nafitovaný polynom je: '); % vypiš získaný polynom
5 polyval(p, 'x');
6 for k=0:n
7     fprintf('Koeficient x%i: %8.4f +- %6.4f\n', k, p(k+1), stderrors(k+1));
8 end
```

Polynomy – fitování dat z mnoha souborů

- Mnoha textovými datovými soubory (obsahují dva sloupce x a y) chceme proložit **lineární funkce**, tj. polynom 1. stupně, a vypsat nafitované směrnice a posuny:

```
1 fajls = '*.dat'           % nastavení: vstupní datové soubory
2 files = dir(fajls);      % získkej seznam vstupních souborů
3 fnames = { files.name }  % ze seznamu vyber pouze jména souborů
4 fprintf('\nProkládám přímku (lineární polynom,  $y(x)=a*x+b$ ) naměřenými daty\n');
5 fprintf('Počet datových souborů ke zpracování: %i\n', length(fnames));
6 for k=1:length(fnames)
7     a = load(fnames{k});  % načti celý soubor do matice
8     x = a(:,1);          % první sloupec: hodnoty x
9     y = a(:,2);          % druhý sloupec: hodnoty y
10    [p,s] = polyfit(x,y,1); % prolož načtenými daty přímku (polynom 1. stupně)
11    stderrors = sqrt(diag(s.C)/s.df)*s.normr; % získkej hodnoty nejistot
12
13    fprintf('Datový soubor: %s: ', files(k).name); % vypiš výsledek
14    fprintf('směrnice %6.3f +- %6.3f, posun %6.3f +- %6.3f\n', ...
15            p(1), stderrors(1), p(2), stderrors(2));
16 end
```

Jednorozměrná náhodná procházka

- **Jednorozměrná náhodná procházka** aneb opilý námořník na přímce. Nechť se námořník pohybuje po přímce o délce např. 800 kroků, začne uprostřed, a v každém tahu se náhodně rozhodne, zdali udělá krok vlevo nebo vpravo. Spočítejte počet výskytů námořníka podél přímky (s přesností na kroky).

```
1 % Nastaveni:
2 n = 800;           % delka usecky, na ktere se namornik pohybuje
3 pocet_kroku = 15000; % kolik kroku ma namornik udelat
4
5 % Program:
6 a = zeros(1,n); % pocitadlo vyskytu namornik v jednotlivych usecich
7 x = floor(n/2); % vychozi misto (poloha hospody)
8 osa_x = [-floor(n/2) : floor((n-1)/2)];
9 a(x) = 1; % namornik se na zacatku nachazi pred hospodou
10 for k=1:pocet_kroku
11     dx = 2*(randi(2)-1) - 1; % mapuj nahodna cisla [1,2] na [-1, +1]
12     x = x + dx; % posun se doleva nebo doprava
13     if (x==0 || x==n+1) break; end % namornik dorazil na kraj usecky
14     a(x) = a(x) + 1; % dorazil jsem na toto misto, tak zvetsi jeho cetnost
15     fprintf('%3i. dx= %2i   x= %2i\n', k, dx, x);
16     if (mod(k,1000)==0) % kazdych 1000 iteraci vykresli obrazek
17         k, plot(osa_x, a); pause(0.01); % kratka pauza pro prekresleni
18     end
19 end
20 plot(osa_x, a); grid
```

Dvourozměrná náhodná procházka („Brownův pohyb“)

• Dvourozměrná náhodná procházka aneb opilý námořník na ploše:

```
1 % Nastaveni:
2 n = 600;           % rozmer sachovnice, na ktore se namornik pohybuje
3 pocet_kroku = 10000; % kolik kroku ma namornik udelat

4 % Program:
5 a = zeros(n,n);   % pocitadlo vyskytu namornika na jednotlivych polickach
6 r = floor(n/2);  c = floor(n/2); % vychozi misto

7 a(r,c) = 1;      % namornik se na zacatku nachazi pred hospodou
8 for k=1:pocet_kroku
9     delta = 2*(randi(2,1,2)-1) - 1; % mapuj nahodna cisla [1,2] na [-1, +1]
10    r = r + delta(1); % posun se nahoru nebo dolu
11    c = c + delta(2); % posun se doleva nebo doprava
12    if (r==0 || r==n+1) break; end % namornik dorazil na kraj sachovnice
13    if (c==0 || c==n+1) break; end
14    a(r,c) = a(r,c) + 1; % byl jsem tu
15    fprintf('%3i. delta= %2i %2i rc= %2i %2i\n', k, delta(1), delta(2), r, c);
16    if (mod(k,1000)==0) % kazdych 1000 iteraci vykresli obrazek
17        k, imagesc(a); colorbar; pause(0.01); % kratka pauza pro prekresleni
18    end
19 end
20 colormap(rainbow(max(a(:))+1));
21 imagesc(a); colorbar
```