Zadatak 1.

Zadane su realne funkcije $f(x)=\ln^2(x+1)$, $g(y)=e^{\sqrt[3]{1-y^2}}$ i $h(t)=-\sin^3\left(\frac{\pi}{4}\cdot t\right)$. Izračunajte $f(0)+g(1)+h(2)$. (Ne smijete mijenjati oznake nezavisnih varijabli!)

Zadatak 2.

Zadane su realne funkcije $f(y)=e^{y+1}$ i $g(y)=\ln y-1$. Odredite pravila kompozicija $h_1=f\circ g$ i $h_2=g\circ f$. Što uočavate?

Zadatak 3.

Zadane su realne funkcije $f(x)=\sqrt[3]{x+1}-1$ i $g(x)=e^x-1$. Odredite pravilo funkcije $h=(f\circ g)^{-1}$, pa prikažite tu funkciju grafički.

Zadatak 4.

Zadani su polinomi $p_1(x)=x^2+x+1$ i $p_2(x)=-x^3+2\cdot x-1$.

a) Odredite pravilo polinoma $p_3=p_1+p_2+p_1\cdot p_2$, pa izračunajte $p_3(-1)$.

b) Zapišite racionalnu funkciju $f=\frac{p_2}{p_1}$ kao zbroj polinoma i prave racionalne funkcije.

Zadatak 5.

Zadani su polinomi $p_1(x)=x^3-x^2+x-1$ i $p_2(x)=1-x$. Neka su $p_3=p_1\circ p_2$ i $p_4=p_2\circ p_1$. Grafički odredite skup svih nultočaka polinoma p_3-p_4.

Zadatak 6.

Zadani su polinomi $p_1(x)=x^8-17\cdot x^4+16$ i $p_2(x)=x^6-81\cdot x^2$. Odredite skup svih:

a) (realnih i kompleksnih) rješenja jednadžbe $p_1(x)=0$;
b) nultočaka polinoma p_1;
c) polova racionalne funkcije $f=\frac{p_1}{p_2}$.

Zadatak 7.

Odredite polinom p četvrtoga stupnja u varijabli x kojemu je vodeći koeficijent jednak 2, a skup svih rješenja jednadžbe $p(x)=0$ u skupu $\mathbb{C}$ jednak $S=\{-3,-1,2,4\}$.

Zadatak 8.

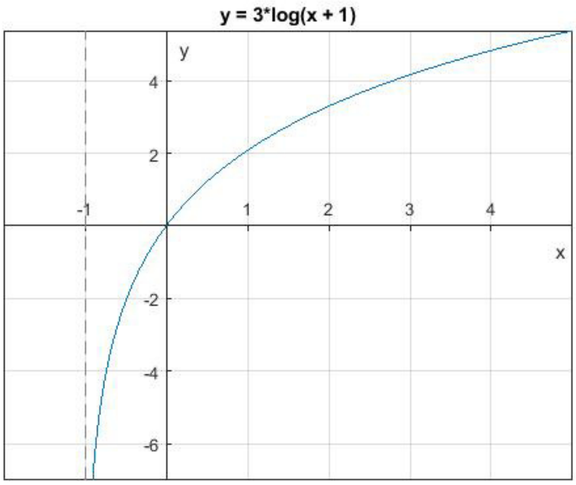
Graf polinoma q drugoga stupnja u varijabli u prolazi točkama $A=(-1,-4)$, $B=(1,-6)$ i $C=(4,6)$. Odredite pravilo toga polinoma, nacrtajte njegov graf i na njemu istaknite sve tri zadane točke.

Zadatak 9.

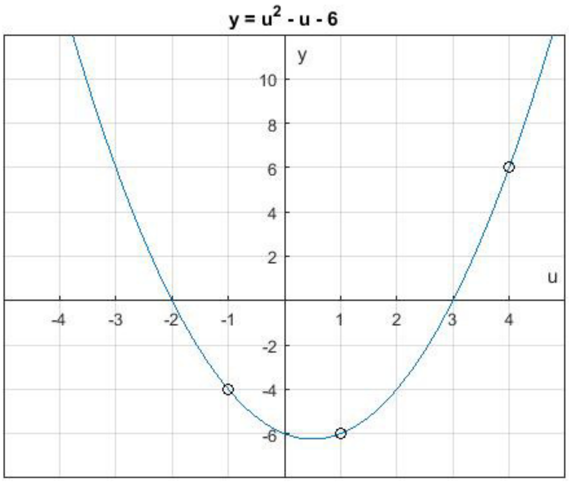
Odredite pravilo polinoma r trećega stupnja u varijabli t koji prolazi točkama $A=(-2,-42)$, $B=(-1,-14)$, $C=(1,0)$ i $D=(2,10)$, nacrtajte njegov graf na segmentu [-3, 3] i na njemu istaknite sve četiri zadane točke.

Rezultati zadataka:

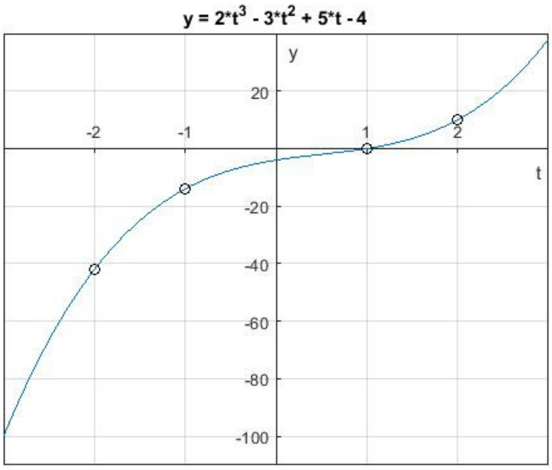
1. 0.
2. Obje kompozicije trebaju biti jednake identiteti. Doista, $h_1(y)=y$. Međutim, $h_2(y)=\ln\left(e^{y+1}\right)-1$. Primijetite da je $h_2(y)=\ln(e^{y+1})-1=(y+1)-1=y,\ \forall y\in\mathbb{R}$.
3. $h^{-1}(x)=3\cdot\ln(x+1)$. Graf dobivene funkcije prikazan je na slici 1.
4. **a)** $p_3(t)=-x^5-x^4+2\cdot x^2+4\cdot x-1.\quad p_3(-1)=-3.$.
b) $f(x)=-x+1+\dfrac{2\cdot x-2}{x^2+x+1}$.
5. $-1,2.$
6. **a)** {−2, −1, 1, 2, −2 · *i*, −*i*, *i*, 2 · *i*}; **b)** {1, −1, 2, −2}; **c)** {−3, 0, 3}.
7. $2\cdot x^4-4\cdot x^3-26\cdot x^2+28\cdot x+48.$
8. $u^2-u-6.$ Graf polinoma q prikazan je na slici 2.
9. $2\cdot t^3-3\cdot t^2+5\cdot t-4.$ Graf polinoma r prikazan je na slici 3.



Slika 1.



Slika 2.



Slika 3.

Komentari/objašnjenja programskih kodova

z1.m

Pogledajmo najprije kako bismo ovaj zadatak riješili koristeći anonimne funkcije. U računanju vrijednosti anonimne funkcije u nekoj točki koristimo ugrađenu funkciju `feval`. Njezini argumenti su redom anonimna funkcija i točka u kojoj treba izračunati vrijednost te funkcije. Nije dozvoljena promjena poretka ovih argumenata.

Programski kod glasio bi ovako:

```
f=@(x) log(x+1)^2;
g=@(y) exp((1-y^2)^(1/3));
h=@(t) -sin(pi/4*t)^3;
rezultat=feval(f,0)+feval(g,1)+feval(h,2)
```

U svakom retku zasebno deklarirali smo anonimnu funkciju, pa smo u četvrtom retku izračunali traženi zbroj vrijednosti tih funkcija u zadanim točkama.

Alternativni način rješavanja ovoga zadatka je pomoću tzv. *simboličkih varijabli.* Prednost toga načina je da u jednom retku popišemo sve simboličke varijable koje ćemo koristiti u programskom kodu. Pritom moramo paziti da u nekom retku koda neku drugu vrstu varijable (npr. numeričku (*double*) varijablu) ne označimo istim slovom kao i simboličku varijablu jer MATLAB nakon izvršenja toga retka koda navedenu varijablu više neće tretirati kao simboličku.

Opišimo ukratko što radi naš kod. U prvom retku popisali smo sve simboličke varijable koje ćemo koristiti. U ovom slučaju to su oznake argumenata funkcija: x, y i t. Da bi MATLAB znao da je riječ o simboličkim varijablama, ispred oznaka tih argumenata koristimo ugrađenu funkciju `syms`. Iz zapisa u 1. retku koda ne čini se da je riječ o ugrađenoj funkciji jer uz tu funkciju ne piše nijedan argument. Međutim, zapis

```
syms t x y
```

je ekvivalentan troretčanom zapisu

```
t=syms('t')
x=syms('x')
y=syms('y')
```

iz kojega je bitno vidljivije da se radi o ugrađenoj funkciji. Da bismo izbjegli korištenje troretčanoga (duljega) zapisa, u pravilu koristimo prvotno navedeni ekvivalentan (skraćeni) zapis.

U retcima 2. – 4. zadajemo funkcije *f*, *g* i *h*. Uočimo način na koji ih zadajemo: najprije napišemo oznaku („ime“) funkcije, potom znak = i naposljetku pravilo funkcije. U Vježbi 2. smo u analognim izrazima pisali konkretne brojeve, dok ovdje umjesto tih brojeva pišemo oznaku varijable.

Oprez: Najčešća greška korisnika je napisati npr. `f(x)=` (…). Prisjetimo se da se ovakav tip zapisa koristi kod rada s matricama kad želimo „dohvatiti“ element na nekoj poziciji. Nadalje, sve funkcije pozivaju se upravo zapisom oblika `f(x)`, pri čemu se umjesto `x` navede konkretna vrijednost argumenta, pa se tim zapisom nikako ne može *deklarirati* neka funkcija. Dakle, iako je u matematici potpuno korektno napisati npr. $f(x)=x+1$, u MATLAB-u doslovno ovakav zapis ne možemo koristiti.

Nakon što smo zadali sve simboličke funkcije, treba izračunati vrijednost svake pojedine funkcije u konkretnoj točki. U tu svrhu nam služi ugrađena funkcija `subs`. Ona najčešće ima točno dva argumenta: simboličku funkciju i točku u kojoj treba izračunati vrijednost te funkcije. (Redoslijed argumenata ne smijemo mijenjati.) Što radi ta funkcija? Ona uzima simboličku funkciju, pronalazi njezinu varijablu (koja prethodno mora biti deklarirana pomoću funkcije `syms`), zamjenjuje se točkom, odnosno konkretnim realnim brojem, te računa vrijednost dobivenoga numeričkoga izraza. Iako je ta vrijednost najčešće realan ili kompleksan broj, varijabla koja označava dobiveni rezultat je *simbolička* varijabla. To znači da je MATLAB „ne shvaća“ kao „normalan“ realan broj. Rad sa brojevima koji su zapisani kao simboličke varijable u nekim je slučajevima bitno sporiji u odnosu na rad s „običnim“ realnim brojevima (zapisanima kao varijable tipa `double`), pa u takvim slučajevima – radi ubrzanja rada programa – treba pribjeći radu s „običnim“ realnim brojevima.

Pitanje za razmišljanje: Riješite zadatak **bez** korištenja simboličkih i anonimnih funkcija. Usporedite sličnosti i razlike među dobivenim programskim kodovima.

Pitanje za razmišljanje: Za funkciju *f* definiranu u ovom zadatku izračunajte $f(1)$ koristeći odgovarajuću simboličku/anonimnu funkciju i odgovarajući numerički izraz. Usporedite sličnosti i razlike među dobivenim rezultatima.

z2.m

U ovom zadatku upoznat ćemo ugrađenu funkciju `compose` namijenjenu određivanju pravila kompozicije funkcija, te ugrađenu funkciju `simplify` koja pojednostavljuje simboličke algebarske izraze. Potonja funkcija je vrlo korisna jer omogućuje značajno pojednostavljivanje različitih algebarskih izraza na temelju bogate kolekcije različitih identiteta pohranjenih u svojoj biblioteci.

Iz *Matematike* 1 znamo da pri određivanju kompozicije funkcija treba biti vrlo oprezan jer ta operacija nije komutativna. Drugim riječima, pri određivanju kompozicije dviju funkcija bitan je poredak tih funkcija. Potpuno analogna tvrdnja vrijedi i za funkciju `compose`. Ona ima dvije ulazne varijable i to su funkcije čiju kompoziciju tražimo. Pravilo te funkcije je izlazna varijabla.

Funkcija `simplify` u svojemu najjednostavnijemu obliku ima samo jedan argument: simbolički izraz ili oznaku simboličkoga izraza kojega treba pojednostavniti. Međutim, vidjet ćemo da nam u nekim situacijama to nije dovoljno, odnosno da je korisno dodati još neke argumente te funkcije kako bi krajnji rezultat bio što jednostavniji.

Sama ideja zadatka je vrlo jednostavna. Najprije deklariramo nezavisnu varijablu (y), potom zadamo funkcije f i g, odredimo kompozicije $f \circ g$ i $g \circ f$ koristeći funkciju `compose`, pojednostavnimo pravila dobivenih funkcija i ispišemo rezultate. Ti rezultati su očito međusobno različiti i čini nam se da je zadatak gotov.

Riješimo li isti zadatak analitički (bez MATLAB-a), u jednom ćemo slučaju dobiti drugačiji rezultat. Naime, funkcije f i g su međusobno inverzne. Prema definiciji inverza bijekcije, to znači da vrijede jednakosti:

$$f \circ g = \mathrm{id}_{D(g)},$$
$$g \circ f = \mathrm{id}_{D(f)},$$

pri čemu je s id_S označena identiteta na skupu S, tj. funkcija $id(x) = x,\ \forall x \in S,$ a s D prirodna domena funkcije. Tek sad uočavamo da, prilikom pojednostavljivanja pravila funkcije h_2, MATLAB nije pojednostavnio njezino pravilo najviše što je moguće. Zašto? Naime, prema polaznim postavkama, MATLAB ne pojednostavljuje izraz koji sadrži potencije i logaritme jer njihova „kombinacija“ nije valjana za tzv. generičke kompleksne vrijednosti. (Za detaljnija pojašnjenja trebali bismo razmatrati neke pojmove analize kompleksnih funkcija kompleksne varijable, a to prelazi okvire ovoga predmeta.)

Nakon ispisa matrice H koja sadrži funkcije $f \circ g$ i $g \circ f$ čija su pravila pojednostavljena što je više moguće, MATLAB se zaustavio i čeka da pritisnemo *Enter* na tipkovnici. Ovo potonje postižemo korištenjem ugrađene funkcije `pause` koja, analogno kao i funkcija `clc` kojom brišemo komandni prozor, nema nijedan argument. Zašto se MATLAB zaustavio i što nas čeka u sljedećem dijelu programskoga koda?

Iako smo dobili pravilo funkcije $g \circ f$ koje nije pojednostavljeno najviše što je moguće, to je pravilo ipak moguće dodatno pojednostavniti tako da dobijemo ispravno rješenje. Funkcija `simplify` ima vrlo korisnu opciju `IgnoreAlgebraicConstaints` (engl.: zanemarite algebarske uvjete). Kako kaže sam naziv te opcije, ona omogućuje zanemarivanje algebarskih uvjeta poput gore spomenute nevaljane „kombinacije“ potencije i logaritama. U desetom retku koda vidimo kako izgleda sintaksa „proširene“ funkcije `simplify` u slučaju kad želimo zanemariti algebarske uvjete. Izvršenjem toga retka koda dobivamo željeni rezultat, tj. pravilo funkcije $g \circ f$ „pojednostavljeno do kraja“.

MATLAB se ponovno zaustavio i čeka da pritisnemo *Enter*. Razlog za ovo, posljednje zaustavljanje je sljedeći. Možemo li na još koji način provjeriti da su funkcije f i g međusobno inverzne? Odgovor je potvrdan. MATLAB ima ugrađenu funkciju `finverse` koja određuje pravila inverza bijekcije. Jedini argument te ugrađene funkcije je pravilo bijekcije čiji inverz tražimo. Dakako da potom trebamo pojednostavniti dobiveni algebarski izraz najviše što možemo, pa je zbog toga u retcima 14. i 15. korištena funkcija `simplify`. Izvršavanjem tih redaka koda vidimo da je inverz funkcije f jednak funkciji g i obratno.

Pitanja za razmišljanje: Pojavljuje li se problem zanemarivanja algebarskih uvjeta ako najprije djeluje logaritamska funkcija, a potom eksponencijalna? Konkretno, možemo li pojednostavniti izraz $e^{\ln x}$ bez zanemarivanja algebarskih uvjeta? U čemu je osnovna razlika između funkcija $f(x) = e^{\ln x}$ i $g(x) = \ln\left(e^{x}\right)$ kad znamo da su oba izraza jednaka identiteti? (*Uputa*: Odredite prirodnu domenu svake funkcije.) Precizno obrazložite svoje odgovore.

z3.m

Primijetimo da su zadane funkcije bijekcije (što su im domene, a što slike?). To znači da je i kompozicija $f \circ g$ bijekcija (što joj je domena, a što slika?), pa postoji njezin inverz i on je jedinstven. Ma koliko god vam se činilo suvišno i nepotrebno, ako nije prekomplicirano, prije rješavanja zadatka je uvijek dobro provesti ovakvu analizu kako bismo unaprijed znali što možemo očekivati, odnosno uz kakve početne pretpostavke (ponajprije o domeni i slici svake funkcije) provodimo „račun“ u MATLAB-u.

I u ovom zadatku određujemo kompoziciju potencije i logaritamske funkcije. S obzirom na napomene iz rješenja prethodnoga zadatka, možemo očekivati male probleme s pojednostavljivanjem pravila inverza. Početak koda je standardan: najprije deklariramo nezavisnu varijablu, potom zadajemo pravila funkcija f i g, pa u 4. retku određujemo pravilo funkcije $(f \circ g)^{-1}$ koristeći funkciju `finverse` i pojednostavljujemo ga što više možemo koristeći funkciju `simplify`. Zbog kompozicije potencije i logaritamske funkcije, uključili smo opciju zanemarivanja algebarskih uvjeta kako bi pravilo funkcije $(f \circ g)^{-1}$ bilo pojednostavljeno što je više moguće.

Ostatak koda je manje-više standardan. Graf dobivene funkcije crtamo koristeći funkciju `fplot`, postavljamo koordinatni sustav tako da se koordinatne osi sijeku u točki (0, 0), ispisujemo oznaku svake osi, te – kao naslov grafa – ispisujemo jednadžbu odgovarajuće ravninske krivulje.

Pitanja za razmišljanje: Što bi se dogodilo da u 4. retku nismo zanemarili algebarske uvjete? Kako bi u tom slučaju glasila jednadžba ravninske krivulje dobivene izvršavanjem 5. retka koda? Usporedite dobivene algebarske izraze. Ako ih izjednačimo, hoćemo li dobiti identitet koji vrijedi za svaki $x \in D\left((f \circ g)^{-1}\right)$? Precizno objasnite svoje odgovore.

z4.m

U ovom zadatku naučit ćemo kako u MATLAB-u relativno jednostavno možemo raditi s polinomima. Kako znamo, polinomi pripadaju u najjednostavnije matematičke funkcije. Upravo zbog toga se brojne složene matematičke funkcije nastoje dovoljno dobro aproksimirati polinomima na intervalu oko neke točke. Time ćemo se baviti nešto kasnije i u *Matematici* 2 i u ovom predmetu. Zbog toga je u MATLAB-u polinom izdvojen u posebnu „kategoriju“ funkcija. Istaknimo da s polinomima možemo postupati kao i sa svim ostalim simboličkim/anonimnim funkcijama. Međutim, neke značajne operacije s njima možemo izvesti brže i jednostavnije ako polinome poistovjetimo s određenim matricama. Zbog toga ćemo najprije opisati matrični zapis polinoma.

Osnova za matrični zapis polinoma je bijekcija koja svakom polinomu $p(x) = a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + \ldots + a_1 \cdot x + a_0$ pridružuje jednoretčanu matricu $A = \begin{bmatrix} a_n & a_{n-1} & \cdots & a_1 & a_0 \end{bmatrix}$. (Dokažite da je navedeno pridruživanje uistinu bijekcija.) Odmah uočimo da je matrica A tipa $(1, n + 1)$, tj. da je ukupan broj njezinih stupaca točno za jedan veći od stupnja polinoma p.

Oprez: Najčešća pogreška je poistovjećivanje ukupnoga broja stupaca matrice A sa stupnjem polinoma p. Kako vidimo, u matrici A su pohranjeni svi *koeficijenti* polinoma p, a njihov je ukupan broj za jedan veći od stupnja polinoma p (jer upisujemo i slobodni član, tj. koeficijent uz x^0). Također, moramo pripaziti i na *poredak* koeficijenata u matrici A. Element $A(1)$ odgovara koeficijentu uz x^n, element $A(2)$ odgovara koeficijentu uz x^{n-1}, te općenito $A(k)$ odgovara koeficijentu uz x^{n+1-k}, za svaki $k = 1, 2, \ldots, n+1$.

Primjedba 1. Ako elemente u matrici A upišemo tako da $A(k)$ odgovara koeficijentu uz x^{k-1}, za svaki $k = 1, 2, \ldots, n+1$, onda gornje svojstvo očito ne vrijedi i s takvom matricom dalje ne možemo raditi ništa. Međutim, u ovakvim slučajevima postoji vrlo jednostavna „pomoć“ u vidu ugrađene funkcije `flip`. Ta funkcija „okreće“ matricu A tako da njezin prvi element postaje posljednji, njezin drugi element postaje pretposljednji itd. Na taj način element $A(k)$ prelazi na poziciju $n+2-k$, pa dobivamo novu jednoretčanu matricu koja ima gornje svojstvo.

Napomena: Za množenje dvaju polinoma zapisanih u matričnom obliku koristimo ugrađenu funkciju `conv`. Ona ima dva argumenta, i to su (jednoretčane) matrice pridružene faktorima. Kao rezultat, funkcija `conv` ispisuje (jednoretčanu) matricu pridruženu umnošku tih polinoma. Znamo da je množenje polinoma komutativna operacija, pa pri korištenju funkcije `conv` ne trebamo paziti na poredak faktora.

Osnovni nedostatak zapisa polinoma u matričnom obliku vidimo upravo u ovom zadatku. Znamo da *bilo koja* dva polinoma možemo zbrojiti ili oduzeti *bez obzira na njihov stupanj*. Ako je npr. prvi pribrojnik stupnja 2020, a drugi stupnja 202, zbroj će biti polinom stupnja 2020. Međutim, analogno pravilo **ne vrijedi** za jednoretčane matrice. Ako npr. matrica A ima 2020 stupaca, a matrica B 202 stupca, onda **ne postoji** zbroj $A + B$ (i analogno za razliku). MATLAB nema ugrađenu funkciju za zbrajanje i oduzimanje polinoma zapisanih u matričnom obliku. Zbog toga ćemo zbrajanje i oduzimanje polinoma uvijek provoditi kao i za sve ostale funkcije, tj. zadajući polinome kao simboličke objekte.

Za razliku od zbrajanja, oduzimanja i množenja polinoma, kod kojih je rezultat uvijek polinom, prilikom dijeljenja polinoma kao rezultat općenito dobijemo količnik q i ostatak r (tim redoslijedom) Drugim riječima, operacija dijeljenja zapravo ima dvije izlazne varijable: količnik i ostatak nastali pri tom dijeljenju. U tu se svrhu koristi ugrađena funkcija `deconv`. Ona ima dvije ulazne varijable: djeljenik i djelitelj **zapisane u matričnom obliku**, te dvije izlazne varijable: količnik i ostatak pri dijeljenju **zapisane u matričnom obliku**. Dakle, osnovni preduvjet za primjenu funkcije `deconv` jest da su djeljenik i djelitelj zapisani u matričnom obliku.

Pogledajmo kako je strukturiran naš programski kod. Polinom p_3 je zbroj triju polinoma, pa zbog toga nemamo izbora: zadane polinome moramo zadati kao i sve ostale simboličke funkcije. To je razlog zašto u prvom retku deklariramo simboličku varijablu, a u drugom i trećem pravila zadanih polinoma analogno kao u rješenjima prethodnih zadataka.

U četvrtom retku određujemo pravilo polinoma p_3. Ideja određivanja je jasna: zbrajanje polinoma označavamo znakom +, množenje znakom *, pa nakon unosa izraza kojim se dobiva pravilo polinoma p_3 pojednostavljujemo dobiveni izraz koristeći funkciju `simplify`. Međutim, i ovdje postoji jedna „kvaka". Funkcija `simplify` ne mora ispisati polinom u standardnom obliku (u kojemu su eksponenti poredani silazno), a u nekim slučajevima može čak i faktorizirati izraz. Da bismo izbjegli ovu „kvaku" koristimo ugrađenu funkciju `collect`. Ona će omogućiti „skupljanje" koeficijenata uz iste potencije, te njihovo reduciranje i silazno slaganje eksponenata, tj. zapis u standardnom obliku. Funkcija `collect` najčešće se primjenjuje upravo pri radu s polinomima kao simboličkim objektima.

Peti redak ne donosi ništa novo: vrijednost polinoma u točki –1 računamo koristeći ugrađenu funkciju `subs` koju smo upoznali u zadatku 1.

U *Matematici* 1 smo naučili da se svaka neprava racionalna funkcija može zapisati u obliku zbroja polinoma i prave racionalne funkcije. Konkretno, ako je $f = \frac{p_2}{p_1}$ neprava racionalna funkcija, onda taj zapis glasi:

$$f = q + \frac{r}{p_1},$$

gdje su q i r redom količnik i ostatak pri dijeljenju polinoma p_2 polinomom p_1. Upravo te polinome (zapisane u obliku jednoretčanih matrica) dobit ćemo primjenom funkcije `deconv` (pazite na poredak ulaznih varijabli!). Međutim, kako smo istaknuli, da bismo mogli primijeniti tu funkciju, polinomi moraju biti zapisani u matričnom obliku. Naši polinomi to nisu. I što sad? Treba li ih ponovno zadavati i zapravo raditi dvostruki posao (zadavati isti polinom na dva različita načina)?

Nasreću, odgovor je negativan. MATLAB ima ugrađenu funkciju `sym2poly` koja polinom deklariran kao simbolički objekt „pretvara" u polinom zadan pomoću matrice svojih koeficijenata. Jedini argument te funkcije je pravilo ili oznaka polinoma deklariranoga kao simbolički objekt. Zbog toga u šestom i sedmom retku koda polinome p_1 i p_2 pretvaramo iz simboličkoga u matrični oblik. Dobivene rezultate označili smo gotovo jednako kao i polazne polinome uz sugestivni dodatak slova `m` kojim ukazujemo da se radi o matričnom zapisu polinoma.

Time smo ispunili potrebne preduvjete za primjenu funkcije `deconv`. Nju primjenjujemo u osmom retku koda, te dobivamo količnik q i ostatak r zapisane u matričnom obliku. Moramo li nužno „dešifrirati“ pravilo polinoma pomoću ranije definirane bijekcije ili možemo prepustiti MATLAB-u da taj posao obavi umjesto nas?

Odgovor je: možete postupiti kako želite. Nitko vam ne brani napisati pravilo polinoma iz njegova matričnoga oblika. Međutim, ako imate dovoljno povjerenja u MATLAB, onda možete koristiti njegovu ugrađenu funkciju `poly2sym` koja će polinom zapisan u matričnom obliku pretvoriti u simbolički objekt i, ako na kraju retka ne stavite znak ;, ispisati rezultate. Upravo to smo i učinili u posljednjim dvama retcima koda.

Pitanje za razmišljanje: Jesmo li $p_3(-1)$ mogli odrediti bez poznavanja pravila polinoma p_3, tj. koristeći samo pravila polinoma p_1 i p_2? Ako jesmo, kako? Precizno objasnite svoje odgovore.

Pitanje za razmišljanje: Kako provjeriti je li dobiveni prikaz ispravan, tj. vrijedi li jednakost $\frac{p_2}{p_1} = q + \frac{r}{p_1}$ ako su svi polinomi zadani u matričnom obliku? (*Uputa*: Pomnožite zadanu jednakost s p_1.)

Pitanje za razmišljanje: Razmislite o algoritmu kojim bi se moglo implementirati zbrajanje polinoma zadanih u matričnom obliku. Ideja je „manju“ matricu (matricu koja ima manje stupaca) proširiti nulama tako da obje matrice imaju jednako mnogo stupaca. Kako odrediti koliko nula treba dodati? Na koja mjesta ih treba zapisati?

z5.m

Ovaj zadatak zapravo ne donosi ništa posebno novoga. Pravilo kompozicije polinoma ne možemo odrediti ako su polinomi zadani u matričnom obliku, pa u prvom retku deklariramo simboličku varijablu (x), a u drugom i trećem retku pravila zadanih polinoma. Budući da je kompozicija dvaju polinoma također polinom, postupamo analogno kao u prethodnom zadatku: nakon primjene funkcije `compose`, dobiveni simbolički izraz pojednostavljujemo koristeći funkciju `simplify`, pa potom poredamo članove polinoma na uobičajeni način koristeći funkciju `collect`.

Koristeći ugrađenu funkciju `fplot` čiji argument može biti i simbolički objekt, crtamo graf polinoma $p_3 - p_4$. Ovdje je od posebne važnosti postaviti pravokutni koordinatni sustav tako da se koordinatne osi sijeku u ishodištu jer ćemo nultočke odrediti kao prve koordinate sjecišta dobivenoga grafa s osi apscisa. To je i učinjeno u retcima 7. – 13. U

14. je retku koda naslov grafa ispisan pomoću pretvorbe simboličkoga izraza u niz znakova, za što je iskorištena ugrađena funkcija `char`. Uočite da je pritom argument funkcije `title` dodatno stavljen unutar uglate zagrade.

Oprez: Kao što smo naučili u *Matematici* 1, nultočke polinoma su *realni brojevi* jer je prirodna domena svakoga polinoma skup $\mathbb{R}$. Skup svih nultočaka polinoma p je podskup skupa svih rješenja jednadžbe $p(x)=0$ jer se među tim rješenjima mogu naći i kompleksni brojevi koji nisu realni brojevi. Kako odrediti sva rješenja navedene jednadžbe (uključujući i kompleksna rješenja), vidjet ćemo nešto kasnije.

Pitanje za razmišljanje: Objasnite vezu svake određene nultočke polinoma $p_3 - p_4$ sa zajedničkim točkama grafova polinoma p_3 i p_4. Možete li, nakon što odredite navedene nultočke, odrediti i zajedničke točke tih grafova? Ako možete, kako? Precizno objasnite svoje odgovore.

z6.m

U ovom zadatku još jednom ćemo ponoviti osnovne pojmove vezane uz polinome. Najproblematičnije je razlikovati pojam nultočke polinoma p i pojam rješenja jednadžbe $p(x)=0$. Izrecimo ukratko osnovne tvrdnje:

Rješenja ili korijeni jednadžbe $p(x)=0$ su svi **kompleksni** brojevi koji zadovoljavaju tu jednadžbu. **Nultočke** polinoma p su svi **realni** brojevi koji zadovoljavaju jednadžbu $p(x)=0$. **Polovi** racionalne funkcije su **nultočke** njezina nazivnika.

U ovom ćemo zadatku koristiti ugrađenu funkciju `roots` koja se primjenjuje **samo za određivanje korijena** jednadžbe $p(x)=0$, gdje je p polinom. Jedina ulazna varijabla te funkcije jest polinom zapisan u **matričnom** obliku. (Kako odrediti korijene jednadžbe polinoma zapisanoga kao simbolički objekt, odnosno kako općenito riješiti jednadžbu $f(x)=0$, gdje je f bilo koja realna funkcija jedne realne varijable, vidjet ćemo u sljedećoj vježbi.) Funkcija vraća sve korijene jednadžbe $p(x)=0$ zapisane u **matričnom** obliku. Dakle, možemo zaključiti da je izlazna varijabla funkcije `roots` kompleksna matrica (matrica čiji su elementi kompleksni brojevi).

S obzirom na stupnjeve polinoma p_1 i p_2, bitno brže ih je zadati kao simboličke objekte. To je učinjeno standardnim postupkom u prvim trima retcima koda. Potom su korištenjem funkcije `sym2poly` zadani polinomi pretvoreni u matrični oblik. Razlog za to je primjena funkcije `roots` u šestom retku koda. Matrica u kojoj će biti pohranjeni svi korijeni jednadžbe $p_1(x) = 0$ označena je sugestivno s *rjesenja*. p_1 je polinom osmoga stupnja, pa će matrica *rjesenja* biti tipa (8, 1). Dakle, ta matrica će biti **jednostupčana** i sadržavat će točno 8 (ne nužno različitih) kompleksnih brojeva. Ponavljanje određenoga korijena ovisi o njegovoj kratnosti. Ako je x_0 korijen kratnosti *k*, tj. ako je $p(x)$ djeljiv sa $(x - x_0)^k$, ali ne i s $(x - x_0)^{k+1}$, onda će se x_0 u matrici *rjesenja* pojaviti ukupno *k* puta.

Pitanje za razmišljanje: Pretpostavimo da nas ne zanima kratnost pojedinoga korijena. Je li moguće „smanjiti“ matricu *rjesenja* tako da se svaki element te matrice u njoj pojavljuje točno jednom? Ako jest, kojim algoritmom to učiniti? Objasnite svoje odgovore.

Iz matrice *rjesenja* u 7. retku koda izdvajamo samo one korijene koji su realni brojevi. Uočite sintaksu ovoga retka: jednostavno smo prošli cijelom matricom *rjesenja*, ostavili u njoj samo elemente čiji je imaginarni dio jednak nuli (a to su upravo realni brojevi) i ddobivenu matricu pohranili pod nazivom *nultocke*. Što znamo o matrici *nultocke*? A priori ništa osim da su njezini elementi **realni** brojevi i da tih brojeva ima najviše osam (jer je ukupan broj nultočaka uvijek jednak ili manji od stupnja polinoma).

Potpuno analogan postupak proveli smo u posljednjim dvama retcima koda određujući polove racionalne funkcije, tj. nultočke polinoma p_2. Pritom nismo provjeravali je li racionalna funkcija potpuno skraćena, tj. imaju li brojnik i nazivnik zajedničkih korijena. Međutim, to nije teško naknadno učiniti utvrđivanjem postoji li kompleksan broj koji se pojavljuje u objema matricama.

Pitanje za razmišljanje: Neka su *A* i *B* zadane matrice. Odredite skup svih elemenata matrice *A* koji se pojavljuju u matrici *B*. Objasnite svaki pojedini redak koda.

z7.m

U ovom zadatku naučit ćemo kako na temelju podataka o skupu **svih korijena** jednadžbe $p(x) = 0$ (u koje, podsjetimo, ubrajamo i kompleksne brojeve koji nisu realni) možemo odrediti taj polinom. Uočite da se u zadatku (namjerno) ne govori o skupu svih nultočaka polinoma *p* jer je taj skup podskup skupa svih rješenja jednadžbe $p(x) = 0$. Međutim, samo na temelju podatka o skupu svih korijena jednadžbe $p(x) = 0$ ne možemo jednoznačno odrediti polinom *p*. Prvi problem se javlja već pri ispisivanju elemenata skupa svih korijena jednadžbe: ako poštujemo uobičajeno načelo da se pri ispisivanju skupa svaki njegov element navodi točno jednom, onda to načelo povlači da svaki korijen jednadžbe ima kratnost jednaku 1, što općenito ne mora biti točno. Ipak, u ovom zadatku nemamo tako kompliciran slučaj: tražimo polinom četvrtoga stupnja, skup S ima točno četiri elementa, pa zaključujemo da svaki korijen ima kratnost 1.

Međutim, čak i kad znamo skup svih rješenja jednadžbe $p(x)=0$ i da svako rješenje ima kratnost jednaku 1, polinom p još uvijek ne možemo jednoznačno odrediti. Npr. polinomi $p_1(x)=x^2-1$ i $p_2(x)=2\cdot x^2-2$ su međusobno različiti (kao funkcije), ali jednadžbe $p_1(x)=0$ i $p_2(x)=0$ imaju jednake skupove rješenja. Da bismo mogli jednoznačno odrediti polinom p, osim skupa svih rješenja jednadžbe $p(x)=0$ i kratnosti svakoga elementa toga skupa, moramo zadati i *vodeći koeficijent* polinoma, tj. koeficijent uz član s najvećim eksponentom. Preciznije, vrijedi sljedeća tvrdnja.

Tvrdnja 1. Neka su $n\in\mathbb{N}$, $a_n\in\mathbb{R}\setminus\{0\}$ i $S\subset\mathbb{C}$ takav da je $\text{card}(S)=n$. Tada postoji jedinstveni polinom $p:\mathbb{C}\to\mathbb{C}$ stupnja n kojemu je vodeći koeficijent jednak a_n, a skup svih rješenja jednadžbe $p(x)=0$ jednak S.

Napomena 1. Tvrdnja 1. obuhvaća i slučaj kad barem jedno rješenje jednadžbe $p(x)=0$ ima kratnost barem 2. U takvim slučajevima pri zadavanju skupa S svako rješenje čija je kratnost barem 2 pišemo točno onoliko puta kolika mu je kratnost. Dakle, ako je $p(x)=x^4-x^2$, onda je $S=\{-1,0,0,1\}$, a ne $S=\{-1,0,1\}$. Želimo li zadržati standardan zapis konačnih skupova u kojemu se svaki element pojavljuje točno jednom, onda dodatno moramo navesti podatak o kratnosti svakoga elementa skupa S.

Pogledajmo jednostavan programski kod koji je rješenje ovoga zadatka. U prvom retku deklariramo varijablu polinoma p. U drugom retku deklariramo skup S u obliku jednoretčane matrice. Broj njezinih stupaca jednak je broju elemenata skupa S.

Traženi polinom odredit ćemo koristeći ugrađenu funkciju `poly`. Jedina ulazna varijabla te funkcije je skup S. Funkcija će vratiti sve koeficijente normiranoga polinoma p (zapisane u obliku jednoretčane matrice) kojemu je skup svih rješenja jednadžbe $p(x)=0$ jednak S. (Podsjetimo, polinom p je *normiran* ako je njegov vodeći koeficijent jednak 1.) Međutim, u zadatku se traži da vodeći koeficijent polinoma p bude jednak 2, a ne 1. Zbog toga izlaznu jednoretčanu matricu trebamo pomnožiti s 2 i dobit ćemo koeficijente traženoga polinoma. Primjenom funkcije `sym2poly` pretvorit ćemo polinom u simbolički objekt. Naposljetku, koristeći funkciju `fprintf` i njezine opcije, taj polinom možemo i ispisati u uobičajenom obliku.

Pitanja za razmišljanje: Vrijedi li tvrdnja 1. ako umjesto vodećega koeficijenta navedemo neki drugi koeficijent polinoma p (npr. slobodni član)? Ako vrijedi, kako u tom slučaju odrediti polinom koristeći funkciju `poly`? Objasnite svoje odgovore.

z8.m

U ovom zadatku koristimo sljedeću tvrdnju.

Tvrdnja 2. Neka su $n \in \mathbb{N}$ i $S \subset \mathbb{R}^2$ takav da je $\text{card}(S) = n$. Tada postoji jedinstveni polinom p stupnja $n-1$ čiji graf prolazi svim elementima skupa S (shvaćenima kao točke u pravokutnom koordinatnom sustavu u ravnini).

U slučajevima $n \in \{2,3\}$ ovu tvrdnju smo primjenjivali u osnovnoj i srednjoj školi. Za $n = 2$ tvrdnja 2. zapravo kaže da je svaki pravac (graf polinoma 1. stupnja) jednoznačno određen zadavanjem bilo koje dvije svoje različite točke. Za $n = 3$ tvrdnja 2. zapravo kaže da je svaka parabola (graf polinoma 2. stupnja) jednoznačno određena zadavanjem bilo koje tri svoje različite točke.

U rješavanju ovoga zadatka koristit ćemo ugrađenu funkciju `polyfit`. Njezina je primjena bitno šira u odnosu na primjenu u rješenju ovoga zadatka. Grubo rečeno, ta funkcija vraća polinom *bilo kojega* (unaprijed zadanoga) stupnja koji najbolje aproksimira zadani skup podataka (točaka u ravnini) u smislu metode najmanjih kvadrata. Pokazuje se da je takav polinom također jedinstven, pa je funkcija `polyfit` dobro definirana. O tome se više uči u predmetu *Numerička matematika* u 4. semestru.

Osnovna ideja rješavanja ovoga zadatka je vrlo jednostavna. U prvom retku koda deklariramo varijablu polinoma (to je u) kako bismo kasnije iznad grafa mogli ispisati pravilo polinoma. U drugom retku zadajemo jednoretčanu matricu prvih koordinata zadanih točaka. Ta je matrica namjerno označena s x jer smo oznaku u „utrošili“ na označavanje nezavisne varijable. U trećem retku zadajemo jednoretčanu matricu y drugih koordinata zadanih točaka u istom poretku kao i prve koordinate (tj. tako da $(x(1),y(1))$ tvore jednu točku, $(x(2),y(2))$ drugu točku itd.).

Četvrti redak koda je zapravo najvažniji dio zadatka. Krenimo od njegova kraja. Primjećujemo da funkcija `polyfit` ima točno tri ulazne varijable. Prva od njih je matrica prvih koordinata, druga matrica drugih koordinata, a treća stupanj polinoma kojim treba aproksimirati zadani skup točaka. U našem je slučaju matrica prvih koordinata označena s x, matrica drugih koordinata s y, dok je stupanj traženoga polinoma jednak 2. Funkcija `polyfit` vratit će matricu koeficijenata traženoga polinoma.

Da bismo dalje mogli lakše raditi s tim polinomom, dobivenu matricu koeficijenata „prebacujemo“ u simboličku funkciju koristeći ugrađenu funkciju `poly2sym`. Rezultat primjene te funkcije je polinom p zapisan kao simbolička funkcija (tj. poput funkcija u zadatku 1.), a ne u matričnom obliku. Koristeći funkcije `simplify` i `collect` pojednostavljujemo pravilo dobivena polinoma i zapisujemo ga u uobičajenom obliku. Time smo ispunili sve potrebne preduvjete da na vrlo jednostavan način *istovremeno* možemo nacrtati graf dobivenoga polinoma i ispisati pripadno pravilo polinoma.

Retci 5. – 13. namijenjeni su upravo tome. Njihovim izvršavanjem će u pravokutnom koordinatnom sustavu u ravnini, čije se osi sijeku u ishodištu, biti prikazan graf dobivenoga polinoma (prema zadanim postavkama, na segmentu $[-5,5]$, uz graf će biti naznačene sve potrebne oznake, a kao naslov slike bit će ispisana jednadžba krivulje čija desna strana predstavlja upravo pravilo polinoma p. Analogno kao u rješenju zadatka 3., za ispis te jednadžbe pravilo polinoma p – koje je simbolička funkcija – moramo prethodno „pretvoriti“ u niz znakova koristeći ugrađenu funkciju `char`.

Preostaje na dobiveni graf ucrtati zadane točke. U tu svrhu „zamrznut“ ćemo graf koristeći opciju `hold on`, pa koristeći funkciju `plot` ucrtati zadane točke. Da bi one bile vidljivije, u okviru funkcije `plot` širinu linije (engl. *Linewidth*) podesit ćemo na 1.

Pitanja za razmišljanje: Može li se zadatak riješiti bez korištenja funkcije `polyfit`? Ako može, objasnite algoritam pomoću kojega biste riješili zadatak. Objasnite sve svoje tvrdnje. (*Uputa*: Kako biste zadatak riješili analitički, tj. „ručno“ odredili jednadžbu parabole koja prolazi zadanim točkama?)

z9.m

Ovaj zadatak ne donosi ništa novoga u odnosu na prethodni, osim što se traži polinom trećega stupnja. Zbog toga je i ideja rješavanja analogna onoj iz prethodnoj zadatka. Nakon deklariranja nezavisne varijable, u 2. i 3. retku zadajemo matrice x i y. Potom određujemo traženi polinom „udruženim djelovanjem“ funkcija `polyfit`, `simplify` i `collect`. U retcima 5. – 13. crtamo dobiveni polinom u pravokutnom koordinatnom sustavu u ravnini, te uz graf navodimo sve potrebne oznake (prilagođavajući raspon vrijednosti na osi ordinata ugrađenom funkcijom `ylim`). Potom „zamrznemo“ graf opcijom `hold on` kako bismo mogli ucrtati zadane četiri točke. Pritom ponovno primijenimo funkciju `plot` uz postavku širine linije na 1.

Pitanje za razmišljanje: Je li moguće obrnuti poredak izvršenja naredbi u kodu tako da najprije budu ucrtane zadane točke, pa nakon pritiskanja tipke *Enter* kroz te točke bude provučena dobivena ravninska krivulja? Ako jest, preradite kod tako da ispisuje ispravne rezultate. Objasnite sve svoje tvrdnje.

Domaća zadaća

1. Zadani su polinomi $p_1(t) = t^3 + t^2 - 2$ i $p_2(t) = t + 1$.

a) Zapišite nepravu racionalnu funkciju $f = \dfrac{p_1 \cdot p_2}{p_1 \circ p_2}$ kao zbroj polinoma i prave racionalne funkcije.

b) Odredite skup svih nultočaka i skup svih polova funkcije *f* iz **a)** podzadatka.

2. Odredite polinom *p* trećega stupnja u varijabli *x* kojemu je koeficijent uz x^2 jednak 4, a skup svih rješenja jednadžbe $p(x) = 0$ jednak $S = \{-2, -i, i\}$.

3. Nacrtajte parabolu čija jednadžba ima oblik $y = a \cdot x^2 + b \cdot x + c$, $a, b, c \in \mathbb{R}$, $a \neq 0$, koja prolazi točkama $A = (-1, -4)$, $B = (2, -1)$ i $C = (4, -9)$. Uz sliku navedite sve potrebne oznake i istaknite sve zadane točke.

4. Zadane su bijekcije $f(t) = 2 \cdot \arcsin\left(t^3 + 1\right) - \pi$ i $g(t) = \sqrt[3]{t-1}$. Odredite pravilo funkcije $h = \left(f \circ g\right)^{-1}$, pojednostavnite dobiveni izraz što više možete, pa nacrtajte graf funkcije *h* na segmentu $[-\pi, \pi]$. Uz graf navedite sve potrebne oznake.

5. Zadane su bijekcije $f(x) = e^{3 \cdot x} - 1$ i $g(x) = \ln(x+1)$. Odredite sva sjecišta grafa funkcije $h = g \circ f - f \circ g$ s osi apscisa. (Pretpostavite da je *h* definirana na svojoj prirodnoj domeni.)

	Matematički alati u elektrotehnici (preddiplomski stručni studij elektrotehnike)	Vježba 7. Simboličke varijable. Realne funkcije. Polinomi.
TEHNIČKO VELEUČILIŠTE U ZAGREBU POLYTECHNICUM ZAGRABIENSE Elektrotehnički odjel		

Rezultati zadataka za domaću zadaću

1. a) $f = t - 2 + \dfrac{4\cdot t^2 + 8\cdot t - 2}{t^3 + 4\cdot t^2 + 5\cdot t}$;

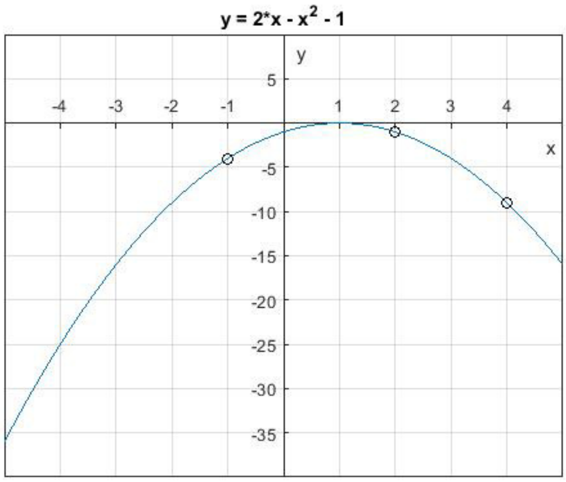
b) $N(f) = \{-1, 1\}$, $P(f) = \{0\}$.

2. $p(t) = 2\cdot t^3 + 4\cdot t^2 + 2\cdot t + 4$.

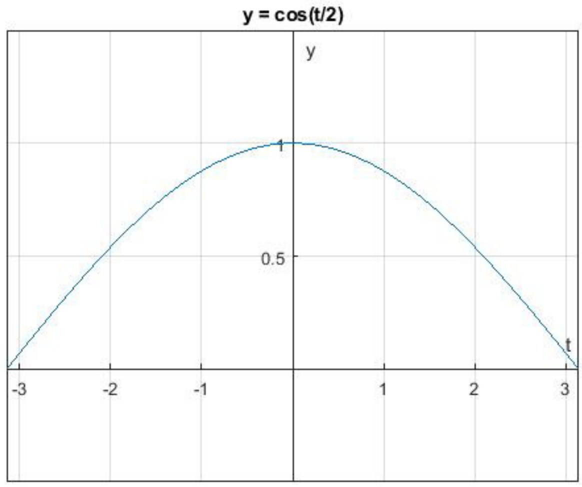
3. $y = -x^2 + 2\cdot x - 1$. Vidjeti sliku 1.

4. $h(t) = \cos\left(\dfrac{t}{2}\right)$. Vidjeti sliku 2.

5. $S_1 = (-3, 0)$, $S_2 = (0, 0)$.



Slika 1.



Slika 2.