UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Računalniško podprto geometrijsko oblikovanje

Krivulje in ploskve v Matlabu

Jan Grošelj

Ljubljana, 2024

O gradivu. V zbirki nalog so obravnavane vsebine iz teorije Bézierjevih krivulj in ploskev. Zbirka je nastala med izvajanjem vaj pri predmetu Računalniško podprto geometrijsko oblikovanje na magistrskih študijskih programih Fakultete za matematiko in fiziko v Ljubljani.

Naloge v tej zbirki podajajo navodila za implementacijo Bézierjevih krivulj in ploskev v Matlabu. Najprej so obravnavane predstavitve in lastnosti polinomskih in racionalnih krivulj ter zlepkov. Nato so ti koncepti posplošeni na dve spremenljivki s ploskvami iz tenzorskega produkta in trikotnimi ploskvami.

Pri vsaki izmed nalog je na kratko predstavljeno teoretično ozadje. Temu sledi opis in specifikacija metod, ki jih je treba v okviru naloge implementirati. Na koncu so prikazani primeri uporabe metod, s katerimi se lahko preveri pravilnost implementacije.

Zadnja sprememba: 16. december 2024

Kazalo

Naloga 1. *Zveza med Bernsteinovo in potenčno bazo* **1**

Naloga 2. *De Casteljaujev postopek* **4**

Naloga 3. *Odvodi Bézierjeve krivulje* **9**

Naloga 4. *Opisi krožnice z Bézierjevimi krivuljami* **14**

Naloga 5. *Subdivizija Bézierjeve krivulje* **18**

Naloga 6. *Višanje stopnje Bézierjeve krivulje* **21**

Naloga 7. *Parametrizacija sestavljene krivulje* **24**

Naloga 8. *Kubični C^2 zlepek* **26**

Naloga 9. *Racionalne Bézierjeve krivulje in Farinove točke* **29**

Naloga 10. *Višanje stopnje racionalne Bézierjeve krivulje* **32**

Naloga 11. *Bézierjeve ploskve iz tenzorskega produkta* **35**

Naloga 12. *Coonsove ploskve* **38**

Naloga 13. *Aproksimacija z Bézierjevimi ploskvami po metodi najmanjših kvadratov* **42**

Naloga 14. *Aproksimacija s sestavljenimi Bézierjevimi ploskvami* **45**

Naloga 15. *Polinomi dveh spremenljivk* **50**

Naloga 16. *Trikotne Bézierjeve ploskve* **54**

Naloga 17. *Argyrisova interpolacijska shema* **57**

Naloga 18. *Argyrisov zlepek* **62**

Naloga 1. *Zveza med Bernsteinovo in potenčno bazo.*

Bernsteinovi bazni polinomi stopnje n so podani z

$$B_i^n(x) = \binom{n}{i} x^i (1-x)^{n-i}, \qquad i = 0, 1, \ldots, n,$$

in sestavljajo bazo za prostor polinomov stopnje manjše ali enake n. S funkcijami $x \mapsto x^i$, $i = 0, 1, \ldots, n$, ki sestavljajo potenčno bazo, so v naslednjih zvezah:

$$x^i = \sum_{j=i}^{n} \frac{\binom{j}{i}}{\binom{n}{i}} B_j^n(x), \quad B_i^n(x) = \sum_{j=i}^{n} (-1)^{i+j} \binom{n}{j} \binom{j}{i} x^j, \qquad i = 0, 1, \ldots, n.$$

1. V Matlabu implementirajte metodo `power2bernstein`, ki koeficiente polinoma, izražene v potenčni bazi, pretvori v koeficiente istega polinoma, izraženega v Bernsteinovi bazi, ter metodo `bernstein2power`, ki napravi obratno.

   Opis metode:

```
function b = power2bernstein(p)
% Opis:
%  power2bernstein pretvori polinom, predstavljen s koeficienti v
%  potenčni bazi, v polinom, predstavljen v Bernsteinovi bazi
%
% Definicija:
%  b = power2bernstein(p)
%
% Vhodni podatek:
%  p    seznam koeficientov dolžine n+1, ki po vrsti pripadajo razvoju
%       polinoma stopnje n v potenčni bazi od x^n do 1
%
% Izhodni podatek:
%  b    seznam koeficientov dolžine n+1, ki po vrsti pripadajo razvoju
%       polinoma stopnje n v Bernsteinovi bazi od 0-tega do n-tega
%       Bernsteinovega baznega polinoma
end
```

   Primeri:

```
% polinom v potenčni bazi
p = [4 7 2 8 9 1 4];
```

```
% prevedba v Bernsteinovo bazo
b = power2bernstein(p)
```

```
b = 1x7
   4.0000   4.1667   4.9333   6.7000  10.0000  16.6667  35.0000
```

```
% prevedba nazaj v potenčno bazo
p = bernstein2power(b)
```

```
p = 1x7
   4.0000   7.0000   2.0000   8.0000   9.0000   1.0000   4.0000
```

2. Oglejte si izražavo polinomov $x \mapsto 1$ in $x \mapsto x$ v Bernsteinovi bazi in na ta način za nekaj nizkih stopenj n utemeljite, da operator $B_n f = \sum_{i=0}^{n} f(i/n) B_i^n$ ohranja polinome stopnje manjše ali enake 1.

   Primeri:

```
power2bernstein(1)
```

```
ans =
     1
```

```
power2bernstein([0 1])
```

```
ans = 1x2
     1     1
```

```
power2bernstein([0 0 1])
```

```
ans = 1x3
     1     1     1
```

```
power2bernstein([1 0])
```

```
ans = 1x2
     0     1
```

```
power2bernstein([0 1 0])
```

```
ans = 1x3
     0    0.5000    1.0000
```

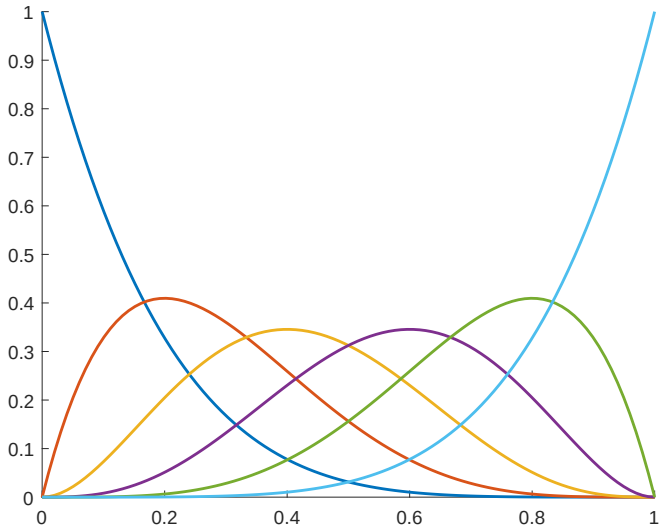
```
power2bernstein([0 0 1 0])
```

```
ans = 1x4
     0    0.3333    0.6667    1.0000
```

3. Narišite Bernsteinove bazne polinome stopnje 5. Pomagajte si s prevedbo v potenčno bazo in ukazom `polyval`.

   Primeri:

```
n = 5;
x = linspace(0,1);
clf
hold on
for i = 0:n
    b = zeros(1,n+1);
    b(i+1) = 1;
    plot(x,polyval(bernstein2power(b),x));
end
hold off
```



Grafi Bernsteinovih baznih polinomov stopnje 5.

Naloga 2. *De Casteljaujev postopek.*

Bézierjeva krivulja stopnje n s kontrolnimi točkami $\boldsymbol{b}_i \in \mathbb{R}^d$, $i = 0, 1, \ldots, n$, je krivulja v $\mathbb{R}^d$, podana s parametrizacijo

$$\boldsymbol{b}(t) = \sum_{i=0}^{n} \boldsymbol{b}_i B_i^n(t), \qquad t \in [0, 1].$$

Točko na krivulji pri izbranem t lahko dobimo s pomočjo de Casteljaujevega postopka

$$\boldsymbol{b}_i^r = (1-t)\boldsymbol{b}_i^{r-1} + t\boldsymbol{b}_{i+1}^{r-1}, \qquad i = 0, 1, \ldots, n-r, \quad r = 1, 2, \ldots, n,$$

kjer na začetku vzamemo $\boldsymbol{b}_i^0 = \boldsymbol{b}_i$, zadnja izračunana točka $\boldsymbol{b}_0^n$ pa ustreza $\boldsymbol{b}(t)$. V Matlabu pripravite naslednje metode za izračun in izris Bézierjeve krivulje s pomočjo tega postopka.

1. Metoda `decasteljau` naj izračuna de Casteljaujevo shemo točk $\boldsymbol{b}_i^r$, ki jo določajo podane kontrolne točke $\boldsymbol{b}_i$ (oziroma njihove koordinate) in izbran parameter $t \in [0, 1]$.

   Opis metode:

```
function D = decasteljau(b,t)
% Opis:
%  decasteljau vrne shemo de Casteljaujevega postopka za dan seznam
%  koordinat b pri danem parametru t
%
% Definicija:
%  D = decasteljau(b,t)
%
% Vhodna podatka:
%  b    seznam koordinat kontrolnih točk Bezierjeve krivulje
%       stopnje n,
%  t    parameter, pri katerem računamo koordinato Bezierjeve
%       krivulje
%
% Izhodni podatek:
%  D    tabela velikosti n+1 x n+1, ki predstavlja de Casteljaujevo
%       shemo za koordinate b pri parametru t (element na mestu
%       (1,n+1) je koordinata Bezierjeve krivulje pri parametru t,
%       elementi na mestih (i,j) za i > n-j+2 so NaN)
end
```

   Primeri:

```
% kontrolne točke Bezierjeve krivulje
B = [0 0; 1 2; 3 3; 4 -1; 0 2];
```

```
decasteljau(B(:,1),0)
```

```
ans = 5x5
     0     0     0     0     0
     1     1     1     1   NaN
     3     3     3   NaN   NaN
     4     4   NaN   NaN   NaN
     0   NaN   NaN   NaN   NaN
```

```
decasteljau(B(:,2),0)
```

```
ans = 5x5
     0     0     0     0     0
     2     2     2     2   NaN
     3     3     3   NaN   NaN
    -1    -1   NaN   NaN   NaN
     2   NaN   NaN   NaN   NaN
```

```
decasteljau(B(:,1),1)
```

```
ans = 5x5
     0     1     3     4     0
     1     3     4     0   NaN
     3     4     0   NaN   NaN
     4     0   NaN   NaN   NaN
     0   NaN   NaN   NaN   NaN
```

```
decasteljau(B(:,2),1)
```

```
ans = 5x5
     0     2     3    -1     2
     2     3    -1     2   NaN
     3    -1     2   NaN   NaN
    -1     2   NaN   NaN   NaN
     2   NaN   NaN   NaN   NaN
```

```
decasteljau(B(:,1),0.5)
```

```
ans = 5x5
         0    0.5000    1.2500    2.0000    2.3750
    1.0000    2.0000    2.7500    2.7500       NaN
    3.0000    3.5000    2.7500       NaN       NaN
    4.0000    2.0000       NaN       NaN       NaN
         0       NaN       NaN       NaN       NaN
```

```
decasteljau(B(:,2),0.5)
```

```
ans = 5x5
         0    1.0000    1.7500    1.7500    1.5000
    2.0000    2.5000    1.7500    1.2500       NaN
    3.0000    1.0000    0.7500       NaN       NaN
   -1.0000    0.5000       NaN       NaN       NaN
    2.0000       NaN       NaN       NaN       NaN
```

2. Metoda `bezier` naj s pomočjo metode `decasteljau` izračuna točke na Bézierjevi krivulji pri danem seznamu parametrov z intervala $[0, 1]$. Bézierjeva krivulja naj bo podana v obliki seznama kontrolnih točk.

   Opis metode:

```
function b = bezier(B,t)
% Opis:
%  bezier vrne točke na Bezierjevi krivulji pri danih parametrih
%
% Definicija:
%  b = bezier(B,t)
%
% Vhodna podatka:
%  B    matrika velikosti n+1 x d, ki predstavlja kontrolne točke
%       Bezierjeve krivulje stopnje n v d-dimenzionalnem prostoru,
%  t    seznam parametrov dolžine k, pri katerih računamo vrednost
%       Bezierjeve krivulje
%
% Izhodni podatek:
%  b    matrika velikosti k x d, kjer i-ta vrstica predstavlja točko
%       na Bezierjevi krivulji pri parametru iz t na i-tem mestu
end
```

Primeri:

```
B = [0 0; 1 2; 3 3; 4 -1; 0 2];
t = linspace(0,1,10);
bezier(B,t)
```

```
ans = 10x2
         0         0
    0.5072    0.7953
    1.0925    1.3449
    1.6790    1.6049
    2.1826    1.5900
    2.5118    1.3733
    2.5679    1.0864
    2.2448    0.9197
    1.4291    1.1218
         0    2.0000
```

3. Metoda `plotbezier` naj na podlagi kontrolnih točk in seznama parametrov nariše Bézierjevo krivuljo ter njen kontrolni poligon. Metoda naj omogoča izris polinoma v Bernsteinovi bazi ter dvodimenzionalne in tridimenzionalne krivulje. Nato narišite Bézierjevo krivuljo stopnje 3 in si oglejte njene lastnosti v odvisnosti od kontrolnih točk. Z upoštevanjem afine invariantnosti premaknite, rotirajte in zrcalite krivuljo.

   Opis metode:

```
function p = plotbezier(B,t,c)
% Opis:
%  plotbezier nariše Bezierjevo krivuljo za dane kontrolne točke in
%  seznam parametrov
%
% Definicija:
%  p = plotbezier(B,t,c)
%
% Vhodni podatki:
%  B    matrika velikosti n+1 x d, ki predstavlja kontrolne točke
%       Bezierjeve krivulje stopnje n v d-dimenzionalnem prostoru,
%  t    seznam parametrov dolžine k, pri katerih računamo vrednost
%       Bezierjeve krivulje,
%  c    opcijski parameter, ki določa barvo krivulje
%
% Izhodni podatek:
%  p    grafični objekt, ki določa krivuljo
end
```

Primeri:

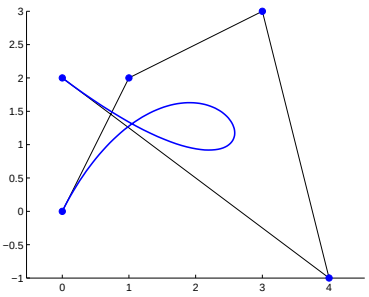
```
B = [0 0; 1 2; 3 3; 4 -1; 0 2];
t = linspace(0,1);
plotbezier(B,t);
```

```
% zrcaljenje krivulje čez y os
Bz = B*[-1 0; 0 1]
plotbezier(Bz,t);
```

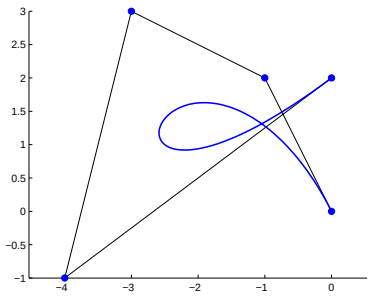
```
Bz = 5x2
     0     0
    -1     2
    -3     3
    -4    -1
     0     2
```

```
% rotacija krivulje za 60 stopinj
fi = pi/3;
Br = B*[cos(fi) -sin(fi); sin(fi) cos(fi)]'
plotbezier(Br,t);
```

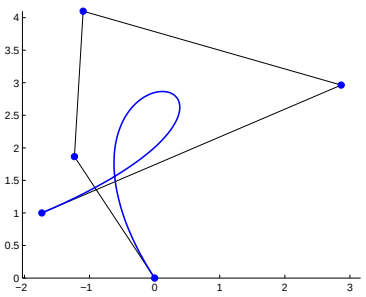
```
Br = 5x2
         0         0
   -1.2321    1.8660
   -1.0981    4.0981
    2.8660    2.9641
   -1.7321    1.0000
```



(a) krivulja



(b) zrcaljenje čez y os



(c) rotacija za 60°

Primer Bézierjeve krivulje in dveh njenih afinih transformacij.

Naloga 3. *Odvodi Bézierjeve krivulje.*

Za računanje r-tega odvoda Bézierjeve krivulje

$$\boldsymbol{b}(t) = \sum_{i=0}^{n} \boldsymbol{b}_i B_i^n(t), \qquad t \in [0,1],$$

stopnje n lahko uporabimo njegovo izražavo v obliki Bézierjeve krivulje

$$\frac{\mathrm{d}^r}{\mathrm{d}t^r}\boldsymbol{b}(t) = \frac{n!}{(n-r)!}\sum_{i=0}^{n-r} \Delta^r \boldsymbol{b}_i B_i^{n-r}(t)$$

stopnje $n-r$, kjer $\Delta^r \boldsymbol{b}_i$ označuje r-to deljeno diferenco kontrolne točke $\boldsymbol{b}_i$,

$$\Delta^0 \boldsymbol{b}_i := \boldsymbol{b}_i, \quad \Delta^r \boldsymbol{b}_i := \Delta^{r-1}\boldsymbol{b}_{i+1} - \Delta^{r-1}\boldsymbol{b}_i, \quad r = 1, 2, \ldots.$$

Če imamo že na voljo de Casteljaujevo shemo za krivuljo $\boldsymbol{b}$ pri parametru t, pa je učinkoviteje vrednost odvoda določiti kar na podlagi vmesnih kontrolnih točk $\boldsymbol{b}_i^{n-r}(t)$, $i = 0, 1, \ldots, r$, v $(n-r)$-tem koraku postopka kot

$$\frac{\mathrm{d}^r}{\mathrm{d}t^r}\boldsymbol{b}(t) = \frac{n!}{(n-r)!}\Delta^r \boldsymbol{b}_0^{n-r}(t).$$

1. V Matlabu pripravite metodo `bezierder`, ki s pomočjo metode `decasteljau` izračuna r-ti odvod Bézierjeve krivulje $\boldsymbol{b}$ pri danih parametrih.

   Opis metode:

```
function db = bezierder(B,r,t)
% Opis:
%  bezierder vrne točke na krivulji, ki predstavlja odvod dane
%  Bezierjeve krivulje
%
% Definicija:
%  db = bezierder(B,r,t)
%
% Vhodni podatki:
%  B    matrika kontrolnih točk Bezierjeve krivulje, v kateri vsaka
%       vrstica predstavlja eno kontrolno točko,
%  r    stopnja odvoda, ki ga računamo,
%  t    seznam parameterov, pri katerih računamo odvod
%
% Izhodni podatek:
%  db   matrika, v kateri vsaka vrstica predstavlja točko r-tega
%       odvoda pri istoležnem parametru iz seznama t
end
```

Primeri:

```
B = [-2/3 -4/5; 1/3 1/5; 0 0; -1/3 1/5; 2/3 -4/5];
t = linspace(0,1,9);
```

```
d1b = bezierder(B,1,t)
```

```
d1b = 9x2
    4.0000    4.0000
    2.2500    2.4750
    1.0000    1.4000
    0.2500    0.6250
         0         0
    0.2500   -0.6250
    1.0000   -1.4000
    2.2500   -2.4750
    4.0000   -4.0000
```

```
d2b = bezierder(B,2,t)
```

```
d2b = 9x2
  -16.0000  -14.4000
  -12.0000  -10.2000
   -8.0000   -7.2000
   -4.0000   -5.4000
         0   -4.8000
    4.0000   -5.4000
    8.0000   -7.2000
   12.0000  -10.2000
   16.0000  -14.4000
```

```
d3b = bezierder(B,3,t)
```

```
d3b = 9x2
   32.0000   38.4000
   32.0000   28.8000
   32.0000   19.2000
   32.0000    9.6000
   32.0000         0
   32.0000   -9.6000
   32.0000  -19.2000
   32.0000  -28.8000
   32.0000  -38.4000
```

```
d4b = bezierder(B,4,t)
```

```
d4b = 9x2
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
```

2. Dopolnite metodo `bezierder` tako, da se jo lahko izvede z dvema izhodnima podatkoma. Drugi izhodni podatek naj bodo kontrolne točke Bézierjeve krivulje stopnje $n - r$, ki predstavlja r-ti odvod podane Bézierjeve krivulje. V tem primeru naj metoda vrednosti `db` pridobi z izračunom točk na Bézierjevi krivulji, ki predstavlja odvod. Preverite, da se vrednosti ujemajo z rezultati, dobljeni s pomočjo de Casteljaujeve sheme.

   Primeri:

```
[d1b,d1B] = bezierder(B,1,t)
```

```
d1b = 9x2
    4.0000    4.0000
    2.2500    2.4750
    1.0000    1.4000
    0.2500    0.6250
    0.0000         0
    0.2500   -0.6250
    1.0000   -1.4000
    2.2500   -2.4750
    4.0000   -4.0000
d1B = 4x2
    4.0000    4.0000
   -1.3333   -0.8000
   -1.3333    0.8000
    4.0000   -4.0000
```

```
[d2b,d2B] = bezierder(B,2,t)
```

```
d2b = 9x2
  -16.0000  -14.4000
```

```
  -12.0000  -10.2000
   -8.0000   -7.2000
   -4.0000   -5.4000
         0   -4.8000
    4.0000   -5.4000
    8.0000   -7.2000
   12.0000  -10.2000
   16.0000  -14.4000
d2B = 3x2
  -16.0000  -14.4000
         0    4.8000
   16.0000  -14.4000
```

```
[d3b,d3B] = bezierder(B,3,t)
```

```
d3b = 9x2
   32.0000   38.4000
   32.0000   28.8000
   32.0000   19.2000
   32.0000    9.6000
   32.0000         0
   32.0000   -9.6000
   32.0000  -19.2000
   32.0000  -28.8000
   32.0000  -38.4000
d3B = 2x2
   32.0000   38.4000
   32.0000  -38.4000
```

```
[d4b,d4B] = bezierder(B,4,t)
```

```
d4b = 9x2
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
         0  -76.8000
d4B = 1x2
         0  -76.8000
```

3. Bézierjeva krivulja $\boldsymbol{b}$ stopnje 4 s kontrolnimi točkami

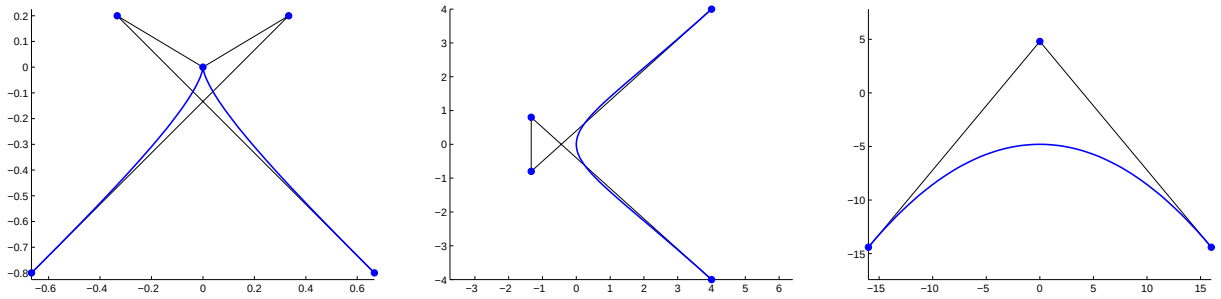
$$\boldsymbol{b}_0 = (-\tfrac{2}{3}, -\tfrac{4}{5}), \quad \boldsymbol{b}_1 = (\tfrac{1}{3}, \tfrac{1}{5}), \quad \boldsymbol{b}_2 = (0, 0), \quad \boldsymbol{b}_3 = (-\tfrac{1}{3}, \tfrac{1}{5}), \quad \boldsymbol{b}_4 = (\tfrac{2}{3}, -\tfrac{4}{5})$$

ima pri parametru $t = 1/2$ ima špico. Narišite krivuljo, ki predstavlja odvod $\boldsymbol{b}$, in si oglejte, kaj se dogaja v okolici tega parametra.

```
t = (0.46:0.01:0.54)';
d1b = bezierder(B,1,t);
tab = table();
tab{:,'t'} = t;
tab{:,'x'} = d1b(:,1);
tab{:,'y'} = d1b(:,2);
```

	t	x	y
1	0.4600	0.0256	0.1928
2	0.4700	0.0144	0.1443
3	0.4800	0.0064	0.0961
4	0.4900	0.0016	0.0480
5	0.5000	0	0
6	0.5100	0.0016	-0.0480
7	0.5200	0.0064	-0.0961
8	0.5300	0.0144	-0.1443
9	0.5400	0.0256	-0.1928

```
t = linspace(0,1);
[d1b,d1B] = bezierder(B,1,t);
[d2b,d2B] = bezierder(B,2,t);
plotbezier(B,t);
plotbezier(d1B,t);
plotbezier(d2B,t);
```



(a) krivulja (b) prvi odvod krivulje (c) drugi odvod krivulje

Primer Bézierjeve krivulje in njenih odvodov.

Naloga 4. *Opisi krožnice z Bézierjevimi krivuljami.*

S polinomskimi krivuljami ni mogoče eksaktno opisati krožnice, zato je treba v sistemih, ki omogočajo le delo s polinomskimi krivuljami, poseči po aproksimaciji. Aproksimacijo krožnice običajno dobimo z aproksimacijami krožnih lokov, ki jih zlepimo skupaj, saj kvalitetna aproksimacija celotne krožnice s polinomsko krivuljo zahteva visoko stopnjo. Pripravite metodo `bezierarc`, s katero lahko na tri različne načine, opisane v nadaljevanju, določimo kontrolne točke Bézierjeve krivulje $\boldsymbol{b} : [0, 1] \to \mathbb{R}^2$ nizke stopnje, ki dobro opisuje krožni lok. Aproksimiramo krožni lok, ki zavzame kote v območju $(-\varphi, \varphi)$, $\varphi > 0$, in za aproksimacijo zahtevamo, da v robnih točkah interpolira krožnico.

Opis metode:

```
function B = bezierarc(fi,m)
% Opis:
%  bezierarc izračuna kontrole točke Bezierjeve krivulje, ki
%  predstavlja interpolacijo krožnega loka po izbrani metodi
%
% Definicija:
%  B = bezierarc(fi,m)
%
% Vhodna podatka:
%  fi   kot, ki določa krozni lok v območju (-fi,fi),
%  m    metoda interpolacije:
%        1 = kvadratični G1 interpolant,
%        2 = kubični C1 interpolant,
%        3 = kubični G1 interpolant s C0 interpolacijo v srednji točki
%
% Izhodni podatek:
%  B    tabela velikosti 3 x 2 ali 4 x 2, v kateri vsaka vrstica
%       predstavlja kontrolno točko Bezierjeve krivulje
end
```

Naj bodo $\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ kontrolne točke Bézierjeve krivulje $\boldsymbol{b}$ nizke stopnje n ($n = 2$ ali $n = 3$). Iz interpolacije točk v robovih krožnega loka sledi

$$\boldsymbol{b}_0 = (\cos(\varphi), -\sin(\varphi)), \qquad \boldsymbol{b}_n = (\cos(\varphi), \sin(\varphi)).$$

Smeri tangent krivulje $\boldsymbol{b}$ se v robovih ujemata s smerema tangent krožnice, če za neki pozitivni realni števili d_1 in d_2 velja $\boldsymbol{b}'(0) = d_1(\sin(\varphi), \cos(\varphi))$ in $\boldsymbol{b}'(1) = d_2(-\sin(\varphi), \cos(\varphi))$, od kjer po formuli za odvod Bézierjeve krivulje sledi

$$\boldsymbol{b}_1 = \boldsymbol{b}_0 + \tfrac{1}{n}\boldsymbol{b}'(0) = \left(\cos(\varphi) + \tfrac{1}{n}d_1 \sin(\varphi), -\sin(\varphi) + \tfrac{1}{n}d_1 \cos(\varphi)\right),$$
$$\boldsymbol{b}_{n-1} = \boldsymbol{b}_n - \tfrac{1}{n}\boldsymbol{b}'(1) = \left(\cos(\varphi) + \tfrac{1}{n}d_2 \sin(\varphi), \sin(\varphi) - \tfrac{1}{n}d_2 \cos(\varphi)\right).$$

1. Obravnavajmo Bézierjevo krivuljo stopnje $n = 2$, ki v robovih interpolira krožnico ter smer njenih tangent. Ker ima taka Bézierjeva krivulja samo tri kontrolne točke, iz zgornjih ugotovitev sledi, da sta $\boldsymbol{b}_0$ in $\boldsymbol{b}_2$ že določeni, za $\boldsymbol{b}_1$ pa dobimo pogoja

$$\cos(\varphi) + \tfrac{1}{2}d_1 \sin(\varphi) = \cos(\varphi) + \tfrac{1}{2}d_2 \sin(\varphi),$$
$$-\sin(\varphi) + \tfrac{1}{2}d_1 \cos(\varphi) = \sin(\varphi) - \tfrac{1}{2}d_2 \cos(\varphi).$$

   Iz prve enačbe sledi $d_1 = d_2$, iz druge $d_1 = d_2 = 2\tan(\varphi)$. Torej je $\boldsymbol{b}_1 = (1/\cos(\varphi), 0)$.

2. Določimo Bézierjevo krivuljo stopnje $n = 3$, ki v robovih interpolira krožnico ter njena odvoda. To pomeni, da mora biti $d_1 = d_2 = 1$. S tem so po zgornjih formulah določene vse štiri kontrolne točke ($\boldsymbol{b}_0$, $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, $\boldsymbol{b}_3$) Bézierjeve krivulje.

3. Izpeljimo še Bézierjevo krivuljo stopnje $n = 3$, ki v robovih interpolira krožnico ter smer njenih tangent, poleg tega pa zanjo velja še $\boldsymbol{b}(1/2) = (1, 0)$. Ta dodaten pogoj implicira

$$\tfrac{1}{8}\boldsymbol{b}_0 + \tfrac{3}{8}\boldsymbol{b}_1 + \tfrac{3}{8}\boldsymbol{b}_2 + \tfrac{1}{8}\boldsymbol{b}_3 = (1, 0).$$

   V drugi komponenti zgornje enačbe dobimo

$$-\tfrac{1}{8}\sin(\varphi) + \tfrac{3}{8}\left(-\sin(\varphi) + \tfrac{1}{3}d_1 \cos(\varphi)\right) + \tfrac{3}{8}\left(\sin(\varphi) - \tfrac{1}{3}d_2 \cos(\varphi)\right) + \tfrac{1}{8}\sin(\varphi) = 0,$$

   iz česar je razvidno, da mora biti $d = d_1 = d_2$. Iz prve komponente enačbe, ki jo zapišemo v obliki

$$\tfrac{1}{8}\cos(\varphi) + \tfrac{3}{8}\left(\cos(\varphi) + \tfrac{1}{3}d_1 \sin(\varphi)\right) + \tfrac{3}{8}\left(\cos(\varphi) + \tfrac{1}{3}d_2 \sin(\varphi)\right) + \tfrac{1}{8}\cos(\varphi) = 1,$$

   potem sledi

$$\cos(\varphi) + \tfrac{1}{4}d \sin(\varphi) = 1,$$

   kar pomeni, da je $d = 4(1/\sin(\varphi) - \cot(\varphi))$. Z nekaj dodatnega računanja izpeljemo, da sta kontrolni točki $\boldsymbol{b}_1$ in $\boldsymbol{b}_2$ dani z

$$\boldsymbol{b}_1 = \tfrac{1}{3}\left(4 - \cos(\varphi), 4\cot(\varphi) - \tfrac{4}{\sin(\varphi)} + \sin(\varphi)\right),$$
$$\boldsymbol{b}_2 = \tfrac{1}{3}\left(4 - \cos(\varphi), -4\cot(\varphi) + \tfrac{4}{\sin(\varphi)} - \sin(\varphi)\right).$$

Izračunajte aproksimacije krožnih lokov po zgoraj opisanih interpolacijskih metodah pri izbiri $\varphi = \pi/6$. Primerjajte radialne napake $\max_{t\in\boldsymbol{t}} |1 - \|\boldsymbol{b}(t)\|\|$ za $\boldsymbol{t} = (i/1000)_{i=0}^{1000}$. Z rotacijami kontrolnih točk narišite aproksimacijo krožnice za vse tri primere.

```
n = 5;
fi = pi/n;
```

```
B1 = bezierarc(fi,1)
```

```
B1 = 3x2
   0.80901699  -0.58778525
   1.23606798            0
   0.80901699   0.58778525
```

```
B2 = bezierarc(fi,2)
```

```
B2 = 4x2
   0.80901699  -0.58778525
   1.00494541  -0.31811292
   1.00494541   0.31811292
   0.80901699   0.58778525
```

```
B3 = bezierarc(fi,3)
```

```
B3 = 4x2
   0.80901699  -0.58778525
   1.06366100  -0.23729784
   1.06366100   0.23729784
   0.80901699   0.58778525
```

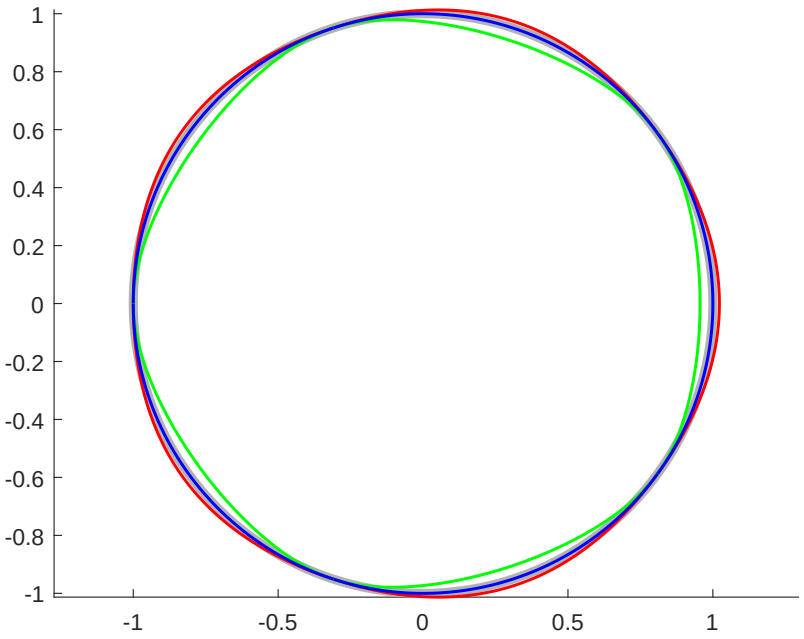
```
% izračun točk na Bezierjevih krivuljah
t = linspace(0,1,1001);
b1 = bezier(B1,t);
b2 = bezier(B2,t);
b3 = bezier(B3,t);
```

```
% radialne napake
[e1,e2,e3] = deal(0);
for i = 1:length(t)
    e1 = max(e1,abs(1-norm(b1(i,:))));
    e2 = max(e2,abs(1-norm(b2(i,:))));
    e3 = max(e3,abs(1-norm(b3(i,:))));
end
[e1,e2,e3]
```

```
ans = 1x3
   0.02254249   0.04403669   0.00007131
```

```
% rotacijska matrika
c = cos(2*fi);
s = sin(2*fi);
R = [c -s; s c];
```

```
clf
hold on
plot(cos(2*pi*t),sin(2*pi*t),'Color',0.7*[1 1 1],'LineWidth',4);

plot(b1(:,1),b1(:,2),'r');
plot(b2(:,1),b2(:,2),'g');
plot(b3(:,1),b3(:,2),'b');

for i = 1:n-1
    B1 = (R*B1')';
    B2 = (R*B2')';
    B3 = (R*B3')';

    b1 = bezier(B1,t);
    b2 = bezier(B2,t);
    b3 = bezier(B3,t);

    plot(b1(:,1),b1(:,2),'r');
    plot(b2(:,1),b2(:,2),'g');
    plot(b3(:,1),b3(:,2),'b');
end
hold off
axis equal
```



Aproksimacije krožnice z rotacijami aproksimacij krožnih lokov.

Naloga 5. *Subdivizija Bézierjeve krivulje.*

Subdivizija Bézierjeve krivulje $\boldsymbol{b}$ s kontrolnimi točkami $\boldsymbol{b}_i$, $i = 0, 1, \ldots, n$, pri parametru $t \in [0, 1]$ je proces delitve Bézierjeve krivulje na dve krajši,

$$\boldsymbol{b}(s) = \sum_{i=0}^{n} \boldsymbol{b}_0^i(t) B_i^n(\tfrac{s}{t}), \quad s \in [0, t], \qquad \boldsymbol{b}(s) = \sum_{i=0}^{n} \boldsymbol{b}_i^{n-i}(t) B_i^n(\tfrac{s-t}{1-t}), \quad s \in [t, 1],$$

ki skupaj opišeta prvotno Bézierjevo krivuljo in ju lahko predstavimo s pomočjo vmesnih točk de Casteljaujevega postopka za $\boldsymbol{b}$.

1. V Matlabu pripravite metodo `beziersub`, ki sprejme kontrolne točke Bézierjeve krivulje in parameter delitve ter s pomočjo funkcije `decasteljau` določi kontrolne točke dveh novih krivulj, ki opisujeta prvotno.

   Opis metode:

```
function BS = beziersub(B,t)
% Opis:
%  beziersub izvede subdivizijo Bezierjeve krivulje
%
% Definicija:
%  BS = beziersub(B,t)
%
% Vhodni podatki:
%  B    matrika kontrolnih točk Bezierjeve krivulje, v kateri
%       vsaka vrstica predstavlja eno kontrolno točko,
%  t    parameter subdivizije Bezierjeve krivulje
%
% Izhodni podatek:
%  BS   celica, ki vsebuje kontrolne točke dveh krivulj, ki jih
%       dobimo s subdivizijo prvotne Bezierjeve krivulje
end
```

   Primer:

```
B = [0 0; 2 3; 4 2; 5 -1];
BS = beziersub(B,0.4);
[BS{1} BS{2}]
```

```
ans = 4x4
         0         0  |  2.3360    1.8080
    0.8000    1.2000  |  3.4400    1.8800
    1.6000    1.7600  |  4.4000    0.8000
    2.3360    1.8080  |  5.0000   -1.0000
```

2. Razširite metodo `beziersub` z dodatnim parametrom k, ki določa, koliko zaporednih subdivizij želimo izvesti, vrne pa kontrolne točke končnih 2^k krivulj, ki v procesu nastanejo. Na primeru preverite, da lomljenka, ki povezuje kontrole točke vseh krivulj, z večanjem parametra k konvergira proti prvotni Bézierjevi krivulji.

   Primeri:

```
BS1 = beziersub(B,0.5,1);
[BS1{1} BS1{2}]
```

```
ans = 4x4
         0         0  |  2.8750    1.7500
    1.0000    1.5000  |  3.7500    1.5000
    2.0000    2.0000  |  4.5000    0.5000
    2.8750    1.7500  |  5.0000   -1.0000
```

```
BS2 = beziersub(B,0.5,2);
[BS2{1} BS2{2}; BS2{3} BS2{4}]
```

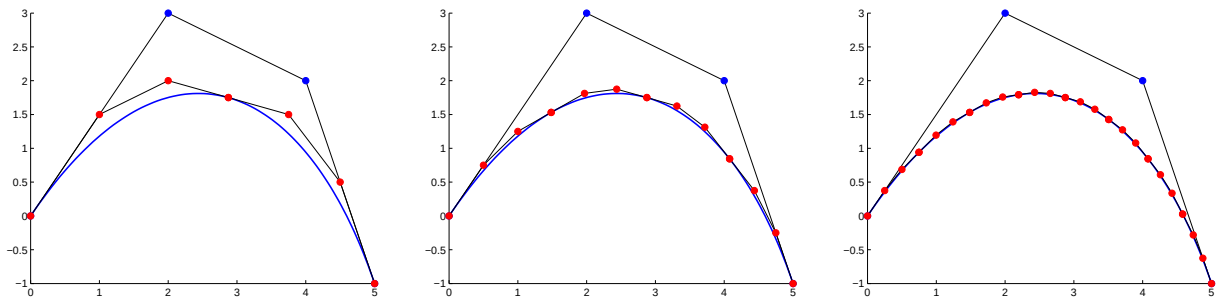
```
ans = 8x4
         0         0  |  1.4844    1.5312
    0.5000    0.7500  |  1.9688    1.8125
    1.0000    1.2500  |  2.4375    1.8750
    1.4844    1.5312  |  2.8750    1.7500
    -----------------------------------
    2.8750    1.7500  |  4.0781    0.8438
    3.3125    1.6250  |  4.4375    0.3750
    3.7188    1.3125  |  4.7500   -0.2500
    4.0781    0.8438  |  5.0000   -1.0000
```

```
BS3 = beziersub(B,0.5,3);
[BS3{1} BS3{2}; BS3{3} BS3{4}; BS3{5} BS3{6}; BS3{7} BS3{8}]
```

```
ans = 16x4
         0         0  |  0.7480    0.9414
    0.2500    0.3750  |  0.9961    1.1953
    0.5000    0.6875  |  1.2422    1.3906
    0.7480    0.9414  |  1.4844    1.5312
    -----------------------------------
    1.4844    1.5312  |  2.1973    1.7930
    1.7266    1.6719  |  2.4297    1.8281
    1.9648    1.7578  |  2.6562    1.8125
    2.1973    1.7930  |  2.8750    1.7500
    -----------------------------------
```

```
    2.8750    1.7500  |  3.5059    1.4258
    3.0938    1.6875  |  3.7070    1.2734
    3.3047    1.5781  |  3.8984    1.0781
    3.5059    1.4258  |  4.0781    0.8438
    ------------------------------------
    4.0781    0.8438  |  4.5801    0.0273
    4.2578    0.6094  |  4.7344   -0.2812
    4.4258    0.3359  |  4.8750   -0.6250
    4.5801    0.0273  |  5.0000   -1.0000
```

```
t = linspace(0,1);
subt = 0.5;
for k = 1:3
    subplot(1,3,k);
    plotbezier(B,t);
    BS = beziersub(B,subt,k);
    for d = 1:length(BS)
        plotbezier(BS{d},t);
    end
end
```



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Primeri subdivizije Bézierjeve krivulje na 2, 4 in 8 delov.

Naloga 6. *Višanje stopnje Bézierjeve krivulje.*

Naj bo $\boldsymbol{b}$ Bézierjeva krivulja stopnje n s kontrolnimi točkami $\boldsymbol{b}_i$, $i = 0, 1, \ldots, n$. Krivuljo $\boldsymbol{b}$ lahko reparametriziramo tako, da jo predstavimo kot Bézierjevo krivuljo stopnje $n+1$ s kontrolnimi točkami $\boldsymbol{b}_i^{(1)}$, $i = 0, 1, \ldots, n+1$. Te so podane z

$$\boldsymbol{b}_0^{(1)} = \boldsymbol{b}_0, \qquad \boldsymbol{b}_i^{(1)} = \left(1 - \frac{i}{n+1}\right)\boldsymbol{b}_i + \frac{i}{n+1}\boldsymbol{b}_{i-1}, \quad i = 1, 2, \ldots, n, \qquad \boldsymbol{b}_{n+1}^{(1)} = \boldsymbol{b}_n.$$

Če postopek ponavljamo, lahko prvotno krivuljo predstavimo kot Bézierjevo krivuljo stopnje m za poljuben $m > n$.

1. V Matlabu sestavite metodo `bezierelv`, ki sprejme kontrolne točke Bézierjeve krivulje stopnje n in naravno število k, vrne pa kontrolne točke, ki predstavljajo podano krivuljo kot Bézierjevo krivuljo stopnje $n + k$.

   Opis metode:

```
function Be = bezierelv(B,k)
% Opis:
%  bezierelv izvede višanje stopnje dane Bezierjeve krivulje
%
% Definicija:
%  Be = bezierelv(B,k)
%
% Vhodna podatka:
%  B    matrika velikosti (n+1) x d, v kateri vsaka vrstica
%       predstavlja d-dimenzionalno kontrolno točko Bezierjeve
%       krivulje stopnje n,
%  k    število, ki določa, za koliko želimo zvišati stopnjo
%       dane Bezierjeve krivulje
%
% Izhodni podatek:
%  Be   matrika velikosti (n+k+1) x d, v kateri vsaka vrstica
%       predstavlja d-dimenzionalno kontrolno točko Bezierjeve
%       krvulje stopnje n+k, ki ustreza dani Bezierjevi krivulji
end
```

2. Testirajte metodo z Bézierjevo krivuljo stopnje $n = 6$, ki jo določajo kontrolne točke

$$\boldsymbol{b}_0 = (0,0), \quad \boldsymbol{b}_1 = (2,6), \quad \boldsymbol{b}_2 = (3,0), \quad \boldsymbol{b}_3 = (5,4),$$
$$\boldsymbol{b}_4 = (7,1), \quad \boldsymbol{b}_5 = (5,5), \quad \boldsymbol{b}_6 = (10,6).$$

   Preverite, da z višanjem stopnje kontrolni poligoni novih Bézierjevih krivulj počasi konvergirajo k dani krivulji.

```
B = [0 0; 2 6; 3 0; 5 4; 7 1; 5 5; 10 6];
```

```
bezierelv(B,1)
```

```
ans = 8x2
         0         0
    1.7143    5.1429
    2.7143    1.7143
    4.1429    2.2857
    5.8571    2.7143
    6.4286    2.1429
    5.7143    5.1429
   10.0000    6.0000
```

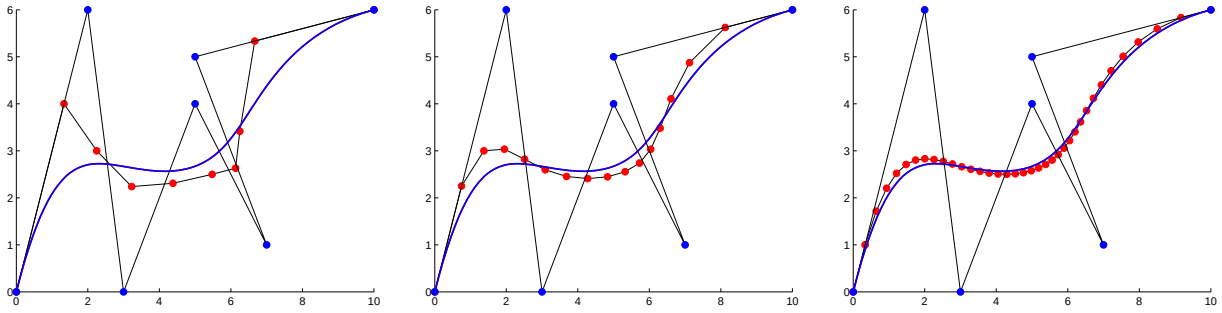
```
bezierelv(B,2)
```

```
ans = 9x2
         0         0
    1.5000    4.5000
    2.4643    2.5714
    3.6071    2.0714
    5.0000    2.5000
    6.0714    2.5000
    6.2500    2.8929
    6.2500    5.2500
   10.0000    6.0000
```

```
bezierelv(B,3)
```

```
ans = 10x2
         0         0
    1.3333    4.0000
    2.2500    3.0000
    3.2262    2.2381
    4.3810    2.3095
    5.4762    2.5000
    6.1310    2.6310
    6.2500    3.4167
    6.6667    5.3333
   10.0000    6.0000
```

```
t = linspace(0,1);
k = [3 10 30];
for j = 1:3
    Br = bezierelv(B,k(j));
    subplot(1,3,j);
    plotbezier(B,t);
    plotbezier(Br,t);
end
```



(a) $k = 3$ (b) $k = 10$ (c) $k = 30$

Primeri višanja stopnje Bézierjeve krivulje s stopnje 6 na stopnje 9, 16 in 36.

Naloga 7. *Parametrizacija sestavljene krivulje.*

Na obliko krivulje, ki jo sestavimo na podlagi točk $\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$, iz m kosov, parametriziranih nad delitvijo

$$u_0 < u_1 < \ldots < u_m,$$

vpliva izbira parametrov delitve. Ti navadno niso podani vnaprej, zato pri njihovi določitvi izhajamo iz znanih podatkov. Za izbran parameter $\alpha \in [0, 1]$ lahko vzamemo na primer

$$u_0 = 0, \qquad u_i = u_{i-1} + \|\boldsymbol{p}_i - \boldsymbol{p}_{i-1}\|^{\alpha}, \quad i = 1, 2, \ldots, m.$$

Če izberemo $\alpha = 0$, dobimo enakomerno parametrizacijo, ki je neodvisna od podatkov. Izbiri $\alpha = 1$ in $\alpha = 1/2$ vodita k tetivni in centripetalni parametrizaciji. V splošnem tovrstne izbire delilnih parametrov imenujemo α-parametrizacije.

1. V Matlabu sestavite metodo `alphaparam`, ki sprejme nabor točk in parameter α ter vrne delitev, ki določa parametrizacijo krivulje.

   Opis metode:

```
function u = alphaparam(P,a)
% Opis:
%  alphaparam sestavi alfa parametrizacijo oziroma delitev domene na
%  podlagi točk
%
% Definicija:
%  u = alphaparam(P,a)
%
% Vhodna podatka:
%  P    matrika z m+1 vrsticami, v kateri vsaka vrstica predstavlja
%       eno točko,
%  a    parameter, ki določa alfa parametrizacijo
%
% Izhodni podatek:
%  u    seznam parametrov delitve, ki so določeni rekurzivno tako, da
%       se trenutnemu parametru iz seznama u prišteje z a potencirana
%       norma razlike zaporednih točk iz seznama P
end
```

2. Izračunajte enakomerno ($\alpha = 0$), centripetalno ($\alpha = 1/2$) in tetivno ($\alpha = 1$) parametrizacijo za točke $(-5, 0)$, $(-2, -1)$, $(0, 3)$, $(3, 0)$, $(7, -1)$.

```
P = [-5 0; -2 -1; 0 3; 3 0; 7 -1];
```

```
ue = alphaparam(P,0)
```

```
ue = 1x5
     0     1     2     3     4
```

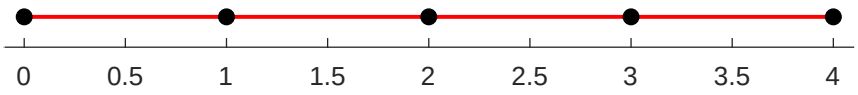
```
uc = alphaparam(P,0.5)
```

```
uc = 1x5
     0   1.77827941   3.89302194   5.95278908   7.98333227
```

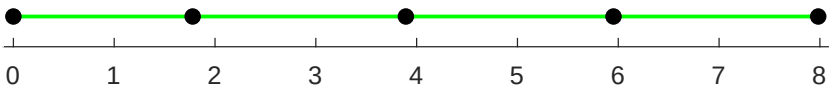
```
ut = alphaparam(P,1)
```

```
ut = 1x5
     0   3.16227766   7.63441362  11.87705430  16.00015993
```

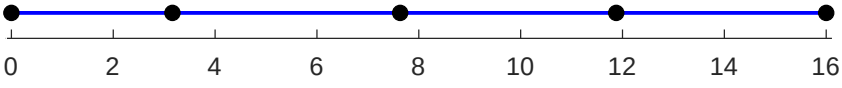
```
z = zeros(1,5);
plot(ue,z,'ro-','MarkerEdgeColor','k','MarkerFaceColor','k');
plot(uc,z,'go-','MarkerEdgeColor','k','MarkerFaceColor','k');
plot(ut,z,'bo-','MarkerEdgeColor','k','MarkerFaceColor','k');
```



(a) enakomerna parametrizacija ($\alpha = 0$)



(b) centripetalna parametrizacija ($\alpha = 1/2$)



(c) tetivna parametrizacija ($\alpha = 1$)

Primeri delilnih točk parametrizacije sestavljene krivulje.

Naloga 8. *Kubični C^2 zlepek.*

Bézierjeva krivulja stopnje 3, ki je sestavljena iz m kosov in je v stikih dvakrat zvezno odvedljiva, je določena z $m+3$ kontrolnimi točkami $\boldsymbol{d}_{-1}, \boldsymbol{d}_0, \ldots, \boldsymbol{d}_{m+1}$. Naj

$$\boldsymbol{b}^{(i)}(t) = \boldsymbol{b}_0^{(i)} B_0^3(t) + \boldsymbol{b}_1^{(i)} B_1^3(t) + \boldsymbol{b}_2^{(i)} B_2^3(t) + \boldsymbol{b}_3^{(i)} B_3^3(t)$$

predstavlja kos sestavjene krivulje z indeksom $i \in \{1, 2, \ldots, m\}$. Kontrolne točke krivulj so enolično določene s kontrolnimi točkami zlepka in parametrov delitve $u_0 < u_1 < \ldots < u_m$ na podlagi zahtev o redu gladkosti v stikih. Za $i \in \{1, 2, \ldots, m-2\}$ velja

$$\begin{aligned}
\boldsymbol{b}_1^{(i+1)} &= \frac{\Delta u_i + \Delta u_{i+1}}{\Delta u_{i-1} + \Delta u_i + \Delta u_{i+1}} \boldsymbol{d}_i + \frac{\Delta u_{i-1}}{\Delta u_{i-1} + \Delta u_i + \Delta u_{i+1}} \boldsymbol{d}_{i+1}, \\
\boldsymbol{b}_2^{(i+1)} &= \frac{\Delta u_{i+1}}{\Delta u_{i-1} + \Delta u_i + \Delta u_{i+1}} \boldsymbol{d}_i + \frac{\Delta u_{i-1} + \Delta u_i}{\Delta u_{i-1} + \Delta u_i + \Delta u_{i+1}} \boldsymbol{d}_{i+1},
\end{aligned}$$

kjer je $\Delta u_j = u_{j+1} - u_j$, $j \in \{0, 1, \ldots, m-1\}$. Za $i \in \{1, 2, \ldots, m-1\}$ pa velja

$$\boldsymbol{b}_3^{(i)} = \boldsymbol{b}_0^{(i+1)} = \frac{\Delta u_i}{\Delta u_{i-1} + \Delta u_i} \boldsymbol{b}_2^{(i)} + \frac{\Delta u_{i-1}}{\Delta u_{i-1} + \Delta u_i} \boldsymbol{b}_1^{(i+1)}.$$

Na robu vzamemo:

$$\begin{aligned}
&\boldsymbol{b}_0^{(1)} = \boldsymbol{d}_{-1}, \qquad \boldsymbol{b}_1^{(1)} = \boldsymbol{d}_0, \qquad \boldsymbol{b}_2^{(1)} = \frac{\Delta u_1}{\Delta u_0 + \Delta u_1} \boldsymbol{d}_0 + \frac{\Delta u_0}{\Delta u_0 + \Delta u_1} \boldsymbol{d}_1, \\
&\boldsymbol{b}_3^{(m)} = \boldsymbol{d}_{m+1}, \qquad \boldsymbol{b}_2^{(m)} = \boldsymbol{d}_m, \qquad \boldsymbol{b}_1^{(m)} = \frac{\Delta u_{m-1}}{\Delta u_{m-2} + \Delta u_{m-1}} \boldsymbol{d}_{m-1} + \frac{\Delta u_{m-2}}{\Delta u_{m-2} + \Delta u_{m-1}} \boldsymbol{d}_m.
\end{aligned}$$

1. V Matlabu sestavite metodo `beziercubspline`, ki sprejme delilne točke $u_0, u_1, \ldots, u_m$ in kontrolne točke $\boldsymbol{d}_{-1}, \boldsymbol{d}_0, \ldots, \boldsymbol{d}_{m+1}$, vrne pa seznam naborov kontrolnih točk

$$\left\{ \boldsymbol{b}_0^{(i)}, \boldsymbol{b}_1^{(i)}, \boldsymbol{b}_2^{(i)}, \boldsymbol{b}_3^{(i)} \right\}, \qquad i = 1, 2, \ldots, m,$$

ki predstavljajo posamezne kose sestavljene Bézierjeve krivulje.

Opis metode:

```
function B = beziercubspline(u,D)
% Opis:
%  beziercubspline izračuna sestavljeno Bezierjevo krivuljo stopnje 3,
%  ki je dvakrat zvezno odvedljiva v stikih
%
% Definicija:
%  B = beziercubspline(u,D)
%
% Vhodna podatka:
%  u    seznam parametrov delitve dolžine m+1,
%  D    matrika, v kateri vsaka izmed m+3 vrstic predstavlja eno
%       kontrolno točko sestavljene krivulje
%
```

```
% Izhodni podatek:
%  B    celični seznam dolžine m, v kateri je vsak element matrika
%       s štirimi vrsticami, ki določajo kontrolne točke kosa
%       sestavljene krivulje
end
```

2. Narišite primere sestavljenih Bézierjevih krivulj ($m = 4$) s kontrolnimi točkami

$$\boldsymbol{d}_{-1} = (-5, 0), \quad \boldsymbol{d}_0 = (-4, 1), \quad \boldsymbol{d}_1 = (-2, -1), \quad \boldsymbol{d}_2 = (0, 3),$$
$$\boldsymbol{d}_3 = (3, 0), \quad \boldsymbol{d}_4 = (5, 2), \quad \boldsymbol{d}_5 = (7, -1)$$

za različne α-parametrizacije, ki so določene na podlagi točk $\boldsymbol{d}_{-1}$, $\boldsymbol{d}_1$, $\boldsymbol{d}_2$, $\boldsymbol{d}_3$, $\boldsymbol{d}_5$.

Primeri:

```
D = [-5 0; -4 1; -2 -1; 0 3; 3 0; 5 2; 7 -1];
P = D([1 3 4 5 7],:);
```

```
ue = alphaparam(P,0);
Be = beziercubspline(ue,D);
[Be{1} Be{2}; Be{3} Be{4}]
```

```
ans = 8x4
  -5.00000000            0  -2.16666667   0.16666667
  -4.00000000   1.00000000  -1.33333333   0.33333333
  -3.00000000            0  -0.66666667   1.66666667
  -2.16666667   0.16666667   0.16666667   1.83333333
   0.16666667   1.83333333   3.00000000   1.00000000
   1.00000000   2.00000000   4.00000000   1.00000000
   2.00000000   1.00000000   5.00000000   2.00000000
   3.00000000   1.00000000   7.00000000  -1.00000000
```

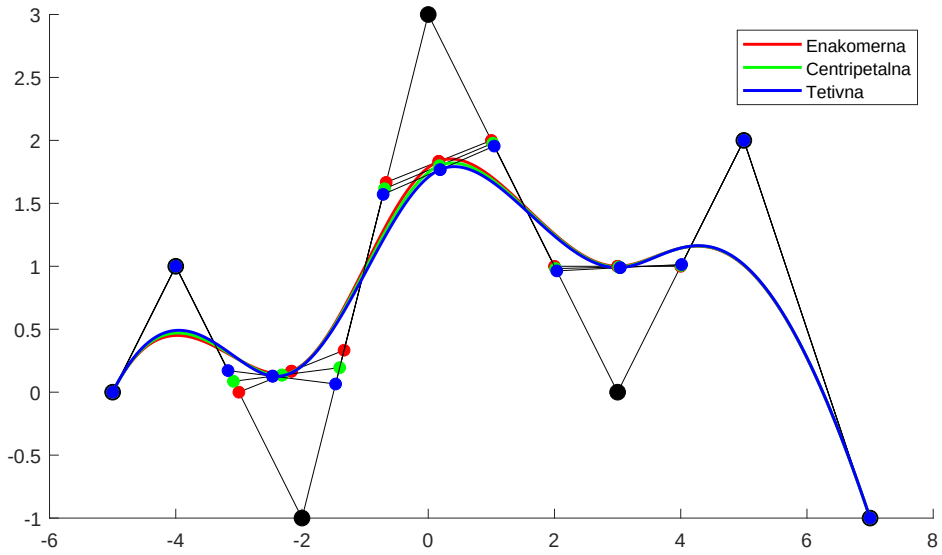
```
uc = alphaparam(P,0.5);
Bc = beziercubspline(uc,D);
[Bc{1} Bc{2}; Bc{3} Bc{4}]
```

```
ans = 8x4
  -5.00000000            0  -2.31725006   0.13598609
  -4.00000000   1.00000000  -1.40253908   0.19492183
  -3.08642723   0.08642723  -0.69203431   1.61593138
  -2.31725006   0.13598609   0.17648662   1.79913233
   0.17648662   1.79913233   3.01981684   0.99452357
   1.02242926   1.97757074   4.00714468   1.00714468
   2.01827918   0.98172082   5.00000000   2.00000000
   3.01981684   0.99452357   7.00000000  -1.00000000
```

```
ut = alphaparam(P,1);
Bt = beziercubspline(ut,D);
[Bt{1} Bt{2}; Bt{3} Bt{4}]
```

```
ans = 8x4
   -5.00000000            0   -2.46572195    0.12743061
   -4.00000000    1.00000000   -1.46749799    0.06500403
   -3.17157288    0.17157288   -0.71442642    1.57114716
   -2.46572195    0.12743061    0.18848601    1.76809497
    0.18848601    1.76809497    3.03952360    0.98925783
    1.04506394    1.95493606    4.01428863    1.01428863
    2.03649865    0.96350135    5.00000000    2.00000000
    3.03952360    0.98925783    7.00000000   -1.00000000
```

```
t = linspace(0,1);
clf
hold on
plot(D(:,1),D(:,2),'ko-','MarkerFaceColor','k','MarkerSize',8);
for i = 1:4
    s1 = plotbezier(Be{i},t,'r');
    s2 = plotbezier(Bc{i},t,'g');
    s3 = plotbezier(Bt{i},t,'b');
end
legend([s1 s2 s3], 'Enakomerna', 'Centripetalna', 'Tetivna');
hold off
```



Primeri kubičnih C^2 zlepkov za različne parametrizacije.

Naloga 9. *Racionalne Bézierjeve krivulje in Farinove točke.*

Racionalna Bézierjeva krivulja $\boldsymbol{b}$ stopnje n je podana s parametrizacijo

$$\boldsymbol{b}(t) = \frac{\sum_{i=0}^{n} w_i \boldsymbol{b}_i B_i^n(t)}{\sum_{i=0}^{n} w_i B_i^n(t)}, \qquad t \in [0,1].$$

Točke $\boldsymbol{b}_i$ predstavljajo kontrolne točke krivulje $\boldsymbol{b}$, parametre w_i, za katere navadno zahtevamo, da so pozitivni, pa imenujemo uteži. S povečanjem specifične uteži dosežemo, da se krivulja bolj približa kontrolni točki z enakim indeksom.

1. V Matlabu imlementirajte metodo `rbezier`, ki izračuna točke na racionalni Bézierjevi krivulji. Temelji naj na prirejenem de Casteljaujevem postopku, pri katerem vmesno točko $\boldsymbol{b}_i^r(t)$, $i = 0, 1, \ldots, n-r$, v koraku r, $r = 1, \ldots, n$, izračunamo s popravkom uteži

   $$w_i^r(t) = (1-t)w_i^{r-1}(t) + t w_{i+1}^{r-1}(t)$$

   kot

   $$\boldsymbol{b}_i^r(t) = (1-t)\,\frac{w_i^{r-1}(t)}{w_i^r(t)}\boldsymbol{b}_i^{r-1}(t) + t\,\frac{w_{i+1}^{r-1}(t)}{w_i^r(t)}\boldsymbol{b}_{i+1}^{r-1}(t).$$

   Na začetku postavimo $\boldsymbol{b}_i^0(t) = \boldsymbol{b}_i$ in $w_i^0(t) = w_i$, $i = 0, 1, \ldots, n$, končna točka $\boldsymbol{b}_0^n(t)$ pa predstavlja točko na krivulji $\boldsymbol{b}$ pri parametru t.

   Opis metode:

```
function b = rbezier(B,w,t)
% Opis:
%  rbezier vrne točke na racionalni Bezierovi krivulji, izračunane z
%  de Casteljaujevim postopkom za racionalne krivulje
%
% Definicija:
%  b = rbezier(B,w,t)
%
% Vhod:
%  B    matrika velikosti n+1 x d, v kateri vsaka vrstica predstavlja
%       eno kontrolno točko racionalne Bezierjeve krivulje stopnje n
%       v prostoru dimenzije d,
%  w    seznam uteži racionalne Bezierjeve krivulje,
%  t    seznam parametrov dolžine N, za katere se računajo točke na
%       racionalni Bezierjevi krivulji
%
% Izhod:
%  b    matrika velikosti N x d, v kateri i-ta vrstica predstavlja
%       točko na racionalni Bezierjevi krivulji pri i-tem parametru
%       iz seznama t
end
```

Primeri:

```
B = [0 0; 2 3; 5 0; 3 -1; 2 -1];
w = [0.5; 0.7; 0.8; 1; 0.6];
rbezier(B,w,linspace(0,1,10))
```

```
ans = 10x2
            0            0
   1.17768357   1.11692050
   2.15024327   1.28705450
   2.85223368   1.00687285
   3.26626004   0.54606107
   3.41079150   0.05884565
   3.32436472  -0.37070254
   3.04918529  -0.70209066
   2.61389251  -0.91784023
   2.00000000  -1.00000000
```

2. Metoda `plotrbezier` naj na podlagi kontrolnih točk in uteži ter seznama parametrov, ki določajo točke na pripadajoči krivulji, s pomočjo metode `rbezier` nariše krivuljo in njen kontrolni poligon. Na kontrolnem poligonu naj označi tudi tako imenovane Farinove točke, ki so določene na podlagi kontrolnih točk in uteži kot

$$\boldsymbol{q}_i = \frac{w_i}{w_i + w_{i+1}}\boldsymbol{b}_i + \frac{w_{i+1}}{w_i + w_{i+1}}\boldsymbol{b}_{i+1}, \qquad i = 0, 1, \ldots, n-1.$$

Na primeru si oglejte, kako se spreminja lokacija točk $\boldsymbol{q}_i$, ko določeno utež povečujete ali zmanjšujete.

Primeri:

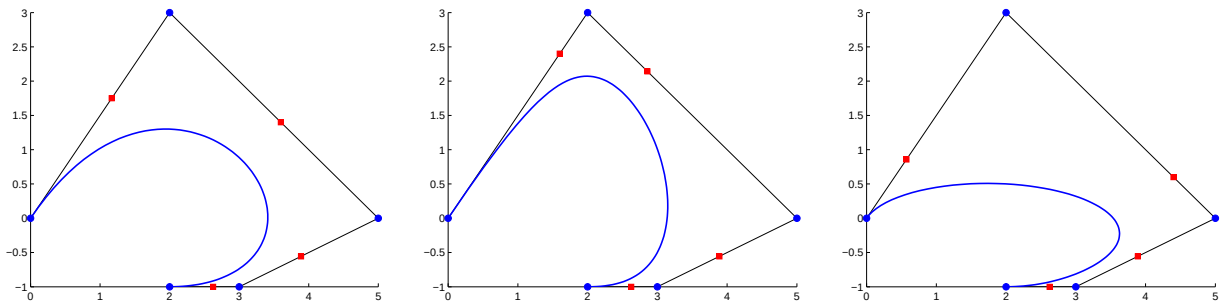
```
w1 = w; w1(2) = 2;
rbezier(B,w1,linspace(0,1,10))
```

```
ans = 10x2
            0            0
   1.51534594   1.89015701
   2.08208167   2.06417480
   2.49699399   1.83767535
   2.83680098   1.37832796
   3.07460977   0.75970213
   3.14618370   0.08279431
   2.99444584  -0.50894041
   2.60865643  -0.88442376
   2.00000000  -1.00000000
```

```
w2 = w; w2(2) = 0.2;
rbezier(B,w2,linspace(0,1,10))
```

```
ans = 10x2
            0            0
   0.87667119   0.42761135
   2.22074768   0.48322331
   3.17535545   0.25118483
   3.57767886  -0.05745063
   3.60376149  -0.34344946
   3.40858506  -0.58505564
   3.07187826  -0.78216369
   2.61593045  -0.93084632
   2.00000000  -1.00000000
```

```
t = linspace(0,1);
plotrbezier(B,w,t);
plotrbezier(B,w1,t);
plotrbezier(B,w2,t);
```



(a) $w_1 = 0.7$ (b) $w_1 = 2$ (c) $w_1 = 0.2$

Primeri racionalnih Bézierjevih krivulj pri različnih vrednostih uteži w_1.

Naloga 10. *Višanje stopnje racionalne Bézierjeve krivulje.*

Stopnjo racionalne Bézierjeve krivulje s kontrolnimi točkami $\boldsymbol{b}_i \in \mathbb{R}^d$ in utežmi $w_i \in \mathbb{R}$, $i = 0, 1, \ldots, n$, lahko zvišamo tako, da jo razumemo kot polinomsko Bézierjevo krivuljo s kontrolnimi točkami $(w_i\boldsymbol{b}_i, w_i) \in \mathbb{R}^{d+1}$. Najprej uporabimo znani postopek za višanje stopnje polinomske krivulje, nato pa dobljene kontrolne točke iz $\mathbb{R}^{d+1}$ projeciramo nazaj v $\mathbb{R}^d$. To napravimo tako, da zadnjo komponento proglasimo za novo utež $w_i^{(1)}$, $i = 0, 1, \ldots, n+1$, preostali del kontrolne točke pa z njo delimo in s tem izračunamo $\boldsymbol{b}_i^{(1)}$.

1. Pripravite metodo `rbezierelv`, ki izvede postopek višanja stopnje. Preizkusite jo na racionalni krivulji stopnje 5, podani s kontrolnimi točkami

$$\boldsymbol{b}_0 = (1,0), \quad \boldsymbol{b}_1 = (1,4), \quad \boldsymbol{b}_2 = (-3,2), \quad \boldsymbol{b}_3 = (-3,-2), \quad \boldsymbol{b}_4 = (1,-4), \quad \boldsymbol{b}_5 = (1,0)$$

in utežmi

$$w_0 = 1, \quad w_1 = \frac{1}{5}, \quad w_2 = \frac{1}{5}, \quad w_3 = \frac{1}{5}, \quad w_4 = \frac{1}{5}, \quad w_5 = 1,$$

ki predstavlja enotsko krožnico.

Opis metode:

```
function [Be,we] = rbezierelv(B,w)
% Opis:
%  rbezierelv izvede višanje stopnje racionalne Bezierjeve krivulje
%
% Definicija:
%  [Be,we] = rbezierelv(B,w)
%
% Vhodna podatka:
%  B    matrika velikosti n+1 x d, v kateri vsaka vrstica predstavlja
%       d-dimenzionalno kontrolno točko racionalne Bezierjeve krivulje
%       stopnje n,
%  w    seznam uteži racionalne Bezierjeve krivulje
%
% Izhodni podatek:
%  Be   matrika velikosti n+2 x d, v kateri vsaka vrstica predstavlja
%       d-dimenzionalno kontrolno točko racionalne Bezierjeve krvulje
%       stopnje n+1, ki ustreza dani racionalni Bezierjevi krivulji,
%  we   seznam dolžine n+2, v katerem vsak element predstavlja utež
%       racionalne Bezierjeve krvulje stopnje n+1, ki ustreza dani
%       racionalni Bezierjevi krivulji
end
```

Primeri:

```
B = [1 0; 1 4; -3 2; -3 -2; 1 -4; 1 0];
w = [1; 1/5; 1/5; 1/5; 1/5; 1];
```

```
[B1,w1] = rbezierelv(B,w)
```

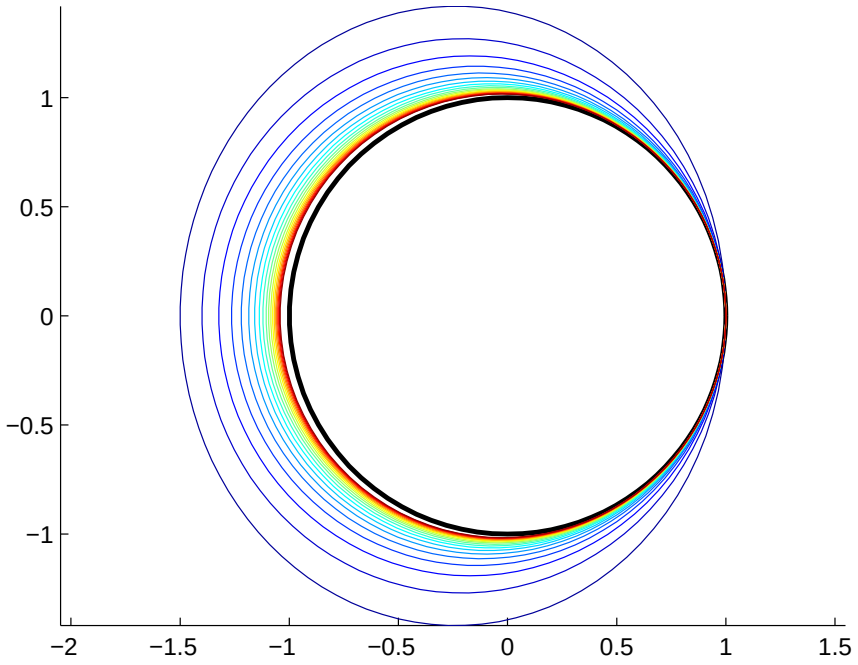
```
B1 = 7x2
   1.00000000            0
   1.00000000   2.00000000
  -1.66666667   2.66666667
  -3.00000000            0
  -1.66666667  -2.66666667
   1.00000000  -2.00000000
   1.00000000            0

w1 = 7x1
   1.00000000
   0.33333333
   0.20000000
   0.20000000
   0.20000000
   0.33333333
   1.00000000
```

```
[B2,w2] = rbezierelv(B1,w1)
```

```
B2 = 8x2
   1.00000000            0
   1.00000000   1.33333333
  -0.60000000   2.40000000
  -2.42857143   1.14285714
  -2.42857143  -1.14285714
  -0.60000000  -2.40000000
   1.00000000  -1.33333333
   1.00000000            0

w2 = 8x1
   1.00000000
   0.42857143
   0.23809524
   0.20000000
   0.20000000
   0.23809524
   0.42857143
   1.00000000
```

2. Preizkusite preprost postopek za aproksimacijo racionalne Bézierjeve krivulje s polinomskimi, ki so določene s kontrolnimi točkami racionalnih krivulj, dobljenih z višanjem stopnje prvotne racionalne krivulje. Uspešnost postopka preverite na parametrizaciji krožnice z racionalno krivuljo stopnje 5.

```
t = linspace(0,1);
b = rbezier(B,w,t);
N = 20;

clf
hold on

plot(b(:,1),b(:,2),'k','LineWidth',2);
C = jet(N);
for i = 1:N
    [B,w] = rbezierelv(B,w);
    b = bezier(B,t);
    plot(b(:,1),b(:,2),'Color',C(i,:));
end

hold off
axis equal
```



Dvajset aproksimacij krožnice s polinomskimi krivuljami, dobljenih z višanjem stopnje.

Naloga 11. *Bézierjeve ploskve iz tenzorskega produkta.*

Bézierjeva ploskev iz tenzorskega produkta stopnje $(m, n) \in \mathbb{N} \times \mathbb{N}$ je podana s parametrizacijo

$$\boldsymbol{b}(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \boldsymbol{b}_{i,j} B_i^m(u) B_j^n(v), \qquad (u, v) \in [0, 1] \times [0, 1].$$

Pri fiksnem parametru v množica $\{\boldsymbol{b}(u, v);\ u \in [0, 1]\}$ predstavlja Bézierjevo krivuljo stopnje m s kontrolnimi točkami

$$\sum_{j=0}^{n} \boldsymbol{b}_{i,j} B_j^n(v), \qquad i = 0, 1, \ldots, m,$$

ki jih lahko izračuamo kot točke na Bézierjevih krivuljah stopnje n pri parametru v. To pomeni, da lahko točko na Bézierjevi ploskvi pri parametru (u, v) dobimo z izvedbo $m + 2$ običajnih de Casteljaujevih postopkov.

1. V Matlabu pripravite metodo `bezier2`, ki sprejme kontrolne točke $\boldsymbol{b}_{i,j}$ ter seznama parametrov $\boldsymbol{u}$ in $\boldsymbol{v}$, vrne pa točke $\boldsymbol{b}(u, v)$, $(u, v) \in \boldsymbol{u} \times \boldsymbol{v}$, na Bézierjevi ploskvi. Izračunajte točke na ploskvi, ki je podana s kontrolnimi točkami

$$\begin{array}{llll} \boldsymbol{b}_{0,0} = (0, -1, -3), & \boldsymbol{b}_{1,0} = (3, -2, -2), & \boldsymbol{b}_{2,0} = (5, -1, 0), & \boldsymbol{b}_{3,0} = (6, -2, -5), \\ \boldsymbol{b}_{0,1} = (1, 1, 3), & \boldsymbol{b}_{1,1} = (4, 2, 6), & \boldsymbol{b}_{2,1} = (5, 1, 3), & \boldsymbol{b}_{3,1} = (7, 2, 2), \\ \boldsymbol{b}_{0,2} = (0, 5, -1), & \boldsymbol{b}_{1,2} = (3, 5, 4), & \boldsymbol{b}_{2,2} = (5, 6, -2), & \boldsymbol{b}_{3,2} = (6, 5, 8). \end{array}$$

   Opis metode:

```
function [bx,by,bz] = bezier2(Bx,By,Bz,u,v)
% Opis:
%  bezier2 vrne točke na Bezierjevi ploskvi iz tenzorskega produkta
%
% Definicija:
%  [bx,by,bz] = bezier2(Bx,By,Bz,u,v)
%
% Vhodni podatki:
%  Bx, By, Bz   matrike velikosti n+1 x m+1, ki predstavljajo
%               koordinate kontrolnih točk,
%  u, v         vrstici dolžine M in N, ki predstavljata parametre
%               v smereh u in v
%
% Izhodni podatki:
%  bx, by, bz   matrike velikosti N x M, ki predstavljajo točke na
%               Bezierjevi ploskvi: [bx(J,I) by(J,I) bz(J,I)] je
%               točka pri parametrih u(I) in v(J)
end
```

Primer:

```
% kontrolne točke Bezierjeve ploskve
Bx = [0 3 5 6; 1 4 5 7; 0 3 5 6];
By = [-1 -2 -1 -2; 1 2 1 2; 5 5 6 5];
Bz = [-3 -2 0 -5; 3 6 3 2; -1 4 -2 8];
```

```
% izračun točk na ploskvi nad mrežo parametrov velikosti 7 x 7
[u,v] = deal(linspace(0,1,7));
[bx,by,bz] = bezier2(Bx,By,Bz,u,v)
```

```
bx = 7x7
         0    1.4167    2.6667    3.7500    4.6667    5.4167    6.0000
    0.2778    1.6752    2.8827    3.9236    4.8210    5.5980    6.2778
    0.4444    1.8302    3.0123    4.0278    4.9136    5.7068    6.4444
    0.5000    1.8819    3.0556    4.0625    4.9444    5.7431    6.5000
    0.4444    1.8302    3.0123    4.0278    4.9136    5.7068    6.4444
    0.2778    1.6752    2.8827    3.9236    4.8210    5.5980    6.2778
         0    1.4167    2.6667    3.7500    4.6667    5.4167    6.0000

by = 7x7
   -1.0000   -1.3519   -1.4815   -1.5000   -1.5185   -1.6481   -2.0000
   -0.2778   -0.4225   -0.4722   -0.4757   -0.4815   -0.5382   -0.6944
    0.5556    0.5633    0.5802    0.5972    0.6049    0.5941    0.5556
    1.5000    1.6053    1.6759    1.7188    1.7407    1.7488    1.7500
    2.5556    2.7037    2.8148    2.8889    2.9259    2.9259    2.8889
    3.7222    3.8584    3.9969    4.1076    4.1605    4.1254    3.9722
    5.0000    5.0694    5.2222    5.3750    5.4444    5.3472    5.0000

bz = 7x7
   -3.0000   -2.4537   -1.9630   -1.7500   -2.0370   -3.0463   -5.0000
   -1.2778   -0.5629   -0.1327   -0.0590   -0.4136   -1.2681   -2.6944
   -0.1111    0.7824    1.1852    1.1806    0.8519    0.2824   -0.4444
    0.5000    1.5822    1.9907    1.9688    1.7593    1.6053    1.7500
    0.5556    1.8364    2.2840    2.3056    2.3086    2.7006    3.8889
    0.0556    1.5451    2.0648    2.1910    2.5000    3.5683    5.9722
   -1.0000    0.7083    1.3333    1.6250    2.3333    4.2083    8.0000
```

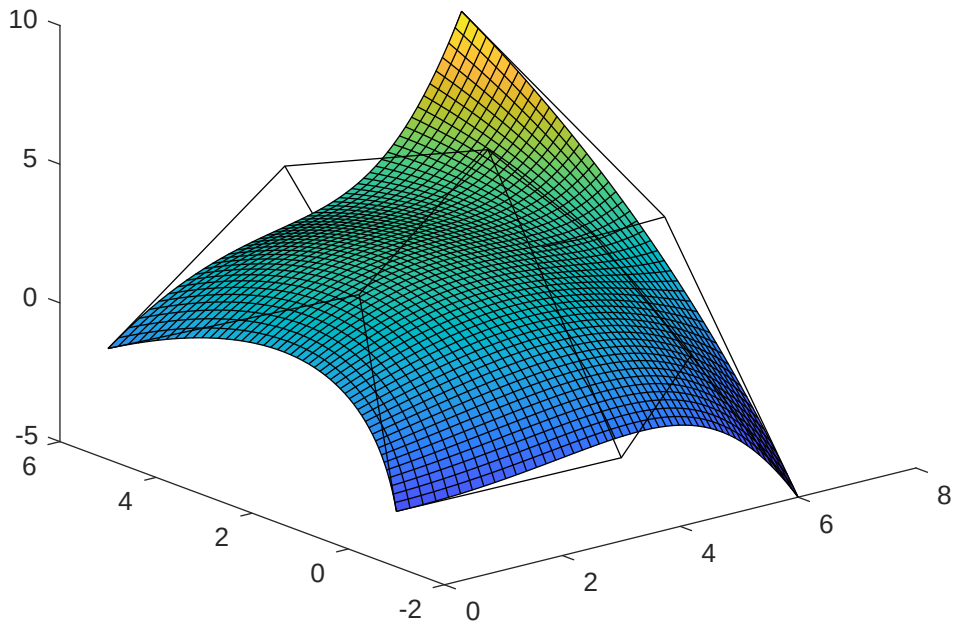
2. Pripravite metodo `plotbezier2`, ki za podane kontrolne točke nariše Bézierjevo ploskev in pripadajočo kontrolno mrežo. Pomagajte si z vgrajenima funkcijama `mesh` in `surf`. Narišite primer Bézierjeve ploskve.

Opis metode:

```
function p = plotbezier2(Bx,By,Bz,u,v)
% Opis:
%  plotbezier nariše Bezierjevo ploskev iz tenzorskega produkta
%
% Definicija:
%  p = plotbezier2(Bx,By,Bz,u,v)
%
% Vhodni podatki:
%  Bx, By, Bz   matrike velikosti n+1 x m+1, ki predstavljajo
%               koordinate kontrolnih točk,
%  u, v         vrstici, ki predstavljata parametre v smereh u in v
%
% Izhodni podatek:
%  p    grafični objekt, ki določa ploskev
end
```

Primer:

```
[u,v] = deal(linspace(0,1,50));
plotbezier2(Bx,By,Bz,u,v);
```



Primer Bézierjeve ploskve iz tenzorskega produkta stopnje $(3, 2)$.

Naloga 12. *Coonsove ploskve.*

Coonsove ploskve lahko interpretiramo kot Bézierjeve ploskve, ki so določene s štirimi robnimi krivuljami. Uporabne so, kadar želimo konstruirati ploskev, a imamo podan le njen okvir. Kontrolne točke

$$\begin{array}{ccccc} \boldsymbol{b}_{0,0} & \boldsymbol{b}_{1,0} & \dots & \boldsymbol{b}_{m-1,0} & \boldsymbol{b}_{m,0} \\ \boldsymbol{b}_{0,1} & & & & \boldsymbol{b}_{m,1} \\ \vdots & & & & \vdots \\ \boldsymbol{b}_{0,n-1} & & & & \boldsymbol{b}_{m,n-1} \\ \boldsymbol{b}_{0,n} & \boldsymbol{b}_{1,n} & \dots & \boldsymbol{b}_{m-1,n} & \boldsymbol{b}_{m,n} \end{array}$$

določajo Bézierjeve krivulje

$$u \mapsto \sum_{i=0}^{m} \boldsymbol{b}_{i,0} B_i^m(u), \quad u \mapsto \sum_{i=0}^{m} \boldsymbol{b}_{i,n} B_i^m(u), \quad v \mapsto \sum_{j=0}^{n} \boldsymbol{b}_{0,j} B_j^n(v), \quad v \mapsto \sum_{j=0}^{n} \boldsymbol{b}_{m,j} B_j^n(v),$$

ki omejujejo iskano Bézierjevo ploskev $\boldsymbol{b}$ iz tenzorskega produkta stopnje (m, n) s kontrolnimi točkami $\boldsymbol{b}_{i,j}$, $i = 0, 1, \dots, m$, $j = 0, 1, \dots, n$. Definiramo jo s pomočjo treh premonosnih ploskev.

- Prva je Bézierjeva ploskev stopnje $(m, 1)$, ki je kot ploskev stopnje (m, n) podana s kontrolnimi točkami
$$\boldsymbol{b}_{i,j}^{(1)} = \left(1 - \frac{j}{n}\right) \boldsymbol{b}_{i,0} + \frac{j}{n} \boldsymbol{b}_{i,n}.$$
- Druga je Bézierjeva ploskev stopnje $(1, n)$, ki je kot ploskev stopnje (m, n) podana s kontrolnimi točkami
$$\boldsymbol{b}_{i,j}^{(2)} = \left(1 - \frac{i}{m}\right) \boldsymbol{b}_{0,j} + \frac{i}{m} \boldsymbol{b}_{m,j}.$$
- Tretja je Bézierjeva ploskev stopnje $(1, 1)$, ki je kot ploskev stopnje (m, n) podana s kontrolnimi točkami
$$\boldsymbol{b}_{i,j}^{(3)} = \left(1 - \frac{i}{m}\right)\left(1 - \frac{j}{n}\right) \boldsymbol{b}_{0,0} + \left(1 - \frac{i}{m}\right) \frac{j}{n} \boldsymbol{b}_{0,n} + \frac{i}{m}\left(1 - \frac{j}{n}\right) \boldsymbol{b}_{m,0} + \frac{i}{m}\frac{j}{n} \boldsymbol{b}_{m,n}.$$

Coonsovo ploskev $\boldsymbol{b}$ definiramo s kontrolnimi točkami

$$\boldsymbol{b}_{i,j} = \boldsymbol{b}_{i,j}^{(1)} + \boldsymbol{b}_{i,j}^{(2)} - \boldsymbol{b}_{i,j}^{(3)}.$$

Izračunati je treba le točke z indeksi $1 \le i \le m-1$ in $1 \le j \le n-1$, saj se robne kontrolne točke po konstrukciji ujemajo s podanimi.

1. V Matlabu pripravite metodo `coons`, ki sprejme kontrolne točke $\boldsymbol{b}_{i,j}$ z indeksi
$$(i, j) \in (\{0, 1, \dots, m\} \times \{0, n\}) \cup (\{0, m\} \times \{0, 1, \dots, n\}),$$
vrne pa kontrolne točke Coonsove ploskve $\boldsymbol{b}_{i,j}$, $(i, j) \in \{0, 1, \dots, m\} \times \{0, 1, \dots, n\}$.

Opis metode:

```
function [Bx,By,Bz] = coons(Bx,By,Bz)
% Opis:
%  coons vrne kontrolne točke Coonsove ploskve
%
% Definicija:
%  [Bx,By,Bz] = coons(Bx,By,Bz)
%
% Vhodni podatki:
%  Bx, By, Bz   matrike velikosti n+1 x m+1, ki določajo koordinate
%               robnih kontrolnih točk (v konstrukciji Coonsove
%               ploskve se upoštevajo kontrolne točke, ki jih določa
%               prva in zadnja vrstica ter prvi in zadnji stolpec
%               posamezne matrike)
%
% Izhodni podatki:
%  Bx, By, Bz   matrike velikosti n+1 x m+1, ki določajo koordinate
%               kontrolnih točk Coonsove ploskve
end
```

2. Narišite Coonsovo ploskev stopnje $(6, 8)$, ki je določena s kontrolnimi točkami

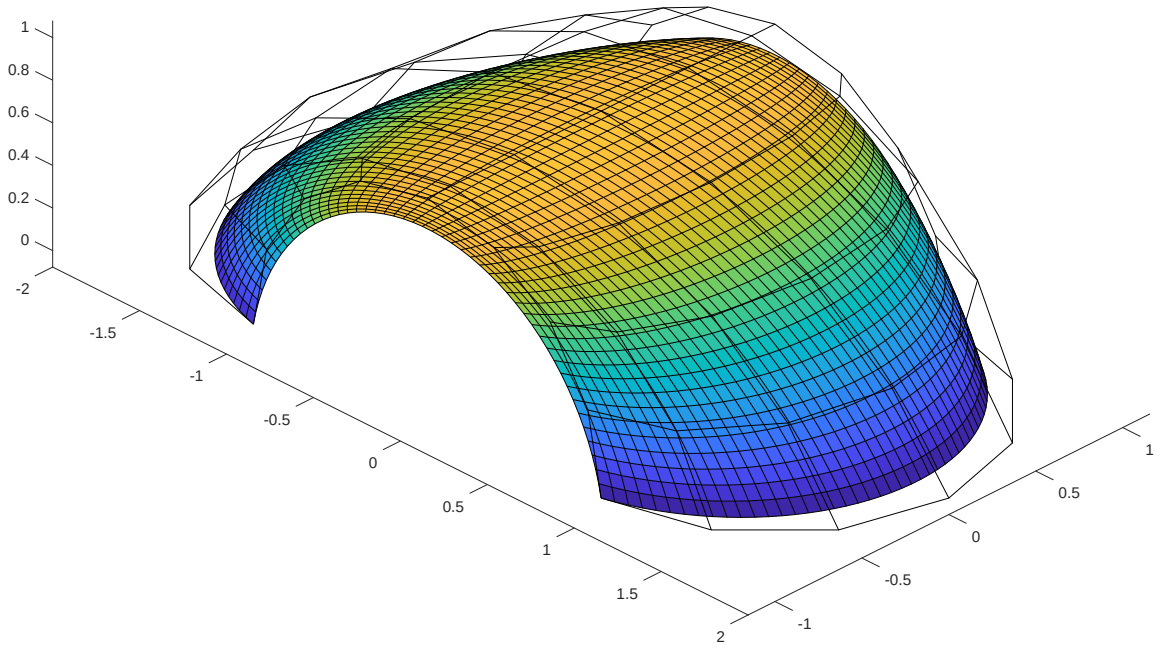
$$\begin{aligned}
\boldsymbol{b}_{i,0} &= \left(1+\sin(\tfrac{\pi}{6}i), -\cos(\tfrac{\pi}{6}i), 0\right), && i = 0,1,\ldots,6,\\
\boldsymbol{b}_{i,8} &= \left(-1-\sin(\tfrac{\pi}{6}i), -\cos(\tfrac{\pi}{6}i), 0\right), && i = 0,1,\ldots,6,\\
\boldsymbol{b}_{0,j} &= \left(\cos(\tfrac{\pi}{8}j), -1, \sin(\tfrac{\pi}{8}j)\right), && j = 0,1,\ldots,8,\\
\boldsymbol{b}_{6,j} &= \left(\cos(\tfrac{\pi}{8}j), 1, \sin(\tfrac{\pi}{8}j)\right), && j = 0,1,\ldots,8.
\end{aligned}$$

```
% kontrole točke robnih krivulj
[Bx,By,Bz] = deal(zeros(9,7));

phi = linspace(0,pi,7);
psi = linspace(0,pi,9);

Bx(1,:) = 1+sin(phi);
By(1,:) = -cos(phi);

Bx(9,:) = -1-sin(phi);
By(9,:) = -cos(phi);

Bx(:,1) = cos(psi);
By(:,1) = -1;
```

```
Bz(:,1) = sin(psi);
Bx(:,7) = cos(psi);

By(:,7) = 1;
Bz(:,7) = sin(psi);

% kontrolne točke Coonsove ploskve
[Bx,By,Bz] = coons(Bx,By,Bz)
```

```
Bx = 9x7
    1.0000    1.5000    1.8660    2.0000    1.8660    1.5000    1.0000
    0.9239    1.2989    1.5734    1.6739    1.5734    1.2989    0.9239
    0.7071    0.9571    1.1401    1.2071    1.1401    0.9571    0.7071
    0.3827    0.5077    0.5992    0.6327    0.5992    0.5077    0.3827
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
   -0.3827   -0.5077   -0.5992   -0.6327   -0.5992   -0.5077   -0.3827
   -0.7071   -0.9571   -1.1401   -1.2071   -1.1401   -0.9571   -0.7071
   -0.9239   -1.2989   -1.5734   -1.6739   -1.5734   -1.2989   -0.9239
   -1.0000   -1.5000   -1.8660   -2.0000   -1.8660   -1.5000   -1.0000

By = 9x7
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000
   -1.0000   -0.8660   -0.5000   -0.0000    0.5000    0.8660    1.0000

Bz = 9x7
         0         0         0         0         0         0         0
    0.3827    0.3827    0.3827    0.3827    0.3827    0.3827    0.3827
    0.7071    0.7071    0.7071    0.7071    0.7071    0.7071    0.7071
    0.9239    0.9239    0.9239    0.9239    0.9239    0.9239    0.9239
    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
    0.9239    0.9239    0.9239    0.9239    0.9239    0.9239    0.9239
    0.7071    0.7071    0.7071    0.7071    0.7071    0.7071    0.7071
    0.3827    0.3827    0.3827    0.3827    0.3827    0.3827    0.3827
    0.0000         0         0         0         0         0    0.0000
```

```
[u,v] = deal(linspace(0,1,50));
plotbezier2(Bx,By,Bz,u,v);
```



Primer Coonsove ploskve stopnje $(6, 8)$.

Naloga 13. *Aproksimacija z Bézierjevimi ploskvami po metodi najmanjših kvadratov.*

Za vrednosti $f_k \in \mathbb{R}$, $k = 1, 2, \ldots, K$, prirejene parametrom $(u_k, v_k) \in [0,1] \times [0,1]$ iščemo polinom

$$b(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{i,j} B_i^m(u) B_j^n(v), \quad (u,v) \in [0,1] \times [0,1]$$

stopnje (m,n) s koeficienti $b_{i,j}$, ki minimizira vrednost

$$\sum_{k=1}^{K} \left(f_k - b(u_k, v_k)\right)^2.$$

Tak polinom predstavlja najboljšo aproksimacijo za dane podatke po metodi najmanjših kvadratov. Določimo ga z reševanjem predoločenega sistema

$$\begin{bmatrix} B_0^m(u_1)B_0^n(v_1) & \cdots & B_0^m(u_1)B_n^n(v_1) & \cdots & B_m^m(u_1)B_0^n(v_1) & \cdots & B_m^m(u_1)B_n^n(v_1) \\ B_0^m(u_2)B_0^n(v_2) & \cdots & B_0^m(u_2)B_n^n(v_2) & \cdots & B_m^m(u_2)B_0^n(v_2) & \cdots & B_m^m(u_2)B_n^n(v_2) \\ \vdots & & \vdots & & \vdots & & \vdots \\ B_0^m(u_K)B_0^n(v_K) & \cdots & B_0^m(u_K)B_n^n(v_K) & \cdots & B_m^m(u_K)B_0^n(v_K) & \cdots & B_m^m(u_K)B_n^n(v_K) \end{bmatrix} \begin{bmatrix} b_{0,0} \\ \vdots \\ b_{0,n} \\ \vdots \\ b_{m,0} \\ \vdots \\ b_{m,n} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_K \end{bmatrix}.$$

Pri tem predpostavljamo, da so podatki taki, da je $K \geq (m+1)(n+1)$ in da je matrika zgornjega sistema polnega ranga.

1. Sestavite metodo `lsqbezier2` za izračun koeficientov polinoma stopnje (m,n), ki po metodi najmanjših kvadratov najbolje aproksimira podatke (u_k, v_k, f_k), $k = 1, 2, \ldots, K$.

   Opis metode:

```
function B = lsqbezier2(m,n,P)
% Opis:
%  lsqbezier2 vrne koeficiente tenzorskega polinoma, ki po metodi
%  najmanjših kvadratov najbolje aproksimira dane podatke
%
% Definicija:
%  B = lsqbezier2(m,n,P)
%
% Vhodni podatki:
%  m,n      parametra, ki določata stopnjo polinoma,
%  P        matrika podatkov, ki v vsaki vrstici vsebuje parametra z
%           intervala [0,1] ter njima pripadajočo vrednost
%
% Izhodni podatek:
%  B        matrika velikosti n+1 x m+1 s koeficienti polinoma, ki po
%           metodi najmanjših kvadratov najbolje aproksimira podatke
end
```

2. Uporabite zgornjo metodo za konstrukcijo Bézierjeve ploskve izbrane stopnje (m, n), ki po metodi najmanjših kvadratov najbolje aproksimira 2500 točk na grafu funkcije, ki jo določa v Matlabu vgrajena funkcija `peaks`. Uporabite spodnje ukaze.

```
[X,Y,Z] = peaks(50);
P = [(X(:)+3)/6 (Y(:)+3)/6 Z(:)];
```

Tabeli `X` in `Y` sta velikosti 50×50 in določata kartezične koordinate enakomerno razporejenih točk v kvadratu $[-3, 3] \times [-3, 3]$. Tabela `Z` je prav tako velikosti 50×50, vsak njen element pa ustreza vrednosti funkcije v točki, ki je določena z istoležnima elementoma iz tabel `X` in `Y`. Vrednostim iz tabele `Z` priredimo parametre (u_k, v_k), $k = 1, 2, \ldots, 2500$, ki jih dobimo z linearno transformacijo kartezičnih koordinat s kvadrata $[-3, 3] \times [-3, 3]$ na kvadrat $[0, 1] \times [0, 1]$. Na ta način je določena tabela podatkov `P` velikosti 2500×3.

Podatke iz tabele `P` aproksimirajte po metodi najmanjših kvadratov in s tem določite aplikate kontrolnih točk Bézierjeve ploskve. Za izračun abscis in ordinat upoštevajte, da lahko funkciji $(x, y) \mapsto x$ in $(x, y) \mapsto y$ na kvadratu $[-3, 3] \times [-3, 3]$ parametriziramo s polinomoma

$$\sum_{i=0}^{m}\sum_{j=0}^{n}\left(6\frac{i}{m}-3\right)B_i^m(u)B_j^n(v), \quad \sum_{i=0}^{m}\sum_{j=0}^{n}\left(6\frac{j}{n}-3\right)B_i^m(u)B_j^n(v), \quad (u,v)\in[0,1]\times[0,1].$$

Izračunajte, kakšno je maksimalno absolutno odstopanje med aplikatami Bézierjeve ploskve pri parametrih (u_k, v_k), $k = 1, 2, \ldots, 2500$, in podanimi vrednostmi iz tretjega stolpca tabele `P`.

Primeri:

```
% izračun kontrolnih točk Bezierjeve ploskve
m = 6; n = 5;
[Bx,By] = meshgrid(linspace(-3,3,m+1),linspace(-3,3,n+1));
Bz = lsqbezier2(m,n,P)
```

```
Bz = 6x7
   -0.0640    1.2497   -6.9336   11.3035   -3.8045    1.2077   -0.1178
    0.4847  -10.6778   61.5171 -120.9325   50.8883  -14.8798    1.3845
    4.3289  -33.7952   60.9573   15.4253  -34.4044   23.1744   -3.2218
   -3.4583   32.3424  -94.5316    3.9893   56.2546  -20.2017    2.2863
    0.1022    2.7193  -23.6613  127.2018  -86.8229   21.8897   -1.8355
   -0.0321    0.0113    1.0979  -12.3700    9.3843   -2.3252    0.1926
```

```
% izračun vrednosti v točkah mreže
[u,v] = deal(linspace(0,1,7));
[~,~,bz] = bezier2(Bx,By,Bz,u,v)
```

```
bz = 7x7
   -0.0640   -0.3366    0.2283    1.2431    0.9859    0.2322   -0.1178
    0.7541    0.6380   -0.4826   -3.5943   -2.4867    0.0998    0.0588
    1.0111   -0.0803    0.1081   -1.5000   -0.6808    0.9262   -0.3188
    0.3608   -1.1137   -0.2862    1.4325    2.0699    1.2941   -0.3605
   -0.3769   -1.1661    0.4246    4.0176    3.3267    0.6862   -0.3001
   -0.3870   -0.1604    2.5365    5.0499    2.3442   -0.2834   -0.3919
   -0.0321   -0.3745   -1.6190   -1.6233   -0.1305    0.3617    0.1926
```

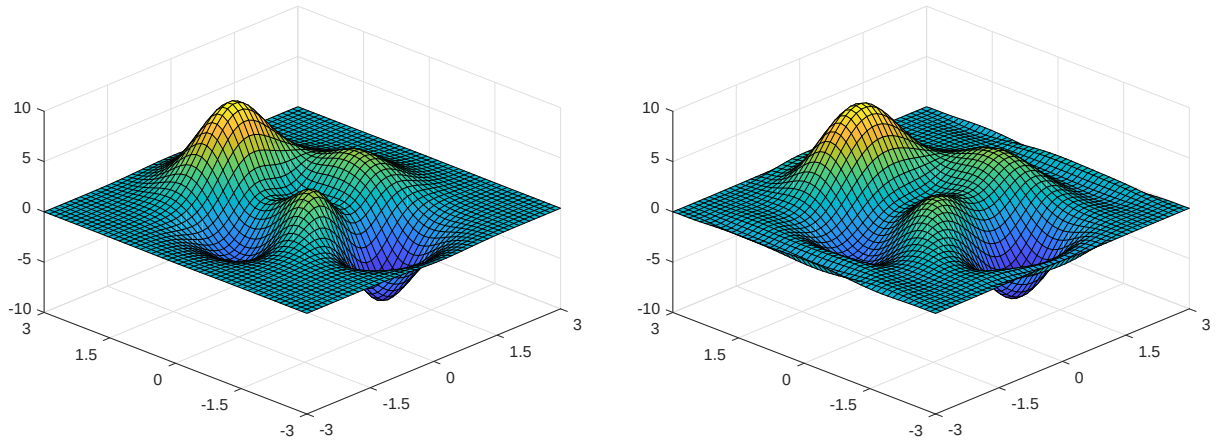
```
% maksimalna absolutna napaka v točkah mreže
[u,v] = deal(linspace(0,1,50));
[~,~,bz] = bezier2(Bx,By,Bz,u,v);
norm(bz(:)-P(:,3),Inf)
```

```
ans = 4.0880
```

```
% izračun za višjo stopnjo
m = 9; n = 10;
[Bx,By] = meshgrid(linspace(-3,3,m+1),linspace(-3,3,n+1));
Bz = lsqbezier2(m,n,P);
[bx,by,bz] = bezier2(Bx,By,Bz,u,v);
norm(bz(:)-P(:,3),Inf)
```

```
ans = 0.9221
```

```
surf(X,Y,Z);
surf(bx,by,bz);
```



(a) Graf funkcije `peaks` (b) Aproksimacijska Bézierjeva ploskev

Aproksimacija po metodi najmanjših kvadratov.

Naloga 14. *Aproksimacija s sestavljenimi Bézierjevimi ploskvami.*

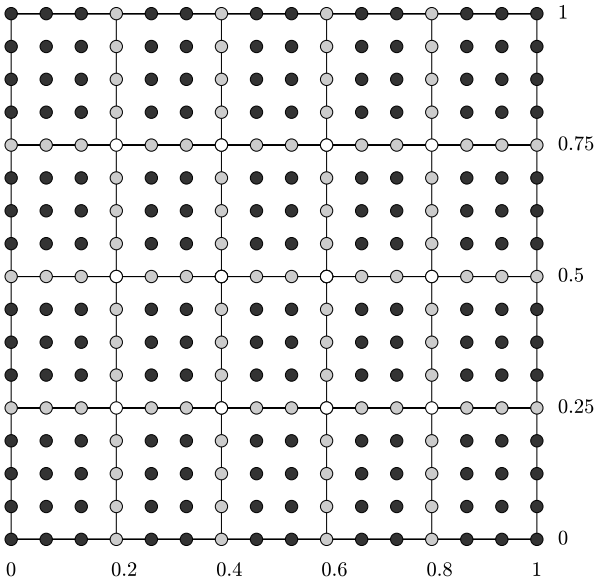
Z višanjem stopnje polinoma pridobimo več parametrov svobode. Pri metodi najmanših kvadratov lahko na ta način dobimo boljšo aproksimacijo podatkov f_k, $k = 1, 2, \ldots, K$, prirejenim parametrom $(u_k, v_k) \in [0,1] \times [0,1]$, a matrika predoločenega sistema s tem postaja večja, reševanje problema pa zato računsko zahtevnejše in numerično občutljivejše. S konceptom sestavljanja polinomov lahko ostanemo pri nizki stopnji, število parametrov svobode pa večamo z delitvijo domene parametrizacije.
Naj točke $U_I = \frac{I}{M}$, $I = 0, 1, \ldots, M$, razmaknjene za $\frac{1}{M}$, $M \in \mathbb{N}$, določajo delitev intervala $[0,1]$ v smeri parametra u, točke $V_J = \frac{J}{N}$, $J = 0, 1, \ldots, N$, razmaknjene za $\frac{1}{N}$, $N \in \mathbb{N}$, pa delitev intervala $[0,1]$ v smeri parametra v. Sestavljen polinom ali zlepek definiramo na kvadratu $[0,1] \times [0,1]$ kot funkcijo, ki na vsakem pravokotniku $[U_{I-1}, U_I] \times [V_{J-1}, V_J]$, $(I, J) \in \{1, 2, \ldots, M\} \times \{1, 2, \ldots, N\}$, ustreza polinomu. Nadaljevanje opisuje konstrukcijo zvezno odvedljivega zlepka, ki določa aproksimacijo podatkov v smislu metode najmanjših kvadratov. Sestavljena je iz dveh faz: lokalne aproksimacije in glajenja.
Najprej za vsak par (I, J) določimo polinom $b^{(I,J)}$ oblike

$$b^{(I,J)}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{i,j}^{(I,J)} B_i^m \left(\tfrac{u-U_{I-1}}{U_I - U_{I-1}}\right) B_j^n \left(\tfrac{v-V_{J-1}}{V_J - V_{J-1}}\right), \quad (u,v) \in [U_{I-1}, U_I] \times [V_{J-1}, V_J],$$

ki po metodi najmanjših kvadratov najbolje aproksimira podatke

$$\left(\tfrac{u_k-U_{I-1}}{U_I - U_{I-1}}, \tfrac{v_k-V_{J-1}}{V_J - V_{J-1}}, f_k\right), \quad (u_k, v_k) \in [U_{I-1}, U_I] \times [V_{J-1}, V_J].$$

Zlepek določajo koeficienti $b_{i,j}^{(I,J)}$, $(I, J) \in \{1, 2, \ldots, M\} \times \{1, 2, \ldots, N\}$, z indeksi $0 < i < m$ in $0 < j < n$ ter nekaterimi indeksi, ki so povezani z robom domene. Ti koeficienti so na spodnji sliki, ki prikazuje primer $M = 5$, $N = 4$, $m = 3$, $n = 4$, ponazorjeni s črno obarvanimi točkami.



Koeficiente, ki so na sliki ponazorjeni s sivo in belo barvo, določimo na podlagi pogojev zvezne odvedljivosti. Koeficienti $b_{i,0}^{(I,J)} = b_{i,n}^{(I,J-1)}$, $J > 1$, za indekse $0 < i < m$ ter $i = 0$ za

$I = 1$ in $i = m$ za $I = M$, ki so ponazorjeni s sivo barvo, so določeni s predpisom

$$b_{i,0}^{(I,J)} = b_{i,n}^{(I,J-1)} = \frac{1}{2} b_{i,n-1}^{(I,J-1)} + \frac{1}{2} b_{i,1}^{(I,J)}.$$

Podobno so koeficienti $b_{0,j}^{(I,J)} = b_{m,j}^{(I-1,J)}$, $I > 1$, za indekse $0 < j < n$ ter $j = 0$ za $J = 1$ in $j = n$ za $J = N$ določeni s predpisom

$$b_{0,j}^{(I,J)} = b_{m,j}^{(I-1,J)} = \frac{1}{2} b_{m-1,j}^{(I-1,J)} + \frac{1}{2} b_{1,j}^{(I,J)}.$$

Koeficient $b_{0,0}^{(I,J)} = b_{m,0}^{(I-1,J)} = b_{m,n}^{(I-1,J-1)} = b_{0,n}^{(I,J-1)}$ za $I > 1$ in $J > 1$, ki je ponazorjen z belo barvo, pa določimo s predpisom

$$b_{0,0}^{(I,J)} = b_{m,0}^{(I-1,J)} = b_{m,n}^{(I-1,J-1)} = b_{0,n}^{(I,J-1)} = \frac{1}{4} b_{1,1}^{(I,J)} + \frac{1}{4} b_{m-1,1}^{(I-1,J)} + \frac{1}{4} b_{m-1,n-1}^{(I-1,J-1)} + \frac{1}{4} b_{1,n-1}^{(I,J-1)}.$$

1. Sestavite metodo `lsqbezier2spline` za izračun koeficientov zlepka nad delitvijo kvadrata domene, ki je določen s parametroma M in N, ki so določeni s koeficienti polinomov stopnje (m, n), ki po metodi najmanjših kvadratov najbolje aproksimirajo del podatkov (u_k, v_k, f_k), $k = 1, 2, \ldots, K$, nad pravokotnikom delitve.

   Opis metode:

```
function S = lsqbezier2spline(M,N,m,n,P)
% Opis:
%  lsqbezier2spline vrne kontrolne točke tenzorskih polinomov, ki
%  določajo zvezno odvedljiv zlepek, ki v smislu metode najmanjših
%  kvadratov najbolje aproksimira dane podatke
%
% Definicija:
%  S = lsqbezier2spline(M,N,m,n,P)
%
% Vhodni podatki:
%  M,N      parametra, ki določata delitev domene parametrizacije,
%  m,n      parametra, ki določata stopnjo polinoma nad pravokotnikom
%           delitve,
%  P        matrika podatkov, ki v vsaki vrstici vsebuje parametra z
%           intervala [0,1] ter njima pripadajočo vrednost
%
% Izhodni podatek:
%  S        celica velikosti N x M, v kateri vsak element vsebuje
%           matriko s koeficienti polinoma, ki določa zlepek nad
%           pravokotnikom delitve domene
end
```

2. Implementirano metodo uporabite za konstrukcijo sestavljene Bézierjeve ploskve nad delitvijo, določeno s parametroma $M = 4$ in $N = 5$, ki na vsakem pravokotniku delitve

ustreza ploskvi stopnje $(3, 4)$. Ploskev naj v smislu metode najmanjših kvadratov aproksimira 2500 točk na grafu funkcije, ki jih v Matlabu dobite z ukazom `peaks(50)`. Abscise in ordinate kontrolnih točk Bézierjevih ploskev določite tako, da bo ploskev eksaktno opisovala abscise in ordinate točk grafa, aplikate kontrolnih točk pa naj bodo aproksimacija aplikat točk grafa po zgoraj opisani konstrukciji. Izračunajte, kakšno je maksimalno absolutno odstopanje med aplikatami sestavljene Bézierjeve ploskve pri parametrih (u_k, v_k), $k = 1, 2, \ldots, 2500$, in aplikatami točk grafa funkcije. Narišite ploskev.

Primeri:

```
% podatki
[X,Y,Z] = peaks(50);
P = [(X(:)+3)/6 (Y(:)+3)/6 Z(:)];
```

```
% izračun koeficientov zlepka
M = 5; N = 4; m = 3; n = 4;
Sz = lsqbezier2spline(M,N,m,n,P);
```

```
Sz{1,1}
```

```
ans = 5x4
    0.0001   -0.0000    0.0010    0.0041
    0.0003    0.0006    0.0031    0.0235
    0.0000   -0.0017    0.0034   -0.0112
    0.0032   -0.0034    0.0446    0.1163
   -0.0008    0.0087   -0.0282    0.1214
```

```
Sz{1,2}
```

```
ans = 5x4
    0.0041    0.0072   -0.0013   -0.1179
    0.0235    0.0439    0.0439   -0.3146
   -0.0112   -0.0258   -0.1230   -0.6712
    0.1163    0.1881   -0.3798   -4.1526
    0.1214    0.2710    1.0946   -1.9805
```

```
Sz{1,3}
```

```
ans = 5x4
   -0.1179   -0.2345   -0.3141   -0.2064
   -0.3146   -0.6731   -0.9364   -0.6223
   -0.6712   -1.2194   -1.5699   -1.0099
   -4.1526   -7.9253  -10.0177   -6.6070
   -1.9805   -5.0556   -9.0020   -5.7833
```

```
Sz{2,1}
```

```
ans = 5x4
   -0.0008    0.0087   -0.0282    0.1214
   -0.0048    0.0209   -0.1009    0.1265
   -0.0190    0.0370   -0.3094   -0.2168
   -0.0415   -0.0648   -0.4141   -2.1502
   -0.0423   -0.0776   -0.4006   -2.3334
```

```
Sz{2,2}
```

```
ans = 5x4
    0.1214    0.2710    1.0946   -1.9805
    0.1265    0.3539    2.5691    0.1916
   -0.2168   -0.1242    5.1766    8.0874
   -2.1502   -3.8864   -1.6129    1.2770
   -2.3334   -4.2663   -2.7188    0.5700
```

```
Sz{2,3}
```

```
ans = 5x4
   -1.9805   -5.0556   -9.0020   -5.7833
    0.1916   -2.1859   -7.9863   -4.9596
    8.0874   10.9982    3.5525    3.1480
    1.2770    4.1669   -2.8800    0.3834
    0.5700    3.8588   -1.9989    1.0225
```

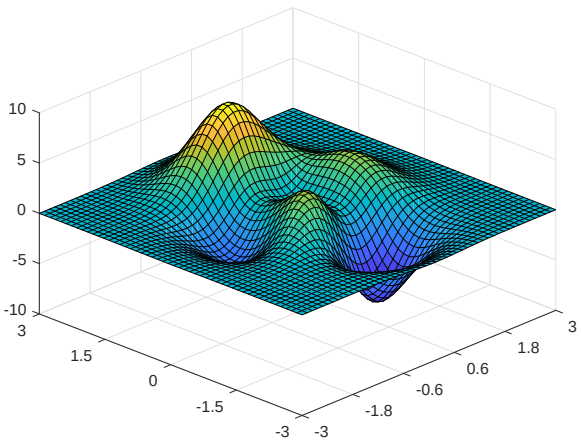
```
% izračun točk na sestavljeni ploskvi
[bx,by,bz] = deal(zeros(50));

% pomožni parametri
U = linspace(0,1,M+1);
V = linspace(0,1,N+1);
[u,v] = deal(linspace(0,1,50));
[Bx0,By0] = meshgrid(linspace(0,1,m+1),linspace(0,1,n+1));

for I = 1:M
    % točke na lokalnem intervalu
    ur = U(I) <= u & u <= U(I+1);

    for J = 1:N
        % točke na lokalnem intervalu
        vr = V(J) <= v & v <= V(J+1);
```

```
        % lokalni parametri
        ul = (u(ur)-U(I))/(U(I+1)-U(I));
        vl = (v(vr)-V(J))/(V(J+1)-V(J));

        % kontrolne točke
        Bx = 6*((U(I+1)-U(I))*Bx0+U(I))-3;
        By = 6*((V(J+1)-V(J))*By0+V(J))-3;

        Bz = Sz{J,I};
        % izračun točk
        [bx(vr,ur),by(vr,ur),bz(vr,ur)] = bezier2(Bx,By,Bz,ul,vl);
    end
end
```

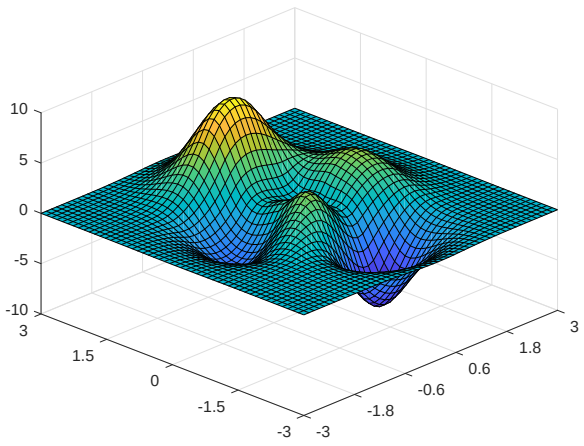
```
% maksimalna absolutna napaka
norm(Z(:)-bz(:),Inf)
```

```
ans = 0.6999
```

```
% izris
surf(X,Y,Z);
surf(bx,by,bz);
```



(a) Graf funkcije `peaks`



(b) Sestavljena Bézierjeva ploskev

Aproksimacija v smislu metode najmanjših kvadratov.

Naloga 15. *Polinomi dveh spremenljivk.*

Prostor polinomov $\mathbb{P}_n^2$ dveh spremenljivk stopnje n je dimenzije $\binom{n+2}{2}$ in ga lahko opišemo s funkcijami

$$(x, y) \mapsto x^i\, y^j, \quad 0 \le i + j \le n.$$

Za geometrijsko oblikovanje je priročnejša predstavitev z Bernsteinovimi baznimi polinomi, pri definiciji katerih namesto kartezičnih uporabljamo baricentrične koordinate. Baricentrične koordinate točke $\boldsymbol{P} = (x, y) \in \mathbb{R}^2$ glede na trikotnik $T = (\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3)$ z oglišči $\boldsymbol{V}_i = (x_i, y_i) \in \mathbb{R}^2$, $i = 1, 2, 3$, so podane s trojico $(u, v, w) \in \mathbb{R}^3$, ki je enolično določena kot rešitev sistema

$$\begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 \\ x \\ y \end{bmatrix}.$$

Bernsteinovi bazni polinomi dveh spremenljivk stopnje n so definirani kot

$$B_{i,j,k}^n(u, v, w) = \frac{n!}{i!\, j!\, k!}\, u^i\, v^j\, w^k, \qquad i + j + k = n,$$

in sestavljajo bazo za $\mathbb{P}_n^2$. Vsak polinom $p \in \mathbb{P}_n^2$ lahko torej predstavimo v tako imenovani Bézierjevi obliki kot

$$p = \sum_{i+j+k=n} b_{i,j,k}\, B_{i,j,k}^n$$

za neka realna števila $b_{i,j,k}$.

1. V Matlabu sestavite metodo `pointbary`, ki kartezične koordinate (x, y) točke $\boldsymbol{P}$ pretvori v baricentrične koordinate (u, v, w) glede na trikotnik T. Pripravite tudi metodo `vectorbary`, ki vrne baricentrične koordinate vektorja $\boldsymbol{P} - \boldsymbol{0}$.

   Opis metode:

```
function u = pointbary(P,T)
% Opis:
%  pointbary vrne baricentrične koordinate točke glede na dan trikotnik
%
% Definicija:
%  u = pointbary(P,T)
%
% Vhodna podatka:
%  P    vrstica, ki predstavlja kartezične koordinate točke v ravnini,
%  T    matrika s tremi vrsticami, v kateri vsaka vrstica predstavlja
%       kartezične koordinate oglišč trikotnika
%
% Izhodni podatek:
%  u    vrstica dolžine 3, ki predstavlja baricentrične koordinate
%       točke, podane s P, glede na trikotnik, podan s T
end
```

Primeri:

```
% trikotnik
T = [0 0; 5 1; 3 3];
```

```
p1 = pointbary([0 0],T)
```

```
p1 = 1x3
     1     0     0
```

```
p2 = pointbary([1 1],T)
```

```
p2 = 1x3
    0.6667         0    0.3333
```

```
p3 = pointbary([4 2],T)
```

```
p3 = 1x3
         0    0.5000    0.5000
```

```
x = vectorbary([1 0],T)
```

```
x = 1x3
   -0.1667    0.2500   -0.0833
```

```
y = vectorbary([0 1],T)
```

```
y = 1x3
   -0.1667   -0.2500    0.4167
```

2. Vrednost polinoma $p \in \mathbb{P}_n^2$ v točki $\boldsymbol{P}$, kjer je polinom p podan v Bézierjevi obliki, točka $\boldsymbol{P}$ pa z baricentričnimi koordinatami (u, v, w), lahko izračunamo z de Casteljaujevim postopkom:

$$b_{i,j,k}^r = u\, b_{i+1,j,k}^{r-1} + v\, b_{i,j+1,k}^{r-1} + w\, b_{i,j,k+1}^{r-1}, \quad i+j+k = n-r, \quad r = 1, 2, \ldots, n.$$

Ta se začne s koeficienti

$$b_{i,j,k}^0 = b_{i,j,k}, \quad i+j+k = n,$$

konča pa z vrednostjo $b^n_{0,0,0}$ polinoma p v točki $\boldsymbol{P}$. Postopek lahko enostavno prilagodimo za računanje vrednosti trikotniškega razcveta $\mathcal{B}[p](\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots, \boldsymbol{P}_n)$ polinoma p pri argumentih $\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots, \boldsymbol{P}_n$. V tem primeru v r-tem koraku postopka uporabimo baricentrične koordinate argumenta $\boldsymbol{P}_r$ (kar so lahko v odvisnosti od konteksta baricentrične koordinate točke ali vektorja). Sestavite metodo `blossom3`, ki izračuna trikotniški razcvet pri podanih baricentričnih koordinatah, in njeno izpeljanko `decasteljau3`, ki izračuna vrednost polinoma.

Opis metode:

```
function b = blossom3(B,U)
% Opis:
%  blossom3 izračuna razcvet polinoma dveh spremenljivk
%
% Definicija:
%  b = blossom3(B,U)
%
% Vhodna podatka:
%  B    matrika velikosti n+1 x n+1, ki predstavlja koeficiente
%       polinoma dveh spremenljivk stopnje n v Bezierjevi obliki
%       (element matrike na mestu (i,j), j <= n+2-i, določa
%       koeficient polinoma z indeksom (n+2-i-j, j-1, i-1)),
%  U    matrika velikosti n x 3, v kateri vrstice predstavljavo
%       baricentrične koordinate točk glede na domenski trikotnik,
%       za katere izvajamo razcvet polinoma
%
% Izhodni podatek:
%  b    vrednost razcveta polinoma, določenega z matriko B, v
%       točkah, določenih z matriko U
end
```

3. S pomočjo implementiranih metod izračunajte vrednost polinoma

$$2B^3_{3,0,0} + B^3_{2,1,0} - B^3_{1,2,0} + 5B^3_{2,0,1} + 3B^3_{1,1,1} - 4B^3_{0,2,1} + B^3_{0,0,3}$$

nad trikotnikom z oglišči $\boldsymbol{V}_1 = (0,0)$, $\boldsymbol{V}_2 = (5,1)$, $\boldsymbol{V}_3 = (3,3)$ v točkah $(0,0)$, $(1,1)$ in $(4,2)$. Z upoštevanjem, da se odvod $p \in \mathbb{P}^2_n$ v smereh $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_r$, $r \leq n$, izvrednoten v točki $\boldsymbol{P}$, izraža kot

$$D_{\boldsymbol{d}_1} D_{\boldsymbol{d}_2} \ldots D_{\boldsymbol{d}_r} p(\boldsymbol{P}) = \frac{n!}{(n-r)!} \mathcal{B}[p](\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_r, \underbrace{\boldsymbol{P}, \ldots, \boldsymbol{P}}_{n-r}),$$

določite še prve in druge odvode polinoma p v teh točkah v smereh x in y.

Primeri:

```
% koeficienti polinoma v Bezierjevi obliki
B = [2 1 -1 0; 5 3 -4 NaN; 0 0 NaN NaN; 1 NaN NaN NaN];
```

```
% vrednosti p v točkah (0,0), (1,1), (4,2)
[decasteljau3(B,p1) decasteljau3(B,p2) decasteljau3(B,p3)]
```

```
ans = 1x3
    2.0000    2.8519   -1.3750
```

```
% vrednosti Dxp v točkah (0,0), (1,1), (4,2)
3*[blossom3(B,[x;p1;p1]) blossom3(B,[x;p2;p2]) blossom3(B,[x;p3;p3])]
```

```
ans = 1x3
   -1.5000   -0.8056   -1.9375
```

```
% vrednosti Dyp v točkah (0,0), (1,1), (4,2)
3*[blossom3(B,[y;p1;p1]) blossom3(B,[y;p2;p2]) blossom3(B,[y;p3;p3])]
```

```
ans = 1x3
    4.5000    0.0278   -0.0625
```

```
% vrednosti Dxxp v točkah (0,0), (1,1), (4,2)
6*[blossom3(B,[x;x;p1]) blossom3(B,[x;x;p2]) blossom3(B,[x;x;p3])]
```

```
ans = 1x3
   -0.4583   -1.0139    0.0208
```

```
% vrednosti Dxyp v točkah (0,0), (1,1), (4,2)
6*[blossom3(B,[x;y;p1]) blossom3(B,[x;y;p2]) blossom3(B,[x;y;p3])]
```

```
ans = 1x3
    1.2917    1.5694   -0.8542
```

```
% vrednosti Dyyp v točkah (0,0), (1,1), (4,2)
6*[blossom3(B,[y;y;p1]) blossom3(B,[y;y;p2]) blossom3(B,[y;y;p3])]
```

```
ans = 1x3
   -7.4583   -4.3472    2.0208
```

Naloga 16. *Trikotne Bézierjeve ploskve.*

Trikotna Bézierjeva ploskev stopnje n je podana s parametrizacijo

$$\boldsymbol{b}(u,v,w) = \sum_{i+j+k=n} \boldsymbol{b}_{i,j,k}\, B^n_{i,j,k}(u,v,w), \qquad 0 \le u,v,w \le 1, \quad u+v+w = 1,$$

kjer so z $B^n_{i,j,k}$ označeni Bernsteinovi bazni polinomi dveh spremenljivk stopnje n. Ploskev leži v konveksni ovojnici kontrolnih točk $\boldsymbol{b}_{i,j,k} \in \mathbb{R}^3$, ki jih organiziramo v kontrolno mrežo.

1. Sestavite metodo `bezier3`, ki s pomočjo de Casteljaujevega postopka izračuna točke na ploskvi pri podanih parametrih.

   Opis metode:

```
function b = bezier3(Bx,By,Bz,U)
% Opis:
%  bezier3 izračuna točke na trikotni Bezierjevi ploskvi
%
% Definicija:
%  b = bezier3(Bx,By,Bz,U)
%
% Vhodni podatki:
%  Bx, By, Bz   matrike velikosti n+1 x n+1, ki predstavljajo
%               koordinate kontrolnih točk Bezierjeve krpe (element
%               posamezne matrike na mestu (i,j), j <= n+2-i, določa
%               koordinato kontrolne točke z indeksom
%               (n+2-i-j, j-1, i-1)),
%  U            matrika, v kateri vrstice predstavljajo baricentrične
%               koordinate točk glede na domenski trikotnik, za katere
%               računamo točke na Bezierjevi krpi
%
% Izhodni podatek:
%  b            matrika, v kateri vsaka vrstica predstavlja točko na
%               Bezierjevi krpi pri istoležnih parametrih iz matrike U
end
```

2. Pripravite metodo `plotbezier3`, ki nariše trikotno ploskev in njeno kontrolno mrežo. Pomagajte si z vgrajenima funkcijama `trisurf` in `trimesh`. Pri pripravi vhodnih podatkov lahko uporabite funkcijo `trimeshgrid`, ki je podana spodaj. S pomočjo implementirane metode narišite kubično ploskev, ki jo določajo kontrolne točke

$$\boldsymbol{b}_{0,0,3} = (4,5,3),$$
$$\boldsymbol{b}_{1,0,2} = (2,4,0), \quad \boldsymbol{b}_{0,1,2} = (5,3,5),$$
$$\boldsymbol{b}_{2,0,1} = (1,2,4), \quad \boldsymbol{b}_{1,1,1} = (3,2,-2), \quad \boldsymbol{b}_{0,2,1} = (7,3,3),$$
$$\boldsymbol{b}_{3,0,0} = (0,0,-2), \quad \boldsymbol{b}_{2,1,0} = (2,1,1), \quad \boldsymbol{b}_{1,2,0} = (6,0,-2), \quad \boldsymbol{b}_{0,3,0} = (8,-1,0).$$

```
function [TRI,U] = trimeshgrid(d)
% Opis:
%  trimeshhgrid sestavi triangulacijo trikotnika z oglišči (0,0),
%  (1,0), (0,1) in izračuna točke triangulacije v obliki
%  baricentričnih koordinat
%
% Definicija:
%  [TRI,U] = trimeshgrid(d)
%
% Vhodni podatek:
%  d    število intervalov, na katere je razdeljena stranica
%       trikotnika
%
% Izhodna podatka:
%  TRI  matrika velikosti d^2 x 3, v kateri vsaka vrstica predstavlja
%       en trikotnik enakomerne triangulacije, ki deli osnovni
%       trikotnik,
%  U    matrika velikosti (d+2)*(d+1)/2 x 3, v kateri vsaka vrstica
%       predstavlja baricentrične koordinate ene točke triangulacije

TRI = zeros(d^2,3);
U = zeros((d+1)*(d+2)/2,3);

k = 1;
l = 1;
for i = 0:d
    for j = 0:d-i
        U(k,:) = [d-i-j j i]/d;
        if i < d && j < d-i
            shift = d+1-i;
            TRI(l,:) = [k k+1 k+shift];
            l = l+1;
            if j < d-i-1
                TRI(l,:) = [k+1 k+shift k+1+shift];
                l = l+1;
            end
        end
        k = k+1;
    end
end

end
```

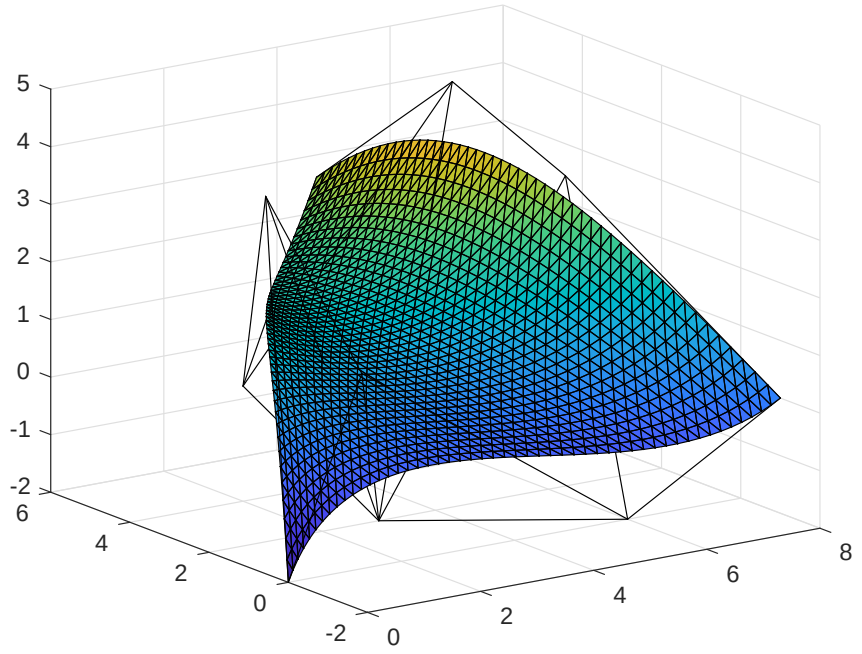
Primeri:

```
Bx = [0 2 6 8; 1 3 7 NaN; 2 5 NaN NaN; 4 NaN NaN NaN];
By = [0 1 0 -1; 2 2 3 NaN; 4 3 NaN NaN; 5 NaN NaN NaN];
Bz = [-2 1 -2 0; 4 -2 3 NaN; 0 5 NaN NaN; 3 NaN NaN NaN];
```

```
[~,U] = trimeshgrid(3);
bezier3(Bx,By,Bz,U)
```

```
ans = 10x3
         0         0   -2.0000
    2.5185    0.4074   -0.5926
    5.4815   -0.0741   -0.7407
    8.0000   -1.0000         0
    1.0370    1.9630    1.2963
    3.6667    2.0370    0.8148
    6.7407    1.8889    2.5556
    2.2963    3.7037    1.7037
    5.2593    3.4444    3.7778
    4.0000    5.0000    3.0000
```

```
% četrti parameter (50) določa vhodni podatek za metodo trimeshgrid
plotbezier3(Bx,By,Bz,50);
```



Primer trikotne Bézierjeve ploskve stopnje 3.

Naloga 17. *Argyrisova interpolacijska shema.*

Za aproksimacijo funkcije f s polinomom dveh spremenljivk so na voljo številne interpolacijske sheme, s pomočjo katerih polinom določimo na podlagi interpolacije v ogliščih trikotnika, vzdolž njegovih stranic, včasih pa tudi v notranjosti trikotnika.
Ena izmed klasičnih shem je Argyrisova shema. Naj bo f dvakrat odvedljiva funkcija nad trikotnikom $T = (\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3) \subset \mathbb{R}^2$. Aproksimacijski polinom $p \in \mathbb{P}_5^2$ je določen z interpolacijo vrednosti ter prvih in drugih odvodov v ogliščih trikotnika ter interpolacijo smernega odvoda v središču vsake izmed stranic trikotnika. Predstavimo ga v Bézierjevi obliki

$$p = \sum_{i+j+k=5} b_{i,j,k} B_{i,j,k}^5$$

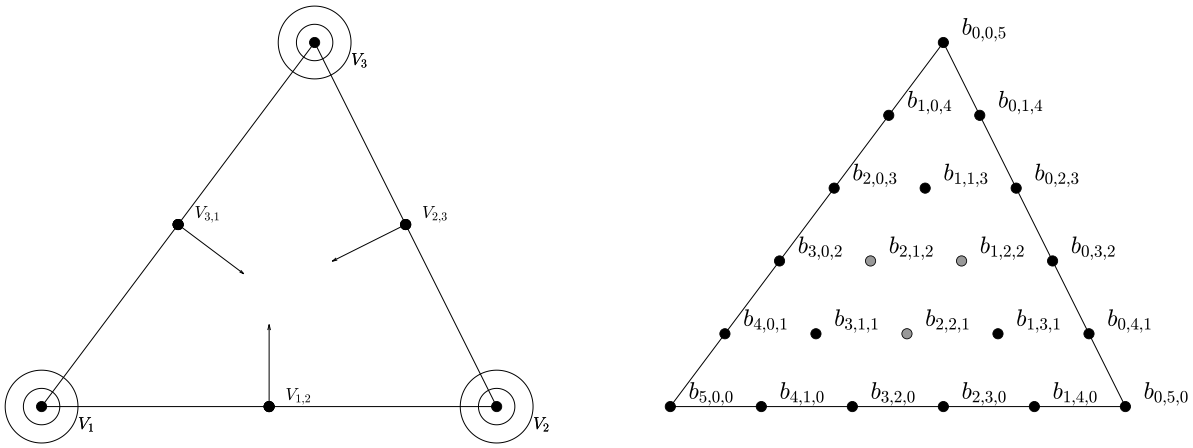
nad trikotnikom T.
Koeficienti polinoma p, pripadajoči točkam domene v okolici oglišča $\boldsymbol{V}_1$, so podani s formulami

$$\begin{aligned}
b_{5,0,0} &= f(\boldsymbol{V}_1),\\
b_{4,1,0} &= f(\boldsymbol{V}_1) + \frac{1}{5}\langle \boldsymbol{D}f(\boldsymbol{V}_1), \boldsymbol{V}_2 - \boldsymbol{V}_1\rangle,\\
b_{4,0,1} &= f(\boldsymbol{V}_1) + \frac{1}{5}\langle \boldsymbol{D}f(\boldsymbol{V}_1), \boldsymbol{V}_3 - \boldsymbol{V}_1\rangle,\\
b_{3,2,0} &= f(\boldsymbol{V}_1) + \frac{2}{5}\langle \boldsymbol{D}f(\boldsymbol{V}_1), \boldsymbol{V}_2 - \boldsymbol{V}_1\rangle + \frac{1}{20}\langle \boldsymbol{V}_2 - \boldsymbol{V}_1, \boldsymbol{H}f(\boldsymbol{V}_1)\cdot(\boldsymbol{V}_2 - \boldsymbol{V}_1)\rangle,\\
b_{3,1,1} &= f(\boldsymbol{V}_1) + \frac{1}{5}\langle \boldsymbol{D}f(\boldsymbol{V}_1), \boldsymbol{V}_2 - \boldsymbol{V}_1\rangle + \frac{1}{5}\langle \boldsymbol{D}f(\boldsymbol{V}_1), \boldsymbol{V}_3 - \boldsymbol{V}_1\rangle\\
&\qquad + \frac{1}{20}\langle \boldsymbol{V}_3 - \boldsymbol{V}_1, \boldsymbol{H}f(\boldsymbol{V}_1)\cdot(\boldsymbol{V}_2 - \boldsymbol{V}_1)\rangle,\\
b_{3,0,2} &= f(\boldsymbol{V}_1) + \frac{2}{5}\langle \boldsymbol{D}f(\boldsymbol{V}_1), \boldsymbol{V}_3 - \boldsymbol{V}_1\rangle + \frac{1}{20}\langle \boldsymbol{V}_3 - \boldsymbol{V}_1, \boldsymbol{H}f(\boldsymbol{V}_1)\cdot(\boldsymbol{V}_3 - \boldsymbol{V}_1)\rangle.
\end{aligned}$$

Pri tem $\boldsymbol{D}f : \mathbb{R}^2 \to \mathbb{R}^2$ označuje gradient, $\boldsymbol{H}f : \mathbb{R}^2 \to \mathbb{R}^{2\times 2}$ pa Hessejevo matriko funkcije f.
Analogne formule veljajo za koeficiente $b_{0,5,0}, b_{0,4,1}, b_{1,4,0}, b_{0,3,2}, b_{1,3,1}, b_{2,3,0}$, ki pripadajo točkam domene v okolici oglišča $\boldsymbol{V}_2$, in koeficiente $b_{0,0,5}, b_{1,0,4}, b_{0,1,4}, b_{2,0,3}, b_{1,1,3}, b_{0,2,3}$, ki pripadajo točkam domene v okolici oglišča $\boldsymbol{V}_3$.
Koeficient $b_{2,2,1}$ polinoma p je določen z vrednostmi v ogliščih $\boldsymbol{V}_1$ in $\boldsymbol{V}_2$ ter vrednostjo odvoda f v točki $\boldsymbol{V}_{1,2} = \frac{1}{2}\boldsymbol{V}_1 + \frac{1}{2}\boldsymbol{V}_2$ v smeri enotskega vektorja $\boldsymbol{n}_{1,2}$, ki je pravokoten na $\boldsymbol{V}_2 - \boldsymbol{V}_1$. Naj bodo z $(\vartheta_1, \vartheta_2, \vartheta_3)$ podane baricentrične koordinate vektorja $\boldsymbol{n}_{1,2}$ glede na $T = (\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3)$. Koeficient je podan s formulo

$$\begin{aligned}
b_{2,2,1} = \frac{8}{15}\frac{1}{\vartheta_3}\langle \boldsymbol{D}f(\boldsymbol{V}_{1,2}), \boldsymbol{n}_{1,2}\rangle &- \frac{1}{6}\left(b_{4,0,1} + 4b_{3,1,1} + 4b_{1,3,1} + b_{0,4,1}\right)\\
&- \frac{1}{6}\frac{\vartheta_1}{\vartheta_3}\left(b_{5,0,0} + 4b_{4,1,0} + 6b_{3,2,0} + 4b_{2,3,0} + b_{1,4,0}\right)\\
&- \frac{1}{6}\frac{\vartheta_2}{\vartheta_3}\left(b_{0,5,0} + 4b_{1,4,0} + 6b_{2,3,0} + 4b_{3,2,0} + b_{4,1,0}\right).
\end{aligned}$$

Podobno sta podana koeficienta $b_{1,2,2}$ in $b_{2,1,2}$, ki ju dobimo z dodatno interpolacijo v središčih stranic med $\boldsymbol{V}_2$ in $\boldsymbol{V}_3$ ter med $\boldsymbol{V}_3$ in $\boldsymbol{V}_1$. S tem je določenih vseh 21 koeficientov polinoma p.



(a) Argyrisova shema (b) Bézierjeva oblika Argyrisovega polinoma

Shematičen prikaz konstrukcije interpolanta po Argyrisovi shemi.

1. Sestavite metodo `argyris`, ki na podlagi ustreznih podatkov o funkciji f izračuna Bézierjeve koeficiente polinoma p nad trikotnikom T po Argyrisovi shemi.

   Opis metode:

```
function B = argyris(T,f,Df,Hf)
% Opis:
%  argyris izračuna Bezierjeve ordinate polinoma dveh spremenljivk
%  stopnje 5, ki interpolira vrednosti, prve in druge odvode podane
%  funkcije f v ogliščih trikotnika ter odvode funkcije f v središčih
%  stranic trikotnika v smereh, pravokotnih na posamezno stranico
%
% Definicija:
%  B = argyris(T,f,Df,Hf)
%
% Vhodni podatki:
%  T    tabela velikosti 3 x 2, v kateri vsaka vrstica predstavlja
%       oglišče trikonika, nad katerim je definiran polinom,
%  f    funkcija, ki jo interpoliramo,
%  Df   gradient funkcije, ki jo interpoliramo,
%  Hf   Hessejeva matrika funkcije, ki jo interpoliramo
%
% Izhodni podatek:
%  B    matrika velikosti 6 x 6, ki predstavlja koeficiente polinoma
%       dveh spremenljivk stopnje 5 v Bezierjevi obliki (element
%       matrike na mestu (i,j), j <= 7-i, določa koeficient polinoma
%       z indeksom (7-i-j, j-1, i-1)), ki interpolira funkcijo f
end
```

2. Izračunajte koeficiente Argyrisovega polinoma nad trikotnikom z oglišči $\boldsymbol{V}_1 = (-1.7 - 1)$, $\boldsymbol{V}_2 = (-0.5, -1)$, $\boldsymbol{V}_3 = (-0.5, 0)$ za funkcijo f, ki jo predstavlja ukaz `peaks` v Matlabu. Vrednosti f in vrednosti odvodov f lahko izračunate s pomočjo naslednjih ukazov.

```
f = @(x,y) 3*(1-x).^2.*exp(-x.^2 - (y+1).^2) ...
    - 10*(x/5 - x.^3 - y.^5).*exp(-x.^2-y.^2) ...
    - 1/3*exp(-(x+1).^2 - y.^2);

dxf = @(x,y) (exp(-(x + 1)^2 - y^2)*(2*x + 2))/3 ...
    + 3*exp(- (y + 1)^2 - x^2)*(2*x - 2) ...
    + exp(- x^2 - y^2)*(30*x^2 - 2) ...
    - 6*x*exp(- (y + 1)^2 - x^2)*(x - 1)^2 ...
    - 2*x*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5);
dyf = @(x,y) (2*y*exp(- (x + 1)^2 - y^2))/3 ...
    + 50*y^4*exp(- x^2 - y^2) ...
    - 3*exp(- (y + 1)^2 - x^2)*(2*y + 2)*(x - 1)^2 ...
    - 2*y*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5);
Df = @(x,y) [dxf(x,y); dyf(x,y)];

dxxf = @(x,y) (2*exp(- (x + 1)^2 - y^2))/3 ...
    + 6*exp(- (y + 1)^2 - x^2) ...
    + 60*x*exp(- x^2 - y^2) ...
    - 6*exp(- (y + 1)^2 - x^2)*(x - 1)^2 ...
    - 2*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5) ...
    - (exp(- (x + 1)^2 - y^2)*(2*x + 2)^2)/3 ...
    + 12*x^2*exp(- (y + 1)^2 - x^2)*(x - 1)^2 ...
    + 4*x^2*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5) ...
    - 12*x*exp(- (y + 1)^2 - x^2)*(2*x - 2) ...
    - 4*x*exp(- x^2 - y^2)*(30*x^2 - 2);
dxyf = @(x,y) 4*x*y*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5) ...
    - (2*y*exp(- (x + 1)^2 - y^2)*(2*x + 2))/3 ...
    - 2*y*exp(- x^2 - y^2)*(30*x^2 - 2) ...
    - 100*x*y^4*exp(- x^2 - y^2) ...
    - 3*exp(- (y + 1)^2 - x^2)*(2*x - 2)*(2*y + 2) ...
    + 6*x*exp(- (y + 1)^2 - x^2)*(2*y + 2)*(x - 1)^2;
dyyf = @(x,y) (2*exp(- (x + 1)^2 - y^2))/3 ...
    - 6*exp(- (y + 1)^2 - x^2)*(x - 1)^2 ...
    - 2*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5) ...
    + 200*y^3*exp(- x^2 - y^2) ...
    - 200*y^5*exp(- x^2 - y^2) ...
    - (4*y^2*exp(- (x + 1)^2 - y^2))/3 ...
    + 4*y^2*exp(- x^2 - y^2)*(10*x^3 - 2*x + 10*y^5) ...
    + 3*exp(- (y + 1)^2 - x^2)*(2*y + 2)^2*(x - 1)^2;
Hf = @(x,y) [dxxf(x,y) dxyf(x,y); dxyf(x,y) dyyf(x,y)];
```

Narišite graf polinoma in izračunajte maksimalno absolutno napako aproksimacije f v točkah

$$\tfrac{N_1}{N}\boldsymbol{V}_1 + \tfrac{N_2}{N}\boldsymbol{V}_2 + \tfrac{N_3}{N}\boldsymbol{V}_3, \quad N_1 + N_2 + N_3 = N, \quad N_1, N_2, N_3 \geq 0,$$

za $N = 50$.

Primeri:

```
T = [-1.7 -1; -0.5 -1; -0.5 0];
Bz = argyris(T,f,Df,Hf)
```

```
Bz = 6x6
    0.0009    0.2372    0.8070    2.4482    2.9490    2.2247
   -0.0441    0.4645    2.1441    3.8935    3.8769       NaN
   -0.1210    1.6988    4.0850    4.7002       NaN       NaN
    0.2230    2.4540    3.2656       NaN       NaN       NaN
    1.3643    2.2532       NaN       NaN       NaN       NaN
    1.4796       NaN       NaN       NaN       NaN       NaN
```

```
[Bx,By] = deal(NaN(6));
for i = 0:5
    for j = 0:5-i
        D = (5-i-j)/5*T(1,:) + j/5*T(2,:) + i/5*T(3,:);
        Bx(i+1,j+1) = D(1);
        By(i+1,j+1) = D(2);
    end
end
[~,U3] = trimeshgrid(3);
bezier3(Bx,By,Bz,U3)
```

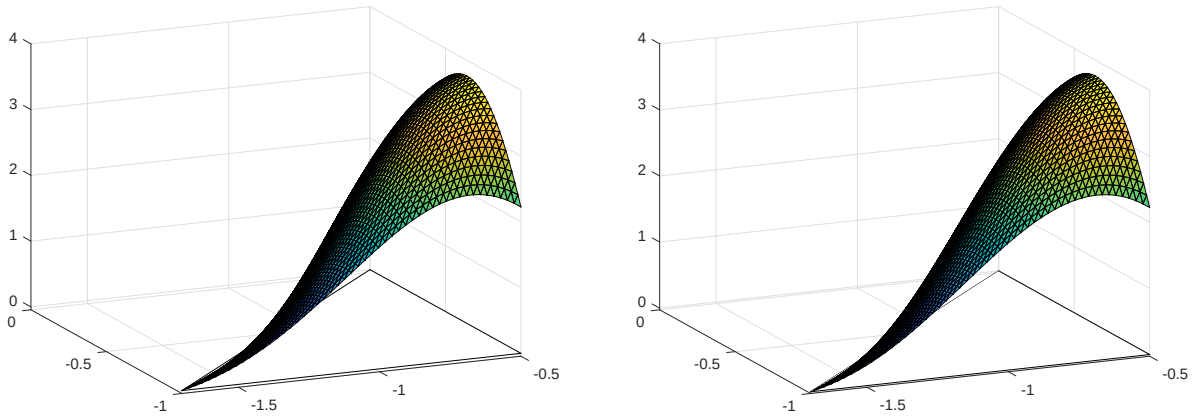
```
ans = 10x3
   -1.7000   -1.0000    0.0009
   -1.3000   -1.0000    0.8774
   -0.9000   -1.0000    2.2124
   -0.5000   -1.0000    2.2247
   -1.3000   -0.6667    0.0447
   -0.9000   -0.6667    2.2395
   -0.5000   -0.6667    3.7531
   -0.9000   -0.3333    0.6957
   -0.5000   -0.3333    2.9541
   -0.5000         0    1.4796
```

```
[TRI,U50] = trimeshgrid(50);
X = U50(:,1)*T(1,1) + U50(:,2)*T(2,1) + U50(:,3)*T(3,1);
Y = U50(:,1)*T(1,2) + U50(:,2)*T(2,2) + U50(:,3)*T(3,2);
Z = f(X,Y);
b = bezier3(Bx,By,Bz,U50);
```

```
% maksimalna absolutna napaka
max(max(abs(Z-b(:,3))))
```

```
ans = 0.1257
```

```
% izris
trisurf(TRI,X,Y,Z);
trisurf(TRI,b(:,1),b(:,2),b(:,3));
```



(a) Funkcija `peaks` (b) Argyrisov polinom

Slika 10: Aproksimacija z Argyrisovo shemo nad trikotnikom.

Naloga 18. *Argyrisov zlepek.*

Argyrisova interpolacijska shema je zasnovana tako, da jo z enega trikotnika enostavno razširimo na množico trikotnikov, ki določajo triangulacijo $\triangle$ domene $\Omega \subset \mathbb{R}^2$. Naj bo $\mathcal{V}$ množica točk triangulacije (to je množica oglišč trikotnikov v triangulaciji). Dvakrat odvedljivo funkcijo $f : \Omega \to \mathbb{R}$ lahko aproksimiramo tako, da konstruiramo interpolacijski zlepek iz prostora

$$\left\{ s \in C^1(\Omega) \cap C^2(\mathcal{V});\ s|_T \in \mathbb{P}_5^2,\ T \in \triangle \right\}.$$

Tak zlepek je nad vsakim trikotnikom $T \in \triangle$ določen po Argyrisovi shemi.

1. Sestavite metodo `argyrisspline`, ki izračuna predstavitev Argyrisovega zlepka v Bézierjevi obliki nad vsakim trikotnikom triangulacije posebej. Triangulacijo v Matlabu predstavimo z razredom `triangulation`. Objekt `tri` tega razreda je določen s tabelo točk triangulacije `tri.Points` (posamezna vrstica te tabele določa kartezične koordinate točke v triangulaciji) in tabelo trikotnikov triangulacije `tri.ConnectivityList` (posamezna vrstica te tabele določa indekse točk, ki predstavljajo trikotnik v triangulaciji).

   Opis metode:

```
function S = argyrisspline(tri,f,Df,Hf)
% Opis:
%  argyrisspline izračuna Bezierjeve predstavitve polinomov, ki
%  določajo Argyrisov zlepek nad triangulacijo
%
% Definicija:
%  S = argyrisspline(tri,f,Df,Hf)
%
% Vhodni podatki:
%  tri  objekt razreda triangulation, ki določa triangulacijo
%       domene, nad katero aproksimiramo funkcijo f,
%  f    funkcija, ki jo interpoliramo,
%  Df   gradient funkcije, ki jo interpoliramo,
%  Hf   Hessejeva matrika funkcije, ki jo interpoliramo
%
% Izhodni podatek:
%  S    celica z dolžino, ki ustreza številu trikotnikov v
%       triangulaciji, v kateri vsak element vsebuje matriko
%       velikosti 6 x 6, ki predstavlja koeficiente Argyrisovega
%       polinoma v Bezierjevi obliki
end
```

2. Izračunajte Argyrisov zlepek za funkcijo

$$f(x,y) = 3(1-x)^2\, e^{-x^2-(y+1)^2} - 10\left(\tfrac{x}{5} - x^3 - y^5\right) e^{-x^2-y^2} - \tfrac{1}{3} e^{-(x+1)^2-y^2},$$

   ki predstavlja ukaz `peaks` v Matlabu, nad triangulacijo, ki je podana z naslednjimi ukazi.

```
X = [-3.0; -3.0; -3.0; -3.0; -1.7; -1.6; -1.5; -1.1; -0.9; ...
    -0.7; -0.5; -0.5; -0.5; -0.4; 0.0; 0.5; 0.5; 0.6; 0.7; ...
    1.0; 1.2; 1.2; 1.6; 2.0; 2.0; 3.0; 3.0; 3.0; 3.0; 3.0];
Y = [-3.0; -2.0; 1.0; 3.0; -1.0; 0.5; 1.7; -3.0; -2.0; 1.0; ...
    -1.0; 0.0; 3.0; 1.6; -2.0; -1.0; 0.5; 1.2; -0.4; 2.2; ...
    -3.0; -1.6; 3.0; -1.0; 0.6; -3.0; -1.8; -0.4; 1.7; 3.0];
TRI = delaunay(X,Y);
tri = triangulation(TRI,X,Y);
```

Narišite graf zlepka in izračunajte maksimalno absolutno napako aproksimacije f v točkah $(6\frac{i}{100} - 3, 6\frac{j}{100} - 3)$, $i, j = 0, 1, \ldots, 100$. Pri izračunu vrednosti zlepka si lahko pomagate z vgrajeno funkcijo `pointLocation`, s pomočjo katere za posamezno točko določite, v katerem trikotniku triangulacije leži in kakšne so njene baricentrične koordinate glede na ta trikotnik.

Primeri:

```
S = argyrisspline(tri,f,Df,Hf);
```

```
S{1}
```

```
ans = 6x6
    0.6385    1.1450    2.0399    3.7117    5.6333    6.9058
    1.1748    2.0231    3.5167    5.2064    6.6443       NaN
    1.5520    2.1733    3.2639    3.7285       NaN       NaN
    0.8422    1.0190    1.1946       NaN       NaN       NaN
    0.4497    0.5215       NaN       NaN       NaN       NaN
    0.2335       NaN       NaN       NaN       NaN       NaN
```

```
S{2}
```

```
ans = 6x6
    0.6385    0.5265    0.0688   -0.9796   -1.8102   -2.2704
    0.9727    0.6615   -0.4504   -1.7560   -2.5124       NaN
    1.4262    0.5971   -1.3054   -2.3817       NaN       NaN
    1.9314    0.0280   -1.4780       NaN       NaN       NaN
    1.9585    0.0464       NaN       NaN       NaN       NaN
    1.7814       NaN       NaN       NaN       NaN       NaN
```

```
S{3}
```

```
ans = 6x6
   -0.8857    0.4861    0.4027    0.0091    0.0025    0.0007
   -0.6412    0.1740    0.4919    0.0080    0.0023       NaN
   -0.2175   -0.0410    0.0609    0.0067       NaN       NaN
   -0.0997    0.0030    0.0358       NaN       NaN       NaN
   -0.0421    0.0025       NaN       NaN       NaN       NaN
   -0.0176       NaN       NaN       NaN       NaN       NaN
```

```
S{12}
```

```
ans = 6x6
    1.4796    1.8336    1.3259   -0.4747   -0.3181    0.3754
    0.5579    1.2864    1.3031   -0.2920   -0.1495       NaN
   -0.1902    0.5315    0.9501    0.5700       NaN       NaN
   -0.6485    0.8456    2.5590       NaN       NaN       NaN
    0.4978    2.3421       NaN       NaN       NaN       NaN
    1.7814       NaN       NaN       NaN       NaN       NaN
```

```
S{15}
```

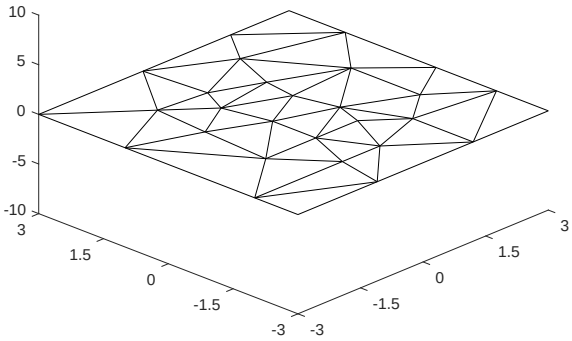
```
ans = 6x6
   -0.8857   -0.4770    0.3111    0.8655    0.4398    0.0009
    0.4861    0.7585   -0.1380   -0.0441    0.0262       NaN
    0.4027    0.1261    0.1083    0.0674       NaN       NaN
    0.0091    0.0071    0.0052       NaN       NaN       NaN
    0.0025    0.0019       NaN       NaN       NaN       NaN
    0.0007       NaN       NaN       NaN       NaN       NaN
```

```
% izračun vrednosti zlepka nad mrežo točk
N = 101;
[X,Y] = meshgrid(linspace(-3,3,N));
[txy,U] = pointLocation(tri,[X(:) Y(:)]);
SXY = zeros(N);
k = 1;
for i = 1:N
    for j = 1:N
        SXY(j,i) = decasteljau3(S{txy(k)},U(k,:));
        k = k+1;
    end
end
```

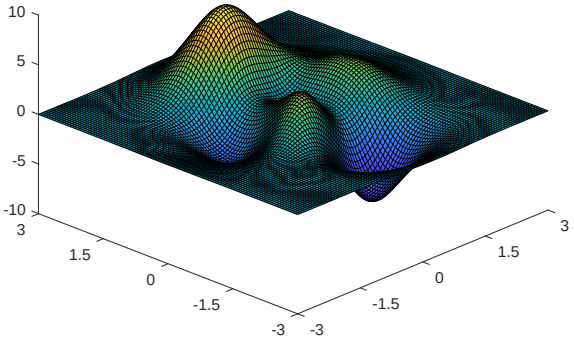
```
% maksimalna absolutna napaka
max(max(abs(f(X,Y)-SXY)))
```

```
ans = 0.2832
```

```
% izris
triplot(tri, 'Color','k');
surf(X,Y,SXY);
```



(a) Triangulacija kvadrata $[-3,3] \times [-3,3]$



(b) Argyrisov zlepek za funkcijo `peaks`

Aproksimacija z Argyrisovo shemo nad triangulacijo.