TVŽ TEHNIČKO VELEUČILIŠTE U ZAGREBU POLYTECHNICUM ZAGRABIENSE Elektrotehnički odjel	**Matematički alati u elektrotehnici** (preddiplomski stručni studij elektrotehnike)	Vježba 11. Funkcijski redovi u MATLAB-u.

Napomena: Kad god je potrebno, pretpostavite da mantisa ima 6 znamenaka.

Zadatak 1.

a) Za svaki $x \in [-1,1\rangle$ izračunajte zbroj reda $\sum_{n=1}^{+\infty} \frac{x^n}{n}$.

b) Koristeći rješenje prethodnoga podzadatka izračunajte egzaktnu i približnu (s točnošću od 10^{-5}) vrijednost zbroja reda $\sum_{n=1}^{+\infty} \frac{1}{n \cdot 2^n}$.

c) Odredite sve $x \in [-1,1\rangle$ za koje je zbroj reda iz a) podzadatka jednak $-\ln 2$.

Zadatak 2.

a) Aproksimirajte funkciju $h(t) = 105 \cdot \mathrm{arctg}(2 \cdot t)$ MacLaurinovim polinomom M_7 stupnja 7.

b) Odredite apsolutnu i relativnu pogrešku aproksimacije vrijednosti $h(0.1)$ vrijednošću $M_7(0.1)$. Zapišite dobiveni rezultat u znanstvenom obliku.

c) Na **istoj** slici prikažite grafove funkcija h i M_7 na segmentu $[-0.5,\ 0.5]$.

d) Nacrtajte graf funkcije $g = h - M_7$ na segmentu $[-0.5,\ 0.5]$.

Zadatak 3.

a) Aproksimirajte funkciju $p(x) = \sqrt[3]{x+2}$ Taylorovim polinomom T_5 stupnja 5 oko točke $c = -1$.

b) Odredite apsolutnu i relativnu pogrešku aproksimacije vrijednosti $p(-0.5)$ vrijednošću $T_5(-0.5)$. Zapišite dobiveni rezultat u znanstvenom obliku.

c) Na **istoj** slici prikažite grafove funkcija p i T_5 na segmentu $[-1.5,-0.5]$.

d) Nacrtajte graf funkcije $g = p - T_5$ na segmentu $[-1.5,-0.5]$.

Zadatak 4.

4-periodična funkcija $f:\mathbb{R}\to\mathbb{R}$ ima svojstvo:

$$f(t)=\begin{cases} t+1, \text{ za } 0<t<2, \\ 2\cdot t-1, \text{ za } t\in[2,4]. \end{cases}$$

Funkciju *f* aproksimiramo Fourierovim polinomom F_5 stupnja 5 na segmentu $[0,4]$.

a) Izračunajte **zbroj** svih koeficijenata polinoma F_5 i zaokružite ga s točnošću od 10^{-5}.

b) Na **istoj** slici prikažite grafički funkciju *f* i polinom F_5 na segmentu $[0,4]$.

Zadatak 5.

$(2\cdot\pi)-$ periodična funkcija $g:\mathbb{R}\to\mathbb{R}$ ima svojstvo:

$$g(t)=2\cdot t^2, \text{ za } t\in[-\pi,\pi].$$

Funkciju *f* aproksimiramo Fourierovim polinomom F_7 stupnja 7 na segmentu $[-\pi,\pi]$.

a) Izračunajte **umnožak** svih koeficijenata polinoma F_7 i zaokružite ga s točnošću od 10^{-5}.

b) Na **istoj** slici prikažite grafički funkciju *g* i polinom F_7 na segmentu $[-\pi,\pi]$.

Zadatak 6.

$(2\cdot\pi)-$ periodična funkcija $h:\mathbb{R}\to\mathbb{R}$ ima svojstvo:

$$h(t)=-t^3, \text{ za } t\in\langle-\pi,\pi\rangle.$$

Funkciju *f* aproksimiramo Fourierovim polinomom F_9 stupnja 9 na segmentu $[-\pi,\pi]$.

a) Izračunajte **aritmetičku sredinu** svih koeficijenata polinoma F_9 i zaokružite je s točnošću od 10^{-5}.

b) Na **istoj** slici prikažite grafički funkciju *f* i polinom F_9 na segmentu $[-\pi,\pi]$.

Rezultati zadataka:

1. a) $-\ln(1 - x)$;

   b) $\ln 2 \approx 0.69315$;

   c) -1.

2. a) $M_7(t) = -1920 \cdot t^7 + 672 \cdot t^5 - 280 \cdot t^3 + 210 \cdot t$;

   b) $5.78424 \cdot 10^{-6}$ i $2.79074 \cdot 10^{-5}$ %;

   c) i d) Vidjeti sliku 1.

3. a) $T_5(x) = \frac{22}{729} \cdot (x+1)^5 - \frac{10}{243} \cdot (x+1)^4 + \frac{5}{81} \cdot (x+1)^3 - \frac{1}{9} \cdot (x+1)^2 + \frac{1}{3} \cdot (x+1) + 1$;

   b) $2.61752 \cdot 10^{-4}$; $2.28661 \cdot 10^{-2}$ %;

   c) i d) Vidjeti sliku 2.

4. a) -2.11139.

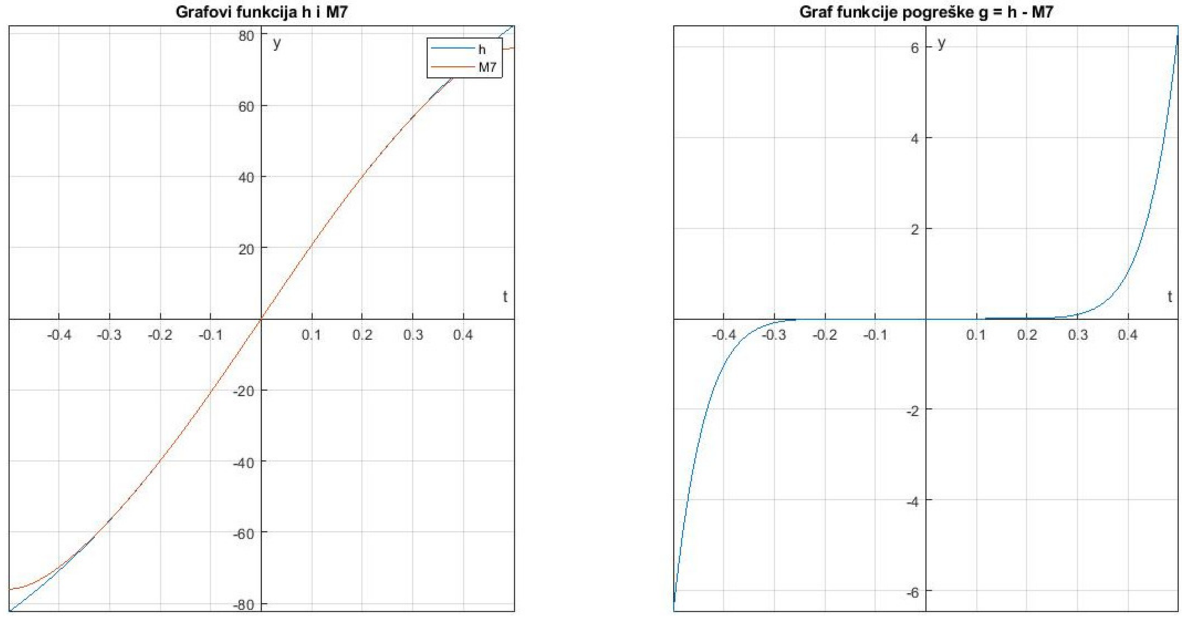
   b) Vidjeti sliku 3.

5. a) 0.54322.

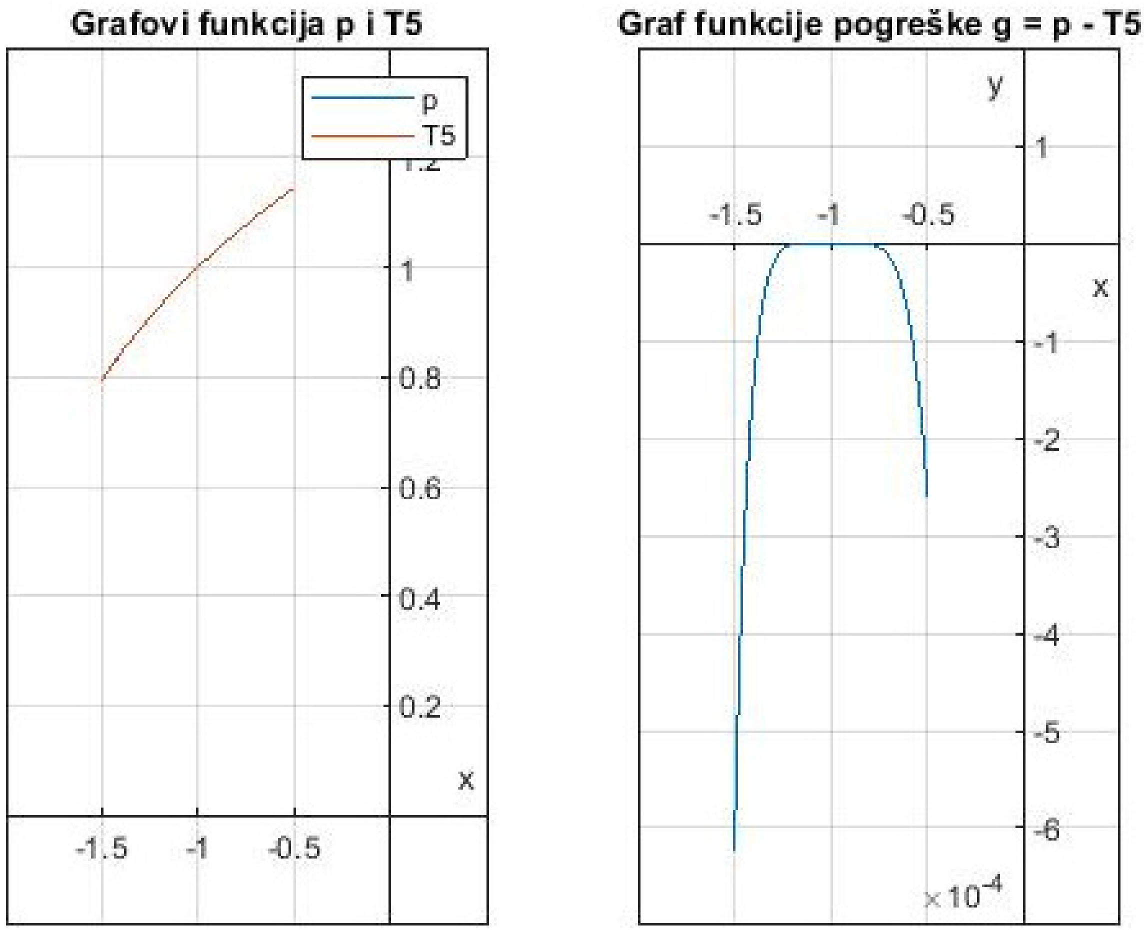
   b) Vidjeti sliku 4.

6. a) -0.43254.

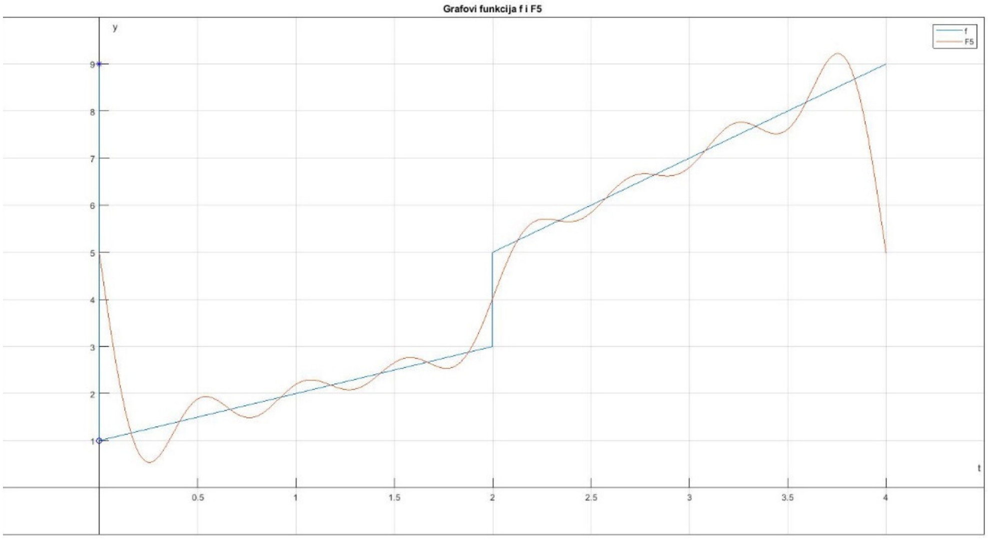
   b) Vidjeti sliku 5.



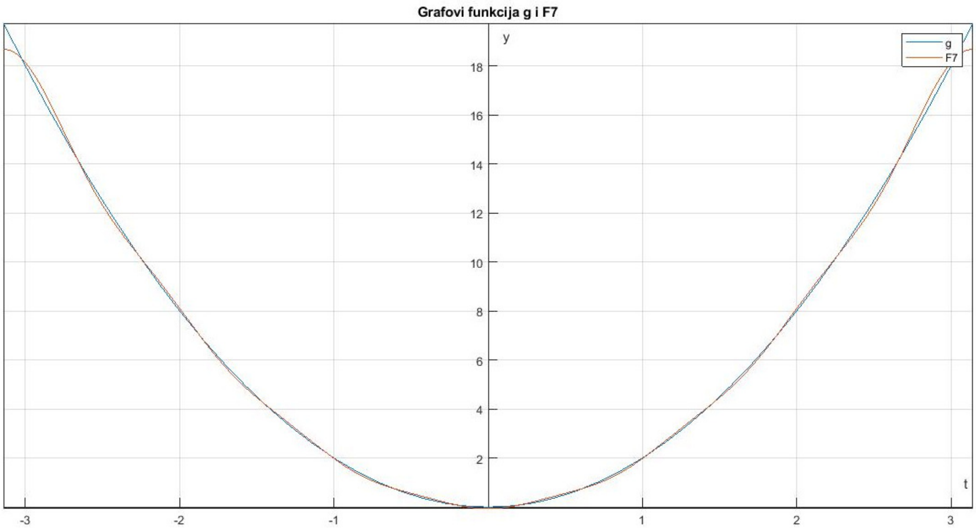
Slika 1.



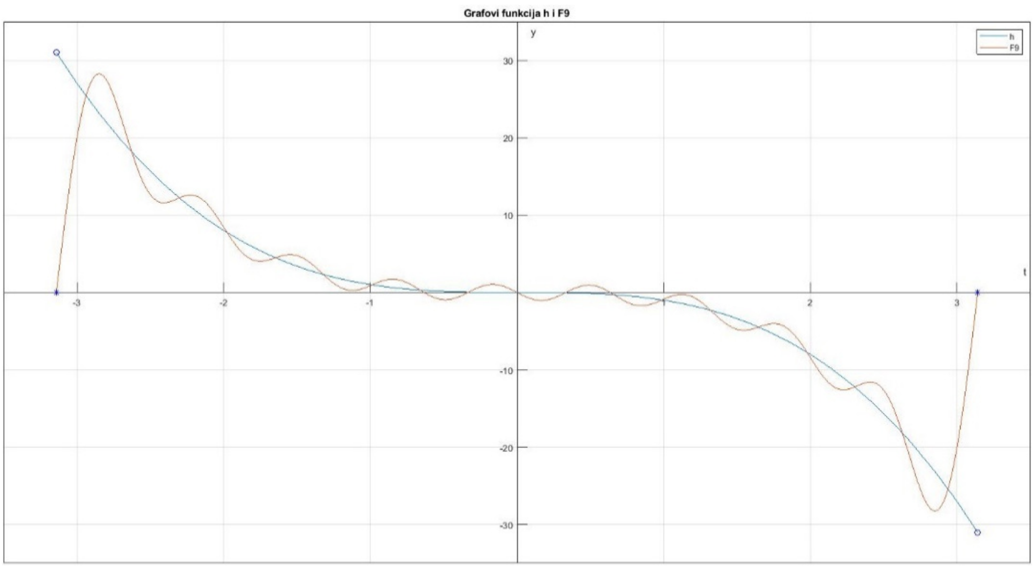
Slika 2.



Slika 3.



Slika 4.



Slika 5.

Komentari i objašnjenja programskih kodova

U ovoj ćemo vježbi na neki način „popuniti nedostatak" iz predmeta *Matematika* 2 vezan uz određivanje zbrojeva nekih funkcijskih redova potencija, a posebno uz grafičku analizu pogreške aproksimacije funkcije Taylorovim, odnosno MacLaurinovim polinomom određenoga stupnja. Analitička analiza tih pogrešaka je predmet proučavanja numeričke matematike. Ovdje ćemo se zadovoljiti grafičkim prikazom funkcije pogreške (na nekom intervalu oko točke oko koje vršimo aproksimaciju) iz koje ćemo moći procijeniti najmanju i najveću vrijednost te funkcije, a time i kvalitetu dobivene aproksimacije.

Prijeđimo na rješavanje zadataka.

z1.m

U prethodnoj smo vježbi rješavali zadatke vezane uz numeričke redove čiji su članovi bili „konkretni" realni brojevi. U ovoj vježbi bavit ćemo se funkcijskim redovima – preciznije, Taylorovim i Fourierovim redovima.

Prva tri zadatka odnose se na razvoje „konkretnih" funkcija u Taylorov red oko određene točke. To znači da je svaki član traženoga reda neka potencija nezavisne varijable. Svaka potencija ima svoj koeficijent, svoju bazu i svoj eksponent. Koeficijenti će biti realni (najčešće, racionalni) brojevi, baza potencije bit će varijabla funkcije, a eksponent će biti neki prirodan broj. Upravo zbog toga u prvoj liniji koda deklariramo dvije varijable: x i n. Prva varijabla označavat će bazu potencije, a druga eksponent.

U drugom retku koda zadajemo pravilo općega člana reda. Opći član reda možemo označiti po želji (ne smijemo koristiti oznake x i n jer smo ih već iskoristili za deklariranje simboličkih varijabli), pa smo odabrali oznaku a.

U trećem retku koda upoznajemo novu ugrađenu funkciju `assume` (engl.: pretpostaviti). Njezina je namjena omogućiti zadavanje *uvjeta* na vrijednost neke varijable. U ovom zadatku pretpostavljamo da je $x \in [-1,1\rangle$. Tu pretpostavku navodimo jer je upravo taj interval područje konvergencije zadanoga reda. (Dokažite tu tvrdnju koristeći Cauchyjev kriterij.) Budući da $x \in [-1,1\rangle$ znači da je x jednak ili veći od -1 **i** da je x strogo manji od 1, zadajemo dva logička uvjeta i povezujemo ih logičkim operatorom &. To je učinjeno u trećem retku koda. Tako smo omogućili MATLAB-u da, nakon izvršavanja trećega retka koda, prilikom izvršavanja svih preostalih redaka koda primjenjuje ovu pretpostavku.

U četvrtom retku koda određujemo zbroj zadanoga reda. Ovdje vidimo značaj navođenja varijable po kojoj se zbraja. Naime, „u igri“ su dvije simboličke varijable: x i n. Obje se pojavljuju u zapisu pravila općega člana zadanoga reda. Zbog toga je vrlo bitno navesti varijablu po kojoj se zbraja jer, barem teorijski, ona može biti ili x ili n. Mi zbrajamo po varijabli n, što znači da x shvaćamo kao konstantu, a vrijednosti varijable n mijenjamo. Početna vrijednost varijable n jednaka je 1, a završna `Inf`.

Nakon što odredimo zbroj reda, pojednostavljujemo dobiveni simbolički izraz što više možemo koristeći ugrađenu funkciju `simplify`. Potom u četvrtom retku koda ispisujemo dobiveni simbolički izraz.

U b) podzadatku treba izračunati egzaktnu i približnu vrijednost zbroja zadanoga reda koristeći rješenje a) podzadatka. Ma koliko god nam se ovaj zadatak činio kompliciranim, on je zapravo vrlo jednostavan. Treba pronaći „konkretan“ $x \in [-1,1\rangle$ koji uvršten u pravilo reda iz a) podzadatka daje red iz b) podzadatka. Lako vidimo da uvrštavanjem $x = \frac{1}{2}$ u red iz a) podzadatka dobijemo upravo red iz b) podzadatka. Zbog toga ćemo traženi zbroj reda u b) podzadatku dobiti tako da u izraz za zbroj reda dobiven u a) podzadatku uvrstimo $x = \frac{1}{2}$. Pritom se ne moramo opterećivati razmišljanjem o tome hoće li MATLAB znati da $\frac{1}{2}$ treba zamijeniti varijablu x. Zbroj reda iz a) podzadatka je ionako simbolička funkcija varijable x, pa se primjena ugrađene funkcije `subs` u šestom retku koda zapravo odnosi na ono što želimo: zamjenu simboličke varijable x u pravilu zbroja reda simboličkim brojem $\frac{1}{2}$. Tako ćemo dobiti egzaktnu vrijednost zbroja reda iz b) podzadatka.

Približnu vrijednost toga zbroja odredit ćemo primjenom ugrađene funkcije `double`. Egzaktnu i približnu vrijednost zbroja reda ispisujemo u sedmom retku koda. Uočite da u okviru ugrađene funkcije `fprintf` ponovno imamo ispis dvaju različitih objekata: simboličkoga broja z_0 i njegove približne vrijednosti (koja je realan broj zapisan u formatu dvostruke preciznosti).

Naposljetku, u c) podzadatku treba odrediti $x \in [-1,1\rangle$ za koji je zbroj reda iz a) podzadatka jednak –ln 2. To zapravo znači rješavanje odgovarajuće jednadžbe s jednom nepoznanicom (x). Rješavanje jednadžbi provodimo pomoću ugrađene funkcije `solve`. Primijetite da se za izjednačavanje lijeve i desne strane jednadžbe koristi dvostruka jednakost (==). Razlog za to je vrlo jednostavan: jednostruka jednakost koristi se prilikom dodjeljivanja vrijednosti određenoj varijabli. Budući da unaprijed ne znamo skup svih rješenja ove jednadžbe, a postoji „opasnost“ da barem jedan element toga skupa bude neki racionalan broj s relativno velikim brojnikom/nazivnikom, koristeći ugrađenu funkciju `double`, *sve* elemente skupa rješenja odmah „prebacujemo“ u format dvostruke preciznosti tako da ih, bude li to potrebno, možemo zaokružiti na određen broj decimalnih mjesta i dalje računati s njima. Dakle, iako nam se to na prvi pogled ne čini tako, varijabla `x1` je jednoretčana *matrica*, a ne realan broj, i ona predstavlja matrični zapis skupa svih rješenja zadane jednadžbe.

- *Pitanje za razmišljanje*: Na samom početku trećega retka koda stavite znak %. Potom izvršite tako dobiveni kod. Utvrdite što se dogodilo i objasnite svoje zaključke.

- *Pitanje za razmišljanje*: Može li se u osmom retku koda izbjeći korištenje znaka ==? Ako može, preinačite kod na odgovarajući način tako da ispisuje ispravne rezultate. U suprotnom, objasnite svoj odgovor.
  (*Uputa*: Ako se izostavi znak ==, MATLAB će pretpostaviti da treba naći sve kompleksne nultočke argumenta funkcije `solve`.)

z2.m

Kad pogledamo ovaj programski kod (ili bilo koji u svim preostalim zadacima ove vježbe), čini nam se „glomaznim“, pa *a priori* zaključujemo da je ovakav zadatak sigurno vrlo težak (jer „samo teški zadaci imaju dugačke programske kodove“, što je potpuna besmislica). Programski kod ima 30-ak linija jer je veći dio tih linija utrošen na „dotjerivanje“ dviju slika koje predstavljaju rezultate **c)** i **d)** podzadatka. Sam zadatak je zapravo prilično lagan, što ćete vidjeti u ostatku ovih komentara.

Cijeli zadatak traži određivanje MacLaurinova polinoma stupnja 7 kojim se aproksimira zadana funkcija, izračun apsolutne i relativne pogreške aproksimacije stvarne vrijednosti funkcije u nekoj točki vrijednošću MacLaurinova polinoma stupnja 7 u istoj točki, te grafičke prikaze funkcije, MacLaurinova polinoma i funkcije pogreške na konkretnom intervalu oko nule. Prije svega ponovimo: MacLaurinov polinom određenoga stupnja aproksimira zadanu funkciju u okolini (tj. na nekom intervalu) oko nule. Što se više udaljavamo od nule, aproksimacija postaje sve lošija. Dakle, aproksimaciju MacLaurinovim polinomom **ne smijemo** shvatiti kao zamjenu polazne funkcije tim polinomom u *bilo kojoj* točki njezine prirodne domene. Aproksimacija je isključivo lokalnoga karaktera i njezina kvaliteta ovisi o nizu drugih podataka (npr. o stupnju polinoma itd.) Mi se ovdje nećemo baviti analitičkom analizom aproksimacija, nego ćemo odgovarajuće zaključke izvesti uglavnom grafički.

Ponovimo i sljedeće: ako je vrijednost $f(x_0)$ aproksimirana vrijednošću $M_n(x_0)$, onda se apsolutna (oznaka: p_a) i relativna pogreška (oznaka: p_r) aproksimacije računaju pomoću izraza:

$$p_a = \left| f(x_0) - M_n(x_0) \right|,$$
$$p_r = \frac{p_a}{\left| f(x_0) \right|} \cdot 100 = \left| \frac{p_a}{f(x_0)} \right| \cdot 100 \ [\%].$$

Istaknimo da nam pritom nije bitan *predznak* aproksimacije, tj. ne znamo je li $M_n(x_0)$ veći ili manji od $f(x_0)$. Želimo li utvrditi i taj podatak, trebamo izostaviti apsolutnu vrijednost. Međutim, u analizi je puno važnija veličina pogreške (pri čemu predznak nije bitan), i to posebno relativne pogreške. Drugim riječima, relativna pogreška je daleko bolji pokazatelj kvalitete aproksimacije u odnosu na apsolutnu pogrešku. Npr. ako broj 1 aproksimiramo brojem 1.5, a broj 100 brojem 150, onda u prvom slučaju imamo apsolutnu pogrešku 0.5, a u drugom apsolutnu pogrešku 50, pa nam se čini da smo u drugom slučaju pogriješili sto puta više negoli u prvom. Međutim, taj zaključak je *netočan* jer su relativne pogreške u oba slučaja jednake 50%. Dakle, napravili smo jednako lošu aproksimaciju u obama slučajevima. Još je bolji primjer u kojemu broj 1 aproksimiramo brojem 1.5, a broj 100 brojem 120. Opet nam se čini da smo u drugom slučaju pogriješili 40 puta više nego u prvom, odnosno da je prva aproksimacija 40 puta bolja od druge. Taj zaključak je *potpuno pogrešan* jer su pripadne relativne pogreške jednake 50% i 20%, pa je druga aproksimacija bitno bolja od prve! Zbog toga u ovakvim slučajevima treba biti vrlo oprezan, te u analizi obavezno koristiti relativne pogreške kako bi se dobio što bolji kvantitativni pokazatelj mjere pogreške.

Prijeđimo na sam programski kod. Kao i obično, u prvom retku deklariramo simboličku varijablu, a u drugom retku definiramo pravilo funkcije h koju treba aproksimirati MacLaurinovim polinomom.

Podsjetimo da je MacLaurinov polinom poseban slučaj Taylorova polinoma za $c=0$. Drugim riječima, Taylorov polinom – uz određene uvjete - aproksimira funkciju na intervalu oko *bilo koje* točke iz njezine prirodne domene, a MacLaurinov polinom aproksimira funkciju isključivo na intervalu oko nule. U MATLAB-u se za dobivanje obaju polinoma koristi ugrađena funkcija `taylor`. Ona ima nekoliko ulaznih varijabli, pa ćemo ih sada precizno navesti.

Prva ulazna varijabla funkcije `taylor` je ili pravilo funkcije koju aproksimiramo ili oznaka te funkcije (ako je pravilo ranije zadano). U drugom retku koda smo zadali pravilo funkcije h pa ovdje navodimo samo njezinu oznaku.

Druga ulazna varijabla je oznaka varijable funkcije koju aproksimiramo. Funkcija h u ovom zadatku ima varijablu t. (Funkcija u svojemu pravilu može sadržavati i opće brojeve, pa je zato bitno istaknuti kojim je slovom označena njezina varijabla.)

Treća ulazna varijabla je točka (preciznije, realan broj, a ne npr. točka u ravnini ili prostoru) oko koje se vrši aproksimacija. Kod MacLaurinova polinoma ta je točka uvijek jednaka nuli, dok kod općenitoga Taylorova polinoma ona mora biti zadana u zadatku.

Četvrta ulazna varijabla je stalna i jednaka je `'Order'`. Kao rezervirana riječ u MATLAB-u, ona se **obavezno** piše pod jednostrukim navodnicima. Interpretiramo je kao *broj stupaca u matričnom zapisu tražena Taylorova polinoma*. Naime, otprije (vidjeti vježbu 7.) znamo da polinome možemo zapisati u obliku jednoretčane matrice. Ta matrica ima unaprijed zadan broj redaka (on je jednak 1), pa moramo zadati broj njezinih stupaca. Zbog toga upisujemo ovu riječ kao četvrtu ulaznu varijablu i **ne smijemo** je izostaviti (teorijski, smijemo, ali će onda MATLAB odrediti Taylorov polinom prema svojim početnim postavkama (*defaultu*), što je vrlo loše i treba obavezno izbjegavati (zašto bismo morali znati/učiti napamet MATLAB-ove početne postavke?)).

Posljednja, peta ulazna varijabla je ukupan broj stupaca jednoretčane matrice u kojoj će biti inicijalno pohranjen Taylorov polinom. Taj broj je, naravno, jednak ukupnom broju koeficijenata polinoma, a ovaj je za točno jedan veći od stupnja polinoma. Dakle, ako želimo dobiti Taylorov polinom stupnja 7, onda upisujemo 8 jer taj polinom ima ukupno 8 koeficijenata (od kojih su neki možda jednaki nuli – jedino za vodeći koeficijent Taylorova polinoma unaprijed znamo da je različit od nule). Spomenuti ukupan broj stupaca ponekad se naziva i *red polinoma* iako u matematici taj naziv treba izbjegavati jer sintagma „red polinoma“ znači funkcijski red čiji su svi članovi polinomi (analogno redu potencija).

Time smo u potpunosti opisali sve ulazne varijable ugrađene funkcije `taylor`. **Ne smijemo** promijeniti njihov redoslijed. Zbog toga se u trećem retku koda, kao ulazne varijable funkcije `taylor`, navode najprije h, potom t, pa 0, pa `'Order'`(pazite na jednostruke navodnike!) i naposljetku 8. MacLaurinov polinom je simbolička varijabla koja se iz jednoretčane matrice dobiva (već ugrađenom) primjenom funkcije `poly2sym`. Dakle, taj će polinom biti ispisan u uobičajenom („razvijenom“), a ne u matričnom obliku. Dakako, koristeći ugrađenu funkciju `sym2poly` uvijek možemo dobiti njegov matrični zapis.

Nakon što smo odredili MacLaurinov polinom, s njim dalje radimo kao i sa svakom drugom simboličkom funkcijom. U petom retku koda računamo vrijednost funkcije h u točki 0.1 i pretvaramo tu vrijednost u „običan“ realan broj zapisan u formatu dvostruke preciznosti. Analogno postupamo u šestom retku koda sa vrijednošću polinoma M_7 u istoj točki. „Pretvorba“ obiju vrijednosti u format dvostruke preciznosti napravljena je zbog potrebe računanja apsolutne, odnosno relativne pogreške aproksimacije. Bez te „pretvorbe“, apsolutna i relativna pogreška aproksimacije mogle bi biti racionalni brojevi s relativno velikim brojnikom/nazivnikom i, kao takve, potpuno beskorisne za daljnju analizu „kvalitete“ aproksimacije.

U sedmom i osmom retku koda, a prema ranije navedenim formulama, izračunane su apsolutna i relativna pogreška aproksimacije vrijednosti funkcije h vrijednošću polinoma M_7 u točki 0.1. One su ispisane u devetom retku koda.

Ostatak koda odnosi se na crtanje grafova. Grafove funkcija h i M_7 na zadanom segmentu treba prikazati na istoj slici, dok graf funkcije g treba prikazati na odvojenoj slici. Napomenimo da funkciju g nazivamo i *funkcija pogreške*, pa otuda i njezina oznaka. S obzirom da trebamo dobiti dvije slike istovremeno, moramo koristiti ugrađenu funkciju `subplot`. Na prvoj slici trebamo prikazati dvije krivulje na istom segmentu, pa je to najjednostavnije i najbrže učiniti koristeći ugrađene funkcije `fplot` i opciju `hold on` koja će omogućiti crtanje druge krivulje na istoj slici. Grafove trebamo prikazati ili kao elemente matrice tipa (2, 1) ili kao elemente matrice tipa (1, 2). Opredijelili smo se za drugu opciju zato da slike budu jedna pored druge. Naime, s obzirom na relativno „uzak“ segment na kojemu prikazujemo tražene tri krivulje, nema potrebe za „širinom“ slike, odnosno „razvlačenjem“ vrijednosti na osi apscisa.

Dakle, u bloku linija 10. – 24. najprije odredimo poziciju slike u matrici tipa (1, 2). Riječ je o slici koja će biti na 1. poziciji, pa je zbog toga treća ulazna varijabla funkcije `subplot` jednaka 1. Potom u 11. retku crtamo graf funkcije h na zadanom segmentu. U bloku od 12. – 15. retka uključujemo koordinatnu mrežu i postavljamo pravokutni koordinatni sustav u ravnini tako da se koordinatne osi sijeku u ishodištu. U 16. retku koda uključujemo opciju `hold on` kako bismo na istoj slici mogli prikazati graf MacLaurinova polinoma na istom segmentu. To činimo u 17. retku koda. Potom dolazi na red „dotjerivanje“ grafa: postavljanje odgovarajućega raspona vrijednosti na osi ordinata (iako se te vrijednosti mogu odrediti analitički, brže i jednostavnije ih je odrediti naknadno (nakon prvoga pokretanja koda), uvrstiti u kod i ponovno ga pokrenuti), postavljanje oznaka uz koordinatne osi i naslova grafa.

Naposljetku, u 23. retku koda određujemo drugi element matrice tipa (1, 2), a u bloku linija 24. – 32. na potpuno analogan način kao i prethodne dvije krivulje crtamo graf funkcije g (graf funkcije pogreške). Time je zadatak u potpunosti riješen.

Što možemo zaključiti promatrajući dobivene grafove? Vidimo da je na segmentu $[-0.2, 0.2]$ greška aproksimacije toliko mala da se graf funkcije g na tom segmentu prividno podudara s grafom konstantne funkcije $h_1(t) = 0$. Pritom posebno naglašavamo riječi *prividno podudara* jer funkcijska jednakost $h(t) = M_7(t)$ na navedenom segmentu nije istinita. Npr. vrijedi jednakost: $g(0.1) \approx 5.784 \cdot 10^{-6}$, a, s obzirom na preciznost prikaza točaka grafa, točka $\left(0.1,\ 5.784 \cdot 10^{-6}\right)$ praktički se podudara s točkom $(0.1, 0)$. Izbjegavanje ovakve „podudarnosti“ eventualno bi se moglo postići redefiniranjem jedinične dužine na osi ordinata, ali u tom slučaju nastaje problem s vrijednostima funkcije g na rubovima promatranoga segmenta koje tim redefiniranjem postaju „prevelike“ (pa bi se potpuno pogrešno činilo kao da krivulja „bježi“ u ∞). Zbog toga je ipak najbolje ostaviti mjerila na osi u sadašnjem obliku.

Iz grafa funkcije g najbolje se ocrtava sva „lokalnost“ kvalitete aproksimacije MacLaurinovim polinomom. Već za $x = \frac{1}{2}$ pogreška je približno jednaka 6.47, a daljnjim udaljavanjem od nule njezina apsolutna vrijednost postaje sve veća i vrlo brzo raste. Naime, očito su $\lim\limits_{t\to+\infty} h(t)=\dfrac{\pi}{2}$ i $\lim\limits_{t\to+\infty} M_7(t)=-\infty$. Zbog toga treba zapamtiti: aproksimaciju MacLaurinovim polinomom primjenjujemo samo za (relativno) male vrijednosti u okolini nule.

z3.m

Ovaj zadatak ne donosi ništa novoga u odnosu na prethodni, pa ćemo ga samo ukratko prokomentirati. On je zadani ponajprije radi aproksimiranja transcendentnih funkcija Taylorovim polinomom oko točke $c\neq 0$, a koji polinom je prilično sporo i mukotrpno odrediti analitički. Pritom je točka c namjerno odabrana tako da vrijednost zadane funkcije u toj točki bude cjelobrojna.

Iz izračunanih relativnih pogrešaka vidi se da su dobivene aproksimacije na zadanom segmentu relativno kvalitetne. Pritom treba pripaziti na „komprimiranje“ članova $\frac{1}{3}\cdot(x+1)+1$ u $\frac{1}{3}\cdot x+\frac{4}{3}$. To „komprimiranje“ se, nažalost, ne može izbjeći.

Također, iako je jasno da je npr.

$$(x-1)^2+(x-1)+1=x^2-x+1,\ \forall x\in\mathbb{R},,$$

treba imati na umu da je Taylorov polinom (konačan početni) dio Taylorova reda čiji je opći član $a_n=\dfrac{f^{(n)}(c)}{n!}\cdot(x-c)^n$, a ne „samostalni objekt“. Zbog toga u ispisu Taylorova polinoma uvijek treba koristiti originalni oblik članova iz Taylorova reda, tj. ne treba primjenjivati ugrađene funkcije `collect`, `expand` i dr.

U sljedećih nekoliko zadataka „kompenzirat“ ćemo još jedan nedostatak iz *Matematike* 2. U tom smo predmetu učili razvoj periodične funkcije u Fourierov red. Međutim, zadržali smo se samo na aproksimaciji funkcije Fourierovim polinomom i nismo je popratili odgovarajućim grafičkim prikazom koji bi nas uvjerio u „kvalitetu“ aproksimacije. Taj nedostatak ćemo ispraviti u ovom predmetu.

Odmah istaknimo da, za razliku od Taylorova reda, u MATLAB-u (još) ne postoji ugrađena funkcija za razvoj periodične funkcije u Fourierov red. U aplikaciji za prilagodbu ili „fitanje“ krivulje postoji opcija za interaktivnu aproksimaciju *konačnoga* skupa podataka Fourierovim polinomom određenoga stupnja. Međutim, u zadacima su nam zadane funkcije čiji se grafovi sastoje od beskonačno (točnije, neprebrojivo) mnogo točaka. Zbog toga ovu opciju nije moguće primijeniti.

U udžbeniku *Matematički alati u elektrotehnici* na str. 146 u okviru Primjera 1. navedena je funkcija **fr** čije su ulazne varijable funkcija f, realni brojevi a i b, te prirodan broj n. Funkcija ispisuje koeficijente Fourierova polinoma stupnja n koji aproksimira funkciju f na segmentu $[a, b]$. Preporučuje se detaljno proučiti taj primjer i razumjeti algoritam prema kojemu je napisana ta funkcija. Mi je na vježbama nećemo primjenjivati iz jednoga osnovnoga razloga: tu funkciju **ne smijete** koristiti na kolokviju i praktičnim ispitima (dozvoljeno je korištenje samo ugrađenih MATLAB-ovih funkcija), a pisati je samo zato da biste je mogli primijeniti u konkretnom zadatku zapravo je ekvivalentno rješavanju zadatka bez te funkcije. (Učiti pripadni kod napamet je također besmisleno i beskorisno.) Zbog toga ćemo zadatke koji slijede riješiti koristeći samo osnovne formule vezane uz Fourierove redove i ugrađene MATLAB-ove funkcije.

z4.m

Započinjemo s osnovnim zadatkom iz ovoga područja. *T*-periodičnu funkciju želimo na nekom segmentu aproksimirati Fourierovim polinomom. Podsjetimo se formula za opće Eulerove koeficijente u tom razvoju:

$$\left.\begin{aligned} a_0 &= \frac{1}{T}\cdot\int_a^{a+T} f(x)\cdot \mathrm{d}x,\\ a_k &= \frac{2}{T}\cdot\int_a^{a+T} f(x)\cdot\cos\left(\frac{2\cdot k\cdot\pi}{T}\cdot x\right)\cdot \mathrm{d}x, \text{ za svaki } k=1,\dots,n;\\ b_k &= \frac{2}{T}\cdot\int_a^{a+T} f(x)\cdot\sin\left(\frac{2\cdot k\cdot\pi}{T}\cdot x\right)\cdot \mathrm{d}x, \text{ za svaki } k=1,\dots,n. \end{aligned}\right\}$$

U tim je formulama T temeljni period funkcije *f*, a $[a, a+T]$ osnovni (temeljni) segment na kojemu je definirana funkcija *f* i s kojega se onda ta funkcija proširuje po periodičnosti na cijeli skup $\mathbb{R}$. Pogledajmo što konkretno imamo u našem zadatku.

Znamo da je $T = 4$. Osnovni segment je $[0,4]$, pa je $a = 0$. Funkcija *f* je zadana po dijelovima, što neće predstavljati osobiti problem. (Morat ćemo samo pripaziti na kojim su intervalima definirani pojedini dijelovi.) Želimo je aproksimirati Fourierovim polinomom stupnja 5, što znači da je $n = 5$. Kad to znamo, gornje formule prelaze u:

$$\left.\begin{aligned} a_0 &= \frac{1}{4}\cdot\int_0^{4} f(t)\cdot \mathrm{d}t,\\ a_k &= \frac{1}{2}\cdot\int_0^{4} f(t)\cdot\cos\left(\frac{k\cdot\pi}{2}\cdot t\right)\cdot \mathrm{d}t, \text{ za svaki } k=1,2,3,4,5;\\ b_k &= \frac{1}{2}\cdot\int_0^{4} f(t)\cdot\sin\left(\frac{k\cdot\pi}{2}\cdot t\right)\cdot \mathrm{d}t, \text{ za svaki } k=1,2,3,4,5. \end{aligned}\right\}$$

Prema tim formulama izračunat ćemo Eulerove koeficijente (*for*-petlja će nam izvrsno poslužiti upravo u tu svrhu), pa potom formirati Fourierov polinom F_5 i grafički ga prikazati zajedno sa funkcijom *f* na segmentu $[0,4]$.

Analizirajmo naš programski kod. U prvomu retku uobičajeno deklariramo simboličku varijablu, odnosno nezavisnu varijablu funkcije *f*.

U drugomu retku zadajemo funkciju *f* po dijelovima. Primijetimo da 0 pripada segmentu $[0,2]$, ali ne pripada ni intervalu $\langle 0,2\rangle$, ni segmentu $[2,4]$. Zbog toga $f(0)$

ne možemo izračunati tako da 0 uvrstimo u neku od dviju navedenih formula. Čini nam se da je zadatak pogrešno zadan, ali ipak nije tako. Previdjeli smo 4-periodičnost funkcije *f*. To znači da je

$$f(0) = f(0+4) = f(4),$$

a posljednju vrijednost možemo izračunati koristeći drugu formulu jer $4 \in [2, 4]$. Dakle,

$$f(0) = f(4) = 2 \cdot 4 + 1 = 9,$$

pa zbog toga u drugomu retku koda moramo dodatno navesti tu jednakost. Uočite da funkcija `piecewise` kao svoje ulazne varijable ima i jednakosti i nejednakosti, što je potpuno korektno. Pritom moramo paziti na poredak ulaznih varijabli: *uvijek* najprije pišemo logički uvjet na nezavisnu varijablu, a tek potom pravilo koje primjenjujemo ako je taj logički uvjet ispunjen. Logičke uvjete i pravila međusobno odvajamo „običnim“ zarezima.

Bolji poznavatelji matematičke teorije odmah će reći da dodavanje jedne točke (u ovom slučaju $(0, f(0))$) ne utječe na određivanje integrala. To je točno. Međutim, korektno je zadati funkciju *f* na cijelom segmentu $[0, 4]$, a ne na intervalu $\langle 0, 4]$. Uostalom, u b) podzadatku funkciju *f* ionako trebamo grafički prikazati na segmentu $[0, 4]$, pa je tim primjerenije zadati vrijednost te funkcije u nuli.

Krenimo dalje. U trećem retku koda računamo slobodni član Fourierova polinoma, tj. a_0, prema gore navedenoj formuli. U tom izrazu nema ništa posebnoga, riječ je o vrlo jednostavnoj primjeni funkcije `int` na računanje određenoga integrala. Jedino treba pripaziti da se ispred integrala ne zaboravi upisati $\frac{1}{4}$.

U retcima 4. – 7. računamo preostalih 14 Eulerovih koeficijenata (15. koeficijent a_0 smo već izračunali) koristeći gornje formule. Primijetite da koeficijente uz kosinuse višestrukih lukova pohranjujemo u jednoretčanu matricu *a*, dok koeficijente uz sinuse višestrukih lukova pohranjujemo u jednoretčanu matricu *b*. Elementi *obiju* tih matrica su *simbolički* brojevi jer funkcija `int` kao rezultat vraća simbolički objekt. *For*-petlja će se izvršiti ukupno 7 puta i u svakom koraku popunit će svaku od matrica *a* i *b* s točno jednim novim elementom.

U 8. retku koda računamo zbroj svih izračunatih Eulerovih koeficijenata, te ga ispisujemo u 9. retku koda. Primijetite da je drugi pribrojnik izračunat kao zbroj svih elemenata jednoretčane matrice $a+b$. To je moguće jer se jednoretčane matrice zbrajaju tako da se zbroje elementi na istim pozicijama, pa je – zbog komutativnosti i asocijativnosti zbrajanja u skupu realnih brojeva – zbroj svih elemenata matrica a i b jednak zbroju svih elemenata matrice $a+b$.

U retcima 10. – 13. stvaramo Fourierov polinom također koristeći *for*-petlju. Uočite da u retku 12. zbrajamo *simboličke objekte*. To zbrajanje je samo formalno jer pri svakom prolazu kroz *for*-petlju nadopisujemo dva nova člana Fourierova polinoma koji su istoga stupnja (tj. članove $a_k \cdot \cos\left(k \cdot \frac{\pi}{2} \cdot t\right)$ i $b_k \cdot \sin\left(k \cdot \frac{\pi}{2} \cdot t\right)$). S obzirom na originalni oblik Fourierova polinoma, *ne* odgovara nam da MATLAB, koristeći funkciju `simplify`, nastoji pojednostavniti dobiveni izraz. Zbog toga ne koristimo tu funkciju. Nakon 7 izvršavanja *for*-petlje dobit ćemo polinom F_5 kao simbolički objekt.

U ostatku koda crtamo funkcije f i F_5 na zadanom segmentu. U tom dijelu koda vrijedi istaknuti samo da, koristeći funkciju `plot`, zasebno ucrtavamo dvije „sporne" točke: (0,9) i (0,1). Prva od njih pripada grafu funkcije *f*, pa je ucrtavamo koristeći znak *, a druga *ne* pripada tom grafu (iako se iz slike čini da pripada), pa je ucrtavamo koristeći znak o (prazan kružić). Graf uobičajeno „dorađujemo" ucrtavanjem koordinatne mreže, te organiziranjem pravokutnoga koordinatnoga sustava u ravnini tako da se koordinatne osi sijeku u ishodištu. Uz graf uobičajeno navodimo i legendu kako bismo kasnije znali koja krivulja se odnosi na koju funkciju (iako u ovom slučaju to vrlo lako možemo zaključiti i bez legende).

- *Pitanje za razmišljanje*: Možemo li i slobodni član uvrstiti u matricu *a* tako da ta matrica sadrži sve „*a*-ove", a da pritom formula za a_k ostane nepromijenjena? Ako možemo, preinačite programski kod na odgovarajući način. Precizno obrazložite svoje tvrdnje.

z5.m

Ovaj zadatak se naizgled ni po čemu bitnom ne razlikuje od prethodnoga. Štoviše, čini nam se lakšim jer funkcija *g* koju aproksimiramo Fourierovim polinomom nije zadana po dijelovima, nego „u komadu“ na segmentu $[-\pi, \pi]$ čija duljina odgovara temeljnom periodu funkcije *g*. Zbog čega onda uopće vrijedi zasebno izdvojiti takav zadatak? Zašto jednostavno ne kopiramo rješenje prethodnoga zadatka, promijenimo odgovarajuće podatke (pravilo funkcije *g*, temeljni period i segment) i time završimo priču?

Razlog za zasebno rješavanje ovoga zadatka je matematički, a ne informatički. Naime, uočimo da je funkcija *g* *parna* funkcija. Podsjetimo se, funkcija $g : A \to \mathbb{R}$ je *parna* ako vrijedi:

$$1.)\ t \in A \Leftrightarrow -t \in A;$$
$$2.)\ g(-t) = g(t),\ \forall t \in A.$$

Slobodno rečeno, *g* je parna ako je *A* interval simetričan s obzirom na nulu (pri čemu ne mora nužno vrijediti relacija $0 \in A$) i ako se svaki par međusobno suprotnih brojeva preslika u isti realan broj. (U predmetu *Matematika* 1 taj uvjet smo zvali „sapuničastim uvjetom“: „dva brata blizanca, od kojih je jedan jako dobar, a drugi jako zao, zaljube se u istu djevojku“.) Tipični primjeri parne funkcije su funkcija iz zadatka, konstantna funkcija i trigonometrijska funkcija oblika $\cos(a \cdot t)$. Što je sa funkcijom oblika $\sin(a \cdot t)$? Ona nije parna, nego *neparna*, ali o tome ćemo nešto više reći u sljedećem zadatku.

I dalje nam nije jasno zašto bi ovo matematičko svojstvo funkcije *g* bilo bitno za rješavanje zadatka. Evo odgovora i na to pitanje. Ako je funkcija parna, onda njezin razvoj u Fourierov red (pa posebno i Fourierov polinom kao konačan početni „komad“ toga reda) sadrži slobodni član i članove oblika $a_k \cdot \cos\left(\frac{2 \cdot k \cdot \pi}{T} \cdot t\right)$. Dakle, efektivno trebamo računati slobodni član i elemente matrice *a*, dok elemente matrice *b* nije potrebno računati. Ovdje se „sakrilo“ još jedno matematičko svojstvo: zbroj i razlika dviju parnih funkcija ponovno su parne funkcije, dok zbroj i razlika parne i neparne funkcije nije ni parna, ni neparna funkcija. Kao aproksimacija zadane parne funkcije, i Fourierov polinom mora imati to isto svojstvo, tj. i on mora biti parna funkcija, pa zato ne može sadržavati sinuse višestrukih kutova.

U *Matematici* 2 smo pokazali da se, u slučaju $(2\cdot\pi)-$ periodične parne funkcije čiji je temeljni segment $[-\pi,\pi]$, gore spomenute formule mogu preinačiti u bitno jednostavnije (detalje ovdje izostavljamo):

$$\left.\begin{aligned}
&a_0=\frac{1}{\pi}\cdot\int_0^{\pi} g(t)\cdot\mathrm{d}t,\\
&a_k=\frac{2}{\pi}\cdot\int_0^{\pi} g(t)\cdot\cos(k\cdot t)\cdot\mathrm{d}t,\ \text{za svaki } k=1,\ldots,n;\\
&b_k=0,\ \text{za svaki } k=1,\ldots,n.
\end{aligned}\right\}$$

Sad je jasno da će programski kod biti nešto kraći u odnosu na onaj iz prethodnoga zadatka. Pogledajmo njegovu strukturu.

U prvom retku zadajemo simboličku varijablu.

U drugom retku zadajemo pravilo funkcije *g*.

U trećem retku zadajemo preinačeni izraz za računanje slobodnoga člana a_0.

U retcima 4. – 6., koristeći *for*-petlju, zadajemo preinačeni izraz za računanje Eulerovih koeficijenata a_k.

U 7. retku računamo umnožak svih Eulerovih koeficijenata kao umnožak slobodnoga člana i umnoška svih elemenata matrice *a*. To smijemo učiniti zbog asocijativnosti množenja realnih brojeva. U 8. retku ispisujemo taj umnožak.

Sam polinom F_7 dobijemo analogno kao u rješenju prethodnoga zadatka. Ponovno koristimo *for*-petlju i u svakom od sedam njezinih koraka polinom „nadograđujemo“ s novim članom (oblika $a_k\cdot\cos(k\cdot t)$). Potom slijedi crtanje grafova funkcija *g* i F_7 na segmentu $[-\pi,\pi]$. Ovdje nema nikakvih trikova, ni dodavanja izoliranih točaka: obje funkcije su u potpunosti definirane na zadanom segmentu, pa ih i crtamo kao takve. Uobičajeno postavljamo i koordinatnu mrežu, kao i pravokutni koordinatni sustav u ravnini takav da se koordinatne osi sijeku u ishodištu. Dodajemo oznake uz koordinatne osi i legendu jer ćemo na istoj slici dobiti dva grafa. I to je sve u ovom zadatku.

Nakon izvršenja programskoga koda, preostaje primijetiti da u ovom slučaju Fourierov polinom vrlo solidno aproksimira funkciju *g*. Kvalitetu te aproksimacije uobičajeno možemo provjeriti i crtanjem odgovarajuće funkcije greške. Učinite to za vježbu.

- *Pitanje za razmišljanje*: Možemo li i u ovom zadatku sve Eulerove koeficijente svrstati u istu matricu tako da formula za a_k ostane nepromijenjena? Ako možemo, preinačite programski kod na odgovarajući način. Precizno objasnite sve svoje tvrdnje.

z6.m

Za ovaj zadatak bismo slobodno mogli reći da je „šećer na kraju" ove vježbe. U odnosu na prethodna dva zadatka, u nekim svojim dijelovima on je jednostavniji, ali zato sadrži nekoliko „zamki" na koje ćemo posebno ukazati.

Osnovna ideja za rješavanje ovoga tipa zadatka je uočiti da je funkcija *h* neparna. Definicija neparne funkcije također sadrži dva uvjeta: prvi je jednak prvom uvjetu iz definicije parne funkcije, dok je drugi

$$h(-t) = -h(t), \ \forall t \in A.$$

Ovaj uvjet čitamo kao: „međusobno suprotni *t*-ovi preslikavaju se u međusobno suprotne brojeve". Tipični primjeri neparnih funkcija su funkcije oblika $a \cdot t^n$, gdje je *n* *neparan* prirodan broj, funkcije oblika $\sin(a \cdot t)$ i druge.

Potpuno analogno kao u komentarima rješenja prethodnoga zadatka zaključujemo da Fourierov razvoj neparne funkcije u red ne može sadržavati slobodni član, ni članove oblika $a_k \cdot \cos\left(\frac{2 \cdot k \cdot \pi}{T} \cdot t\right)$. Dakle, u obzir dolaze samo sinusi višestrukih kutova, tim više što su zbroj i razlika dviju neparnih funkcija ponovno neparne funkcije.

Funkcija iz našega zadatka je $(2 \cdot \pi)-$ periodična funkcija definirana na intervalu $\langle -\pi, \pi \rangle$. Kod pažljivijih rješavača zadatka interval $\langle -\pi, \pi \rangle$ će nametnuti pitanje: „A zašto nije zadan segment $[-\pi, \pi]$?" Netko će zaključiti da se valjda radi o pogreški u zadatku, a oni oprezniji „nanjušit će" neku podvalu. Ovi posljednji bit će potpuno u pravu. Nije riječ o pogreški u zadatku, interval $\langle -\pi, \pi \rangle$ je ispravno zadan, pa ne znamo vrijednost funkcije *h* ni u jednoj od rubnih točaka segmenta $[-\pi, \pi]$. Tako smo efektno upali u prvu zamku: te vrijednosti *ne računamo prema formuli* navedenoj u zadatku, ali nije točno da ih uopće ne znamo. U *Matematici* 2 smo dokazali sljedeću tvrdnju:

Tvrdnja 1. Neka je $h: \mathbb{R} \to \mathbb{R}$ neparna $(2 \cdot \pi)-$periodična funkcija. Tada je $h(k \cdot \pi) = 0, \ \forall k \in \mathbb{Z}$.

Dakle, znamo vrijednosti funkcije h u rubnim točkama segmenta $[-\pi,\pi]$ i one *ne ovise* o formuli kojom je definirana funkcija h. Zbog toga ćemo točke $(-\pi,0)$ i $(\pi,0)$ morati zasebno ucrtati na graf funkcije h. Uočite da je $h(0)=0$ i prema formuli iz zadatka i prema tvrdnji 1., pa točku $(0,0)$ ne moramo zasebno ucrtavati na graf funkcije h.

Izvukli smo se iz zamke, pa nastavljamo rješavati prvi podzadatak. Ako je h $(2\cdot\pi)-$periodična neparna funkcija, onda postoje preinačene formule za Eulerove koeficijente. One glase:

$$\left.\begin{aligned} &a_0=a_k=0, \text{ za svaki } k=1,\ldots,n;\\ &b_k=\frac{2}{\pi}\cdot\int_0^{\pi} h(t)\cdot\sin(k\cdot t)\cdot\mathrm{d}t, \text{ za svaki } k=1,\ldots,n.\end{aligned}\right\}$$

U skladu s njima, ne zadajemo ni slobodni član a_0, ni matricu *a*, nego samo matricu *b*. Pogledajmo kako to učiniti.

Početak koda je standardan: najprije zadajemo simboličku varijablu koja predstavlja nezavisnu varijablu funkcije *h*, a potom pravilo funkcije *h*.

U retcima 3. – 5. kroz *for*-petlju definiramo svih devet elemenata jednoretčane matrice *b*, a prema gornjim formulama. (U ovome je podzadatku očito $n=9$.) Potom u 4. retku računamo aritmetičku sredinu svih elemenata matrice *b*, pa u 5. retku ispisujemo izračunanu vrijednost. Time je u potpunosti riješen **a)** podzadatak.

Prelazimo na rješavanje **b)** podzadatka. U prethodnim dvama podzadacima već smo vidjeli način „konstrukcije“ Fourierova polinoma, pa odgovarajući analogon toga načina primijenimo i u ovom podzadatku. Koristeći *for*-petlju u svakom od devet koraka dodajemo novi član oblika $b_k\cdot\sin(k\cdot t)$. Budući da rješenje ne sadrži slobodni član a_0, kao inicijalno pravilo polinoma F_9 stavljamo nulu. (Nulfunkcija je jedina funkcija koja je istovremeno i parna i neparna, a ovdje je dodatno korisna jer je nula tzv. *neutralni element* za zbrajanje realnih brojeva, tj. vrijedi jednakost: $0+x=x,\ \forall x\in\mathbb{R}.$) Tako u recima 8. – 11. „konstruiramo“ polinom F_9 kojega ćemo u potpunosti „konstruirati“ nakon završnoga, devetoga izvršavanja *for*-petlje.

Imajući u vidu prethodne analize i napomene, ostatak koda ne donosi nikakve novosti. Najprije crtamo graf funkcije *h* na segmentu $[-\pi,\pi]$, pri čemu uobičajeno prikazujemo i koordinatnu mrežu, a koordinatni sustav organiziramo tako da se koordinatne osi sijeku u ishodištu. Potom, „zamrzavajući“ sliku korištenjem opcije `hold on`, crtamo graf polinoma F_9 pomoću funkcije `fplot`, te točke $(-\pi, 0)$ i $(\pi, 0)$ koristeći funkciju `plot`.

Da bismo jasnije istaknuli koje točke pripadaju grafu funkcije h, a koje ne, izolirane točke crtamo koristeći opciju `'*b'` funkcije `plot`, pa ćemo te točke na grafu vidjeti kao plave zvjezdice. Točke $\left(\pi, -\pi^3\right)$ i $\left(-\pi, \pi^3\right)$ *ne* pripadaju grafu funkcije h, pa ih crtamo koristeći opciju `'ob'` funkcije `plot`. Tako ćemo te dvije točke na grafu vidjeti kao dva plava prazna kružića. Naposljetku dodajemo oznake uz koordinatne osi, te legendu radi lakšega snalaženja na dobivenom grafu.

Primjećujemo da je aproksimacija nešto lošija nego u prethodnom zadatku. Za bolju aproksimaciju trebali bismo uzeti polinom većega stupnja. Učinite to sami za vježbu.