TEORIJA – Numerički algoritmi i numerički softver

1. Prezentacija – Uvod u NANS
    1.1. Šta su numerički algoritmi?

    Numerički algoritam je skup tehnika pomoću kojih se matematički problemi formulišu kao problemi koji se mogu rešiti aritmetičkim i logičkim operacijama.

    1.2. Koji su problemi u računarstvu koji se mogu rješiti numeričkim algoritmima?

    Problemi koji se mogu rješiti numeričkim algoritmima su:

    a) **Fizika u kompjuterskim igrama – kretanje, uticaj sile, detekcija kolizije objekata, simualcija tkanine, simulacija tečnosti, kretanje čestica**
    b) **Prediktivno modelovanje – predikcija ishoda sportskih utakmica, predikcija cijena akcija, predikcija valute**
    c) **Razni drugi primjeri – rangiranje na društvenim mrežama, GPS, grafika, generisanje terena**

2. Prezentacija – Sistemi linearnih algebarskih jednačina (Gausova eliminacija)
    2.1. Koja su tri ishoda rješavanja sistema lin. algebarskih jednačina? Šta znači kada je sistem loše uslovljen?

    Tri moguća ishoda su:

    a) **Sistem lin. algebarskih jednačina ima jedinstveno određeno rješenje**
    b) **Sistem lin. algebarskih jednačina nema jedinstveno određeno rješenje, sistem ima beskonačno mnogo rješenja**
    c) **Sistem lin. algebarskih jednačina nema rješenje**

    Loše uslovljen sistem je onaj sistem lin. algebarskih jednačina kod kojeg mala promjena u početnim uslovima rezultuje velikom promjenom u rješenju. Takođe sistem je loše uslovljen ako je determinanta sistema bliska nuli.

    2.2. Koja su tri načina za rješavanje SLAJ?

    Tri načina za rešavanje:

    • Grafički – rješenje se dobija crtanjem jednačina u koor. sistemu, rješenje se ako je jedinstveno dobija u presjeku jednačina, u suprotnom sistem ili nema rješenje (prave paralelne) ili ima beskonačno mnogo rješenja(prave leže jedna na drugoj)

    • Kramerovo pravilo - računaju se determinanta sistema D, determinante Di,

    i = 1,...,n (n je broj promjenljivih). Rješenje sistema se dobija:

x_i = Di / D -> Di – je determinanta sistema kada se umjesto i-te kolone koja označava kolonu i-te promjenljive zamjeni vektor kolone slobodnih koeficijenata.

Komentar: Ovde se vidi posljedica loše uslovljenog sistema (D ≈ 0, djeljenje nulom(ne bukvalno 0) – ima više rješenja, velike greške u zaokruživanju, greška se samo propagira dalje u algoritmu.

- **Eliminacijom - Gausova eliminacija**

2.3. Objasniti metodu Gausove eliminacije!

Gausov metod eliminacije je direktan metod kod koga nema iteracija, trenutnih i prethodnih rješenja.

Koraci:

a) **Eliminacija unaprijed – Kolonu po kolonu elimišemo elemente ispod glavne dijagonale, rezultat toga je gornja trougaona matrica.**
b) **Zamjena unazad – Izračunamo x_n direktno, zamjenom unazad tako što koristimo izračunate x_i do x_n da bismo izračunali x_{i-1} do x_1.**

Eliminacija x_1, množimo prvu jednačinu sa $(-a_{21}/a_{11})$ i sabiramo je sa drugom. Ponavljati algoritam dok se x_1 ne poništi u svakoj jednačini osim prve. Jednačina koja se množi u algoritmu (u ovom slučaju prva) se naziva pivot jednačina, a koeficijent uz promjenljivu koja se eliminiše u pivot jednačini(u ovom slučaju a_{11}) se zove pivot element.

Algoritam se nastavlja tako što druga jednačina postane pivot jednačina, i sve tako do n-te jednačine (n broj jednačina).

Zamjena unazad – kreće od x_n koji se izračuna direktno, pa onda redom $x_{n-1},\ldots,x_1$.

$$x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}}$$

$$x_i = \frac{b_i^{(i-1)} - \sum_{j=i+1}^{n} a_{ij}^{(i-1)} x_j}{a_{ii}^{(i-1)}}$$

Eliminacija unapred izvršava O(n3/3) operacija .

Zamena unazad izvršava O(n2/2).

2.4. Koji su osnovni problemi kod Gausove eliminacije?

Osnovni problemi kod Gausove eliminacije:

a) **Djeljenje nulom**
b) **Djeljenje malim brojevima(dešavaju se greške u zaokruživanju)**
c) **Loše uslovljeni sistemi (sistemi kod kojih mala promjena početnih koeficijenata sistema rezultuje velikom promjenom rješenja sistema)**

2.5. Objasni postupak PP Gausove eliminacije (PP – parcijalni pivoting)!

Postupak parcijalnog pivotinga je postupak koji se koristi u Gausovoj eliminaciji kod pravljenja trougaonog oblika matrice sa nulama (postupak eliminacije).
Kada radimo eliminaciju unaprijed biramo uvijek pivot koji ima najveću apsolutnu vrijednost u toj koloni. Zamjenimo trenutnu jednačinu sa tom kod koje smo pronašli najveći element. PP rješava problem djeljenja sa nulom.
Ako tražimo taj element u cijeloj matrici, onda mjenjamo i red i kolonu i to se naziva kompletan pivoting (algoritamski zahtjevno – ne koristi se u praksi).

3. Prezentacija – Sistemi linearnih jednačina ITERATIVNE METODE

3.1. Zašto se koriste iterativne metode kod rješavanja SLAJ?

U praksi najčešće radimo sa sistemima koji imaju veliki broj jednačina. Za takve sisteme direktne metode su previše računski zahtevne. Vrlo često je bolje, da brzo dobijemo približno rešenje nego da dugo čekamo na tačno rešenje. Iz tih razloga koristimo iterativne metode.

3.2. Koje su osnove bilo kojeg iterativnog metoda?

Kod iteratirvnih metoda krećemo od odabrane početne vrijednosti x^0 (bira je korisnik metode), potom koristimo itrativnu formulu koja daje vezu između x^k i x^{k-1} i na taj način izračunavamo $x^0, x^1,...,x^{k-1}, x^k$
Kao kriterijum zaustavljanja koristimo apsolutnu približnu grešku između k-te i (k-1) iteracije.
Sistem $Ax = b$ se transformiše u oblik $x = Tx + c$, odnosno u iterativnu formulu $x^k = Tx^{k-1} + c$. Matrica T i vektor c se određuju pomoću A i b.

3.3. U zavisnosti od određivanja matrice T i vektora c kod iterativnih metoda koje iterativne metode za rješavanje SLAJ-a razlikujemo?

Jakobijev metod (Jacobi) –

A=L+D+U (*nema veze sa LU faktorizacjiom*)

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ a_{21} & 0 & 0 \\ a_{31} & a_{32} & 0 \end{bmatrix} + \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{bmatrix} + \begin{bmatrix} 0 & a_{12} & a_{13} \\ 0 & 0 & a_{23} \\ 0 & 0 & 0 \end{bmatrix}$$

$Ax=b \Rightarrow (L+D+U)x=b$ $\quad Dx^{k+1} = -(L+U)x^k+b$

$$x_i^{k+1} = \frac{1}{a_{ii}}\left[b_i - \underbrace{\sum_{j=1}^{i-1} a_{ij}x_j^k}_{Lx^k} - \underbrace{\sum_{j=i+1}^{n} a_{ij}x_j^k}_{Ux^k} \right]$$

(Dx^{k+1})

$x^{k+1}=-D^{-1}(L+U)x^k+D^{-1}b$

$T=-D^{-1}(L+U)$

$c=D^{-1}b$

Gaus-Zajedelov (Gauss-Seidel) –

$$x_i^{k+1} = \frac{1}{a_{ii}}\left[b_i - \underbrace{\sum_{j=1}^{i-1} a_{ij}x_j^{k+1}}_{Lx^{k+1}} - \underbrace{\sum_{j=i+1}^{n} a_{ij}x_j^k}_{Ux^k} \right] \qquad (D+L)x^{k+1} = -Ux^k+b$$

(Dx^{k+1})

$x^{k+1}=-(D+L)^{-1}Ux^k+(D+L)^{-1}b$

$T=-(D+L)^{-1}U$

$c=-(D+L)^{-1}b$

3.4. Objasniti konvergenciju iterativnih metoda!

Greška u (k+1) iteraciji se računa po formuli $e^{k+1} = T^{(k+1)}e^0$, gdje kod e na stepen k+1, k označava k-tu iteraciju, a kod T, k označava k-ti stepen.

Ako $\lim_{k\to\infty} \left|\left|T^{k+1}\right|\right| \to 0$ iterativni metod konvergira. (Međutim ovaj kriterijum konvergencije izričito zavisi od izbora norme za T).

Spektralni radijus - Spektralni radijus ρ matrice T je apsolutna vrednost njene maksimalne karakteristične (sopstvene) vrednosti ($q(T) = \max_{0\le i\le n} |\theta_i|$).

Ako spektralni radijus predstavlja broj manji od 1, onda i norma T-a je manja od 1 (metod konvergira).

Jacobi:

Ako je matrica A dijagonalno dominantna, Jakobijev metod konvergira za bilo koje početno rešenje. Primetite da se ovaj uslov proverava za A, a ne T.

Gauss – Seidel:

Metod konvergira za bilo koje početno rešenje ako je matrica A dijagonalno dominantna.

3.5. Objasniti efikasnost iterativnih metoda rjesavanja SLAJ-a!

Ako je broj iteracija potrebnih za konvergenciju mnogo manji od n, tada su iterativne metode efikasnije od direktnih.

Iterativne metode su takoĎe efikasne kad je A matrica koja sadrži puno nula elemenata (što je čest slučaj u praksi – npr. PageRank, sistemi za preporuku itd).

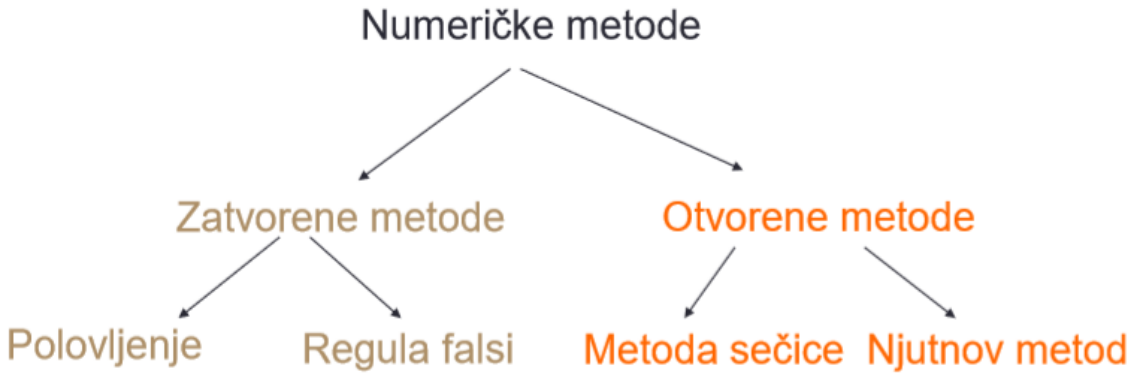
4. Prezentacija – Rješavanje nelinearnih jednačina ITERATIVNE METODE

„Rješenje nelinearne jednačine f(x) = x“ ≡ „nula funkcije f(x) – x = 0“.

Podsjećanje – iterativne metode koriste početno pogađanje i iterativnom formulom koja daje vezu između k-tog i (k-1)-og rješenja, do neke tolerancije daje približno rješenje.

4.1. Koji načini za rješavanje nelinearnih jednačina postoje?

a) **Analitički metod – moguće samo za određene vrste nel. jednačina**
b) **Grafički metod – korisno za procjenu lokacije rješenja, ograničeni smo brojem dimenzija**
c) **Numeričke metode:**



Zatvorene metode – algoritam kreće od zatvorenog intervala koji sadrži rješenje, te svakim sljedećim korakom taj se interval smanjuje i svodi se na jednu tačku, odnosno rješenje ili dok veličina intervala ne padne ispod zadate tolerancije (kriterijum koji se koristi u praksi

– $|x^k – x^{k-1}| < tolerance$).

Otvorene metode – algoritam kreće od početnog rešenja x0, te se svakim korakom dobija nova procjena rješenja. Algoritam se zaustavlja ili kada pronađemo tačku x

(f(x) = 0), ili $\mathbf{|x^k - x^{k-1}| < tolerance}$.

4.2. Objasniti metodu polovljenja (zeroBisectionMethod)!

Jedna od najjednostavnijh metoda za rešavanje nelinearnih jednačina. Kao uslov za korišćenje zahteva **zatvoreni interval** za koji je poznato da sadrži rešenje. Metoda polovljenja sistematski smanjuje (polovi) zatvoreni interval. Pre svakog polovljenja izvršava se jednostavna provera na osnovu koje se donosi odluka koja polovina se dalje polovi. **Polovljenje prestaje kad je trenutni interval dovoljno mali**.

Ova metoda poštuje „Teoremu srednje vrijednosti“ - Ako je f **neprekidna** i ako su

f(a) i f(b) različitog znaka onda funkcija **f ima bar jednu nulu na intervalu [a,b]**.

Loop *********

1. Izračunati polovinu [a,b] c=(a+b)/2
2. Izračunati f(c)
3. Ako f(a) * f(c) < 0 novi interval je [a, c]
   Ako f(a) * f(c) > 0 novi interval je [c, b]
4. Ako |b– a| < tolerance -> vrati c=(a+b)/2 kao rešenje, odnosno vrati se u pelju u suprotnom

End loop *********

KONVERGENCIJA - Ako je f(x) neprekidna i znamo da se rešenje nalazi u [a,b], metoda polovljenja garantovano konvergira. Konvergencija je spora.

Ako je tačno rešenje x’

Ako je greška u k -toj iteraciji $e^k = x^k + x'$

Iterativna metoda konvergira brzinom r ako

$$\lim_{k\to\infty} \frac{||e^{k+1}||}{||e^k||^r} = C$$

- r=1 i C e (0,1) – **linearna konvergencija**
- r=1 i C=0 – superlinearna konvergencija
- r=2 – kvadratna konvergencija

Kod metode polovljenja konvergencija je C= $\frac{1}{2}$ – dakle linearna.

4.3. Objasniti metodu sječice (zeroSecantMethod)!

Pretpostavke:
Dve početne tačke x_i i x_{i-1} takve da važi $f(x_i) \neq f(x_{i-1})$
Sledeća procena, tačka xi+1 dobija se pomoću formule:

$$x_{i+1} = x_i - f(x_i)\frac{(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})}$$

Metoda sječice je otvorena metoda, ne zahtjeva da se rješenje nalzi u intervalu $[x_0, x_1]$.
Brzo konvergira (superlinearno), ne zahtjeva da se rješenje nalazi u intervali, dok mane pretstavljaju to što konvergencija nije uvijek zagarantovana i to što lako izdivergira.

4.5. Objasniti metodu regula falsi (zeroFalsePosition)!

- Zahteva i da na početku važi f(a)*f(b)<0.
- Time dobijamo metodu koja malo sporije ali garantovano konvergira.

U suštini, regula falsi se služi formulom od sječice ali ima uslov $f(x_0) * f(x_1) < 0$ ($f(x_{k-1}) * f(x_k) < 0$).
I kada se nadje nova tacka, gleda se ako je f(nova) * f(stara) < 0 (jer ima i srednja tacka, tri se posmatraju), onda srednja = nova, u suprotnom stara = nova.
Kod metode regula konvergencija je zagarantovana (**zatvorena metoda**), dok sporije konvergira od sječice. Zahtjeva rješenje u početnom intervalu kao i zeroBisection.

4.6. Objasniti Njutnovu metodu (zeroNewton)!

Videli smo da metoda sječice funkciju f(x) aproksimira pravom. Funkcija f(x) aproksimira se tangentom. Poenta je u tome da je tangenta bolja aproksimacija tj. bolje "prati" funkciju f(x), u odnosu na sječicu.
Funkcija f(x) , mora biti neprekidna, i $f(x_0) \neq 0$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Konvergencija Njutnovog metoda **kvadratna**. Kvadratna konvergencija znači da se broj tačnih cifara rešenja duplira u svakoj iteraciji.

Metod	Prednosti	Mane
Polovljenje	- Garantovana konvergencija - Broj iteracija za toleranciju se može unapred odrediti - Laka za implementaciju - Ne zahteva prvi izvod	- Spora konvergencija - Zahteva da se rešenje nalazi u [a,b] tj. da važi f(a)f(b)<0
Sečica	- Brza konvergencija (sporija od Njutnove metode) - Ne zahteva prvi izvod	- Nema garantovanu konvergenciju - Zahteva dve početne tačke takve da važi f(x0)≠f(x1)
Regula falsi	- Garantovana konvergencija (brža od polovljenja, sporija od sečice) Ne zahteva prvi izvod	- Zahteva dve početne tačke takve da važi f(x0)≠f(x1)
Njutnov	- Veoma brza konvergencija (ako je početno rešenje blizu nule)	- Nema garantovanu konvergenciju - Zahteva postojanje prvog izvoda i f'(x0)≠0

5. Prezentacija – Interpolacija

5.1. Šta je interpolacija tačaka?

Problem: odrediti vrednosti nepoznate funkcije $f: x \rightarrow y$, $y \in \mathbb{R}$. Funkcija f je nepoznata u smislu da **ne poznajemo tačnu matematičku formulu zavisnosti y od x**

Aproksimiraćemo funkciju f drugom funkcijom g : $g \approx f$

Iterpolacija:

**funkcija g mora da prođe kroz svaku od poznatih tačaka xi,yi

Najčešće, za funkciju g koristimo polinom (polinomijalna interpolacija).

Za datih N tačaka, polinom stepena $N-1$ koji prolazi kroz svaku od datih tačaka je jedinstven.

Zbog konzistentnosti sa MATLAB-om $g(x)$ ćemo zapisivati kao „standardni zapis“:

$$g(x) = p_1 x^{N-1} + p_2 x^{N-2} + \cdots + p_{N-1} x + p_N$$

(umesto $g(x) = a_1 + a_2 x + a_3 x^2 + \cdots + a_N x^{N-1}$)

Za sve uredjene parove (x_i,y_i) mora da važi $g(x_i) = y_i$. Ako imamo n uredjenih parova, to definiše sistem on N jednačina sa N nepoznatih međutim **rješavanje sistema nije uvijek lako a i problem je ako su loše uslovljeni sistemi**.

Ako je cond(A) (kondicioni broj nesigularne kv. Matrice) prevelik sistem je čoše uslovljen.

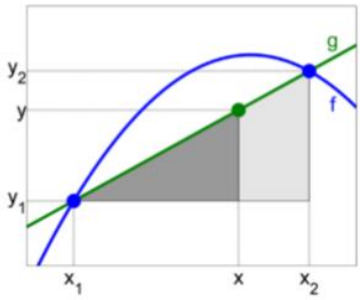
5.2. Objasni Njutnovu interpolaciju!

Koriste se Njutnove “podeljene (konačne) razlike” da bi se odredili koeficijenti polinoma koji ima oblik:

$$f_{n-1}(x) = b_1 + b_2(x - x_1) + b_3(x - x_1)(x - x_2) + \cdots + b_n(x - x_1)(x - x_2)\cdots(x - x_{n-1})$$

Koeficijenti nižeg reda (stepena) ostaju isti kad se poveća stepen polinoma, stoga je lako dodavati nove podatke (tačke) i uraditi interpolaciju polinomom većeg stepena.
Ovo je velika prednost zapisa polinoma u Njutnovoj formi nad zapisom u „standardnoj" formi.

Interpolacioni polinom je prava: $g(x) = p_1x + p_2$
Njutnova forma: $g(x) = b_1 + b_2(x - x_1)$



Sličnost trouglova (jedan od načina da se odredi jednačina date prave):

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

Tj. $b_1 = y_1$ i $b_2 = (y_2 - y_1)/(x_2 - x_1)$

Njutnova linearna interpolaciona formula jeste jednačina prave kroz dvije tačke.

**Njutnova kvadratna interpolacija. U početnom skupu imamo 3 para tačaka. Pravimo polinom 2. stepena. Koji ima oblik: $g_2(x) = b_1 + b_2(x - x_1) + b_3(x - x_1)(x - x_2)$,
gdje su $b_1 = y_1$, b_2 se dobije kada se umjesto x uvrsti x_2, i onda kada se dobije b_1, b_2 onda kad za x ubacimo x_3 dobijamo $g(x_3) = y_3$ i onda b3 je jedinstveno određen. Koeficijent b_1 je translacija po y osi Njutnovog polinoma, b_2 je nagib, dok je b_3 zakrivljenost tog polinoma 2. stepena.

Povećanjem stepena polinoma raste tačnost aprkosimacije polinomom.
Opšta formula za Njutnov polinom:

$$g_{n-1}(x) = b_1 + b_2(x - x_1) + b_3(x - x_1)(x - x_2) + \cdots + \cdots + b_n(x - x_1)(x - x_2)(x - x_3)\cdots(x - x_{n-1})$$

Konačne razlike se koriste da bi se dobili koeficijenti b_i :

$$f[x_n, x_{n-1}, \ldots, x_2, x_1] = \frac{f[x_n, x_{n-1}, \ldots, x_3, x_2] - f[x_{n-1}, x_{n-2}, \ldots, x_2, x_1]}{x_n - x_1}$$

Iterativni algoritam:

1. Izračunati sve konačne razlike prvog reda pomoću vrednosti funkcije f(xi).

2. Izračunati sve konačne razlike drugog reda koristeći razlike prvog reda.

3. Nastaviti proces do razlika n-tog reda.

**** Ne moramo da rešavamo sistem jednačina**

• Važno je napomenuti da tačke $x_1, x_2, \ldots, x_N$ ne moraju biti uniformno raspoređene (na jednakim rastojanjima)

• Takođe, ne mora da važi $x_1 < x_2 < \cdots x_N$ – tačke mogu biti raspoređene u bilo kom redosledu

5.3. Objasni Lagranžovu interpolaciju!

Lagranžov polinom ima sledeći oblik:

$$g_{N-1}(x) = L_1(x) f(x_1) + L_2(x) f(x_2) + \cdots + L_N(x) f(x_N) = \sum_{i=1}^{N} L_i(x) f(x_i)$$

$$L_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{N} \frac{x - x_j}{x_i - x_j} = \frac{P_i(x)}{P_i(x_i)}$$

$$= \frac{(x - x_1)(x - x_2) \cdots\cdots (x - x_{i-1})(x - x_{i+1}) \cdots\cdots (x - x_N)}{(x_i - x_1)(x_i - x_2) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_N)}$$

$L_i(x)$ zavisi isključivo od vrednosti x tačaka u kojima se meri funkcija (ne i od vrednosti same funkcije u tim tačkama)

• Odatle sledi da je Lagranžov oblik polinoma pogodan za slučajeve kada imamo više različitih skupova y_i za iste x_i

• Zgodno kada imamo veličinu koja se meri uvek u istim trenucima x_i

• Za razliku od Njutnovog oblika polinoma, Lagranžov oblik polinoma nije toliko pogodan za dodavanje novih tačaka

5.4. Šta je inverzna interpolacija?

Inverzna interpolacija:

Šta bi bilo ako želimo da odredimo u kojoj tački x funkcija f(x) ima vrednost k?

1. Zameniti x i f(x) i uraditi interpolaciju. Međutim, razmak između y je obično veoma ne-uniforman (za razliku od x) što rezultuje u oscilacijama u

interpolacionom polinomu 2. **Interpolirati f(x) u tačkama x – dobijamo g(x). Upotrebiti metode za određivanje nula funkcija da bi pronašli x takvo da je g(x)=k**

5.6. Šta je ekstrapolacija?

Ekstrapolacija:
Problem koji se dešava izvan intervala tačaka koje smo proslijedili za aproksimaciju polinom funckije f(x). Problem je što se u tim intervalima funkcija ponaša veoma čudno, može naglo da raste/opada.

5.7. Koji su problemi kod interpolacije?

Osnovni problemi koji se javljaju kod interpolacije jesu situacije kada imamo oscilacije polinoma velikog stepena. Iako postoje određene situacije u kojima su nam polinomi velikog stepena od koristi, u većini primena ih izbegavamo. Polinomi nižeg stepena obično mogu efektivno da opišu trend koji postoji u krivoj, a da ne uvedu problem velikih oscilacija.

5.8. Objasni interpolaciju SPLAJ-nom!

Ako imamo N tačaka, možemo Njutnom ili Lagranžom odrediti polinom N-1 stepena, međutim to može dovesti do loših rezultata zbog oscilacija i grešaka pri zaokruživanju.
** splajn ≈ “podinterval”
Koristimo više polinoma nižeg stepena da interpoliramo date tačke (na svakom podintervalu početnog intervala određuje se polinom), i u zavisnosti od broja tačaka koje koristimo u intervalu imamo:

a) Linearni splajn – polinom 1. reda – interpolacija se vrši između svake dvije tačke tako što povlačimo pravu kroz te dvije tačke. Međutim linearni splajn nije gladak i u tačkama spoja dva splajna imamo prekide prvog izvoda polinoma što ne valja za dalju analizu.
b) Kvadratni splajn – polinom 2. reda
c) Kubni splajn – polinom 3. reda –
    1. Prirodni splajn - Pretpostavimo da su drugi izvodi u kranjim tačkama jednaki nuli. Aproksimaciona funkcija postaje prava linija u krajnjim čvorovima. Više simetričan u odnosu na kvadratni splajn (gde je to samo za 1. interval).
    2. Stegnuti splajn - Pretpostavimo da su prvi izvodi u prvoj i poslednjoj tački konstante date unapred. Npr. ako je je konstanta 0, splajn postaje horizontalan na krajevima.

3. “Not-a-know” splajn - Pretpostaviti jednakost trećeg izvoda u drugoj i preposlednjoj tački (u prethodnim uslovima smo već specificirali kontinuitet prvog i drugog izvoda). Ovaj uslov će rezultovati time da su susedni splajnovi za prva dva segmenta međusobno jednaki, kao i to da su susedni splajnovi za poslednja dva segmenta međusobno jednaki. Ime “Not-a-knot” potiče od činjenice da druga i pretposlednja tačka nisu više čvorovi, tj. nisu više spoj različitih splajn funkcija.

6. Prezentacija – Aproksimacija funkcija (REGRESIJA)

6.1. Objasni pojam regresije i aproksimaciju funkcije regresivnom metodom!

Regresija pruža alternativni način za rešavanje problema sa velikim skupovima tačaka. Regresija ima za cilj da pronađe funkciju koja se “najbolje uklapa” u podatke uz sljedeće dvije pretpostavke:

• rezultujuća funkcija ne mora da prođe kroz svaku tačku.

• ako koristimo polinom, njegov stepen ne zavisi od broja tačaka, već ga zadaje korisnik.

• Interpolacija: **pronalazimo funkciju koja prolazi tačno kroz sve zadate tačke (podatke)**

• Regresija: **pronalazimo funkciju koja se „najbolje uklapa“ u zadare tačke (podatke)**

• Mogu da rezultuju veoma različitim funkcijama

Primjer regresije: Određivanje cijene nekretnina (ima puno podataka, veliki stepen polinoma, za interpolaciju to je jako loše, takođe kod interpolacije bi to značilo da sve kuće sa istom kvadraturom imaju istu cijenu)

U regresiji nisu bitni šumovi, mogu postojati, regresija traži neki trend u podacima I pokušava to opisati funkcijom.

Primjer interpolacije: Rekonstrukcija zvučnog signala iz digitalnog zvuka.

Kod interpolacije su uglavnom precizna mjerenja, interpolacija se trudi da pronađe aproksimaciju polinomom koji prolazi kroz sve zadate tačke.

6.2. Regresija opšti problem! Pojasniti pojam “Mjerenje kvaliteta regresione funkcije” i koeficijent determinacije!

Regresija – jedna od strategija je regresija najmanjih kvadrata (Least Squares Regression)

• Nepoznate koeficijente funkcije g ćemo odrediti minimizacijom sume kvadrata grešaka:

$$\sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - g(x_i))^2$$

• **SSR – sum of squers regresion – SSR =** $\sum(\tilde{y} - \bar{y})^2$

• **SSE – sum of squares error**

• **r^2 – koeficijent determinacije -> r² =** $\frac{SSR}{SSE}$

• **s – tipična greška procjene -> s =** $\sqrt{\frac{SSE}{n-m-1}}$ (n – broj tačaka, m- broj nezavisnih promjenljivih (naš slučaj 1))

Koeficijent korelacije - Predstavlja idikator "snage" (povezanosti) linearnog odnosa dve promenljive.

$$\mathbf{r} = \frac{\sum(x-\bar{x})(y-\bar{y})}{(n-1)s_x s_y}$$

Vrednost koeficijenta korelacije je u intervalu [-1,1] i tumači se na sledeći način:

• Vrednosti blizu 1 znače jaku pozitivnu korelaciju

  • ako se povećava x povećava se y

• Vrednosti blizu -1 znače jaku negativnu korelaciju

  • ako se povećava x smanjuje se y

• Vrednost blizu 0 znači da ne postoji korelacija između x i y

<u>Regresija se može uopštiti za pronalaženje polinoma bilo kojeg stepena.</u>

6.3. Objasniti <u>regresiju polinomom proizviljnog stepena!</u>

Zato, umesto da inisitiramo na tome da ovaj polinom manjeg stepena prođe kroz svih n tačaka (jer je to značila jednakost u prethodnom sistemu), polinom uklapamo u tačke tako da minizujemo zbir kvadrata grešaka:

$$\Phi(a,b) = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (f(x_i) - y_i)^2$$

$$(A^T \cdot A) \cdot a = A^T \cdot y,$$

gde je:

$$A = \begin{bmatrix} x_1^0 & x_1^1 & \cdots & x_1^M \\ x_2^0 & x_2^1 & \cdots & x_2^M \\ \vdots & \vdots & \ddots & \vdots \\ x_N^0 & x_N^1 & \cdots & x_N^M \end{bmatrix}_{N\times(M+1)}, \quad a = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_M \end{bmatrix}_{(M+1)\times 1}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}_{N\times 1}$$