Metoda „triedara“ za nalaženje tačnog rešenja linearnog sistema sa velikim brojem nepoznatih

Aleksa Srdanov, Radiša Stefanović, Dragan Milovanović,
Aleksandra Jovanović i Đorđe Marjanović

Visoka tehnička škola strukvnih studija Požarevac
Požarevac, Srbija
aleksa.srdanov@vts-pozarevac.edu.rs, radisastefanovic@yahoo.com, draganmilovan@gmail.com,
beba3602@gmail.com, djordje58@gmail.com

Sažetak— **Pokazaćemo da je moguće definisati algoritam za nalaženje tačnog rešenja sistema linearnih jednačina sa velikim brojem nepoznatih, koji se završava posle konačno mnogo koraka bez postupka njegovog rešavanja. Svaka jednačina linearnog sistema je jedna hiperravan unutar prostora čija je dimenzija jednaka broju njegovih nepoznatih. Predloženi algoritam do rešenja dolazi korišćenjem jednog invarijan- tnog svojstva „triedara“ koje obrazuju po tri date hiperravni. Ukoliko rešenje postoji, metoda koja relativno brzo konvergira, uvek dovodi do tačnog rešenja linearnog sistema. Osim toga, ne menja početno date koeficijente sistema i ne pravi neočekivane underflow - overflow međurezultate. Predložena metoda može se i ubrzati ako se za rešenje može dozvoliti da bude određeno sa unapred zadatom greškom.**

Ključne reči- metoda triedara; sistem jednačina sa velikim brojem nepoznatih.;

I. UVOD

Osmišljavanje metode kojom bi se došlo do rešenja linearnog sistema sa velikim brojem jednačina i nepoznatih je uvek aktuelan. Kada računari dosegnu brzinu i memorijski kapacitet za sisteme od sto hiljada jednačina sa sto hiljada nepoznatih odmah se javlja zahtev da se isti problem reši kada sistem ima milion jednačina i nepoznatih. Svaki postupak koji bi težio postati metoda za rešavanje ovakvog problema neraskidivo prati provera i analiza sledećih parametara: 1. Koliko memorijskog prostora je potrebno za realizaciju postupka. 2. Koliko operacija je potrebno izvršiti za dobijanje tačnog (približnog) rešenja. 3. Kako se metoda snalazi sa problemima u koje spadaju: stabilnost rešenja, izračunljivost (mogućnost uzrokovanja overflow-underflow efekata, uslovi konvergencije, brzina konvergencije.

Od velike koristi bi bilo ako bismo mogli definisati metodu koja: 1) Potrebuje što je moguće manje računarske memorije; 2) Zahteva što je moguće manji broj algebarskih operacija za izvršenje; 3) Ne proiz- vodi neočekivane overflow - underflow efekte; 4) Nema problema vezanih oko stabilnosti rešenja; 5) Primenljiva je na proizvoljano veliki sistem jedna- čina i 6) Bezuslovno i brzo konvergira.

U radovima [3], [4] opisane su metode koje ispunjavaju sve navedne uslove. Međutim u slučaju kada su uglovi koje hiperravni (koje su određene jednačinama polaznog sistema) obrazuju veoma mali navedene metode „sporo“ konvergiraju. Da bi se i taj slučaj obuhvatio predlaže se metoda koja je u ovom radu izložena.

A. *Metoda „triedra“ za* $n = 3$.

Ideja metode „triedra“ zasniva se na sledećem algoritmu, kao jednom od mogućih načina nalaženja tačnog rešenja sistema 3x3, bez postupka njegovog rešavanja. U 3-D prostoru svaka jednačina sistema 3x3 je jednačina jedne ravni. Rešenje sistema je vrh triedra koji te tri ravni međusobno formiraju. Do vrha triedra je moguće doći tako što se u jednoj od ravni uoče proizvoljne tri njene nekolinearne tačke. Te tačke određuju bar dve razne prave koje prodiru drugu ravan u dve razne tačke. Te dve tačke određuju pravu koja prodire treću ravan u temenu triedra – zajedničkoj tački svih triju ravni (rešenju sistema).

B. *Postavka problema*

Pretpostavimo da je u memoriji računara smešten sistem linearnih jednačina n x n (n mnogo veći od 1000). Neka ima oblik:

$$\begin{aligned} &\boldsymbol{a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1} \\ &\boldsymbol{a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n = b_2} \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ &\boldsymbol{a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,n}x_n = b_n} \end{aligned} \quad (1)$$

Za ovaj rad bitne su pretpostavke da sistem (1) ima rešenje i da prilikom provere tog rešenja u bilo kojoj njegovoj jednačini ne dolazi do overflow prekoračenja u međurezulta- tima, jer ako bi se to dešavalo onda je sistem nemoguće tačno rešiti u takvom okruženju bilo kojom metodom. Bez mogućnosti provere rešenja nije moguće tvrditi da je neka dobijena n-torka zaista tačno rešenje sistema.

Da bismo došli do tačnog rešenja sistema (1) nije ga neophodno rešavati. Takav postupak bi u osnovi bio matematički (računski) i doveo bi do pojave barem jednog od već navedenih problema. Za nalaženje tačnog rešenje sistema

(1) pogodno je primeniti kibernetički pristup. Svaka jednačina u zadatom sistemu nosi veliki broj invarijantnih informacija. Jedna od njih je, da je svaka jednačina, geometrijski gledano, jedna hiperravan unutar prostora čija je dimenzija barem koliko ima i nepoznatih. Zatim, svaka hiperravan sadrži u svojoj jednačini koordinate svoje normale itd. Neke od tih informacija je moguće iskoristiti u postupku nalaženja rešenja. Ako rešenje sistema (1) postoji, onda je to tačka preseka svih hiperravni određenih jednačinama sistema (1). Kada se sve hiperravni seku u jednoj tački one obrazuju višedimenzionalni rogalj. Ivice tog roglja predstavljaju preseci svake dve hiperravni. Kako je taj presek čitav prostor čija je dimenzija za jedan manja od dimenzije prostora te dve hiperravni, to je pojam ivice višedi- menzionalnog roglja neuporedivo složeniji od trodimezionalnog slu- čaja. Takođe i pojam vrha nekog triedra nije tačka, kao u trodimenzionom slučaju, već čitav prostor čija je dimenzija samo za dva manja od dimenzija njegovih strana. Međutim, suština se ne menja, svaka tačka tog prostora nosi istu informaciju kao i u trodimezionalnom slučaju.

Zbog pretpostavke o postojanju jedinstvenog rešenja sistema (1) svake tri hiperravni se međusobno seku. Ukupan broj svih vrhova triedara je $\binom{n}{3}$, što može biti prilično veliki broj. Međutim, za predloženu metodu nisu od interesa svi triedri. Broj triedara koji metoda koristi je relativno uvek mali u odnosu na ukupan broj svih mogućih.

II. METODA TRIEDRA – OPŠTI SLUČAJ

Ideja metode „triedra“ je nađena u analogiji sa slučajem $n = 3$. Suština je bila: i) izabrati triedar; ii) odrediti tri tačke u nekoj od strana i iii) odrediti vrh triedra. Mogućnost postavljanja petlje sadržana je u izboru tri tačke u nekoj od strana triedra. Određivanje tri tačke u nekoj od strana postizaće se afinim projektovanjem u odnosu na neku tačku (centar) koju treba odrediti čak i pre triedra. Određivanjem triedra, pa potom njegovog vrha omogućava da se kasnije centar i vrh zamene, odnosno da vrh postane novi centar čime se postiže formiranje petlje.

Na početku centar može biti bilo koje tačka prostora čija je dimenzija jednaka broju nepoznatih. Međutim, praktično se pokazalo da je najpogodnije da to bude tačka koja predstavlja težište sistema (1). Koordinate tačke koja predstavlja centar mase sistema (1) dobijamo na sledeći način. Označimo sumu svih koeficijenta kolona sistema (1) uz nepoznate sa $A_i = \sum_{j=1}^{n} a_{i,j}$ i sa $B = \sum_{i=1}^{n} b_i$, $i = 1,2, \dots, n$. Tada, ako je $A_i \neq 0$ onda je $x_i = \frac{1}{n} \cdot \frac{B}{A_i}$ u suprotnom $x_i = \frac{1}{n} \cdot B$, za sve $i = 1,2, \dots, n$. Za centar mase sistema (1) biramo tačku koja ima koordinate $(x_1, x_2, \dots, x_n)$.

Težište sistema (1) je udaljeno od svih hiperravni za merne brojeve koje treba odrediti po poznatim formulama za odstojanje tačke od hiperravni. Na osnovu dobijenih odstojanja imamo na raspolaganju više mogućnosti za izbor triedra. Ako bismo birali uvek tri najmanja odstojanja i konvergencija ka cilju bi bila saglasna tom izboru. Metoda bi sporije konvergirala. Ako bismo birali tri najdalje hiperravni efekat bi bio sličan. Dobijeni koraci bi na početku veoma oscilovali pre nego što bi se ujednačili i počeli da se uravnoteženo smanjuju. Praktično se pokazalo da je najpogodnije birati ili dve najbliže i jednu najdalju ili dve najdalje i jednu najbližu hiperravan kao strane u izboru triedra.

Kada se izabere triedar potrebno je odrediti tri tačke u jednoj od strana. Za to koristimo prodore vektora normala tih strana triedra kroz izabranu tačku (centar) sa jednom njegovom stranom. Posle toga, na osnovu algoritma opisanog za slučaj $n = 3$ određujemo vrh triedra. U sledećem koraku vrh triedra postaje sledeći centar i ceo postupak ponavljamo.

U metodi triedra pojavljuje se više posebno važnih detalja. Neki od njih su: a) Kako najefektnije izabrati tri hiperravni za formiranje triedra; b) Koje invarijantno svojstvo ima vrh triedra i kako se to svojstvo može upotrebiti; c) Na koji način je moguće triedrima dosegnuti jedinstvenu zajedničku tačku?; d) Kako prepoznati da li se približavamo rešenju i kada je ono dosegnuto?; e) Može li se u toku izvršenja algoritma utvrditi da je sistem nemoguć ili neodređen? f) Da li metoda uvek konvergira, kojom brzinom, ...?

A. Izbor tri hiperravni.

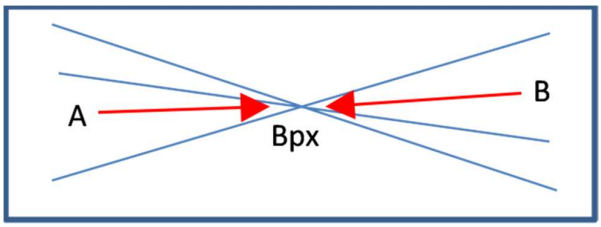
Moguće je formulisati više različitih načina određivanja tri hiperravni kao stranice triedra. Koju taktiku izabrati, najlakše je utvrditi, kada se sastavi algoritam, pa se u toku testiranja malo eksperimentiše.

Praktično se pokazalo da metoda konvergira u svim slučajevima ali da među njima postoji veoma značajna razlika u brzini izvršenja. Tokom testiranja se pokazalo i da nije svejedno u kojoj hiperravni se određuju tri nekolinearne tačke. Na primer, ako se za hiperravan gde se određuju tri tačke izabere najdalja je bolja postavka nego ako se uzme najbliža. Pogotovo kada se značajnije primaknemo rešenju.

U određivanju prodora prava postavljenih kroz centar sa jednom od hiperravni postoje neki specijalni slučajevi koje treba u algoritmu predvideti. Na primer, kada su neke dve ili sve tri hiperravni međusobno upravne. U tim slučajevima neke od postavljenih prava ne moraju seći hiperravan osnove. Tada se taj problem može rešiti na sledeći način. Prvo treba odrediti prodor kroz hiperravan čija je to normala, a iz tačke prodora postaviti normalu na prvu hiperravan čiji prodor može zameniti beskonačno daleku tačku. Takoće, ako jedna (ili više), od dobijene tri tačke, pripada i nekoj drugoj hiperravni tada se mora voditi računa o izboru prava za određivanje ivice diedra osnove i druge hiperravni. U slučaju da sistem nema jedinstveno rešenje postoji još nekoliko specijalnih slučajeva koje je potrebno i moguće algoritmom razrešiti.

B. Invarijantno svojstvo vrha triedra.

Određivanje jedne od tačaka vrha triedra može se postići sa proizvoljne tri nekolinearne tačke smeštene u nekoj od njegovih strana. Vrh bilo kog triedra u odnosu na neku tačku unutar njega (tačka pomoću koje formiramo tri tačke u nekoj od strana) određuje pravac ka rešenju sistema (1), ali ne i smer. Suština metode je zasnovana upravo na tom svojstvu. Posle svakog dosezanja vrha nekog triedra potrebno je još dodatno razrešiti pitanje smera (slika 1.). U protivnom bi metoda činila mnogobrojne „napred-nazad“ korake, što bi značajno umanjilo njenu brzinu. Vrh triedra nije jedinstveno određen u višedimenzionom prostoru. To je samo zajednički naziv iza



Slika 1. Pitanje smera.

koga se krije prostor čija je dimenzija samo za dva manja od prostora njegovih strana.

Svaka tačka takvog prostora zadržava svojstvo vrha triedra. Koju tačku vrha ćemo dosegnuti zavisi od tačke pomoću koje određujemo početne tri tačke. U opštem slučaju, $(n > 3)$, različite trojke određuju „različite vrhove“ triedra, što je prilično zbunjujuće za naše poimanje triedra i njegovog vrha.

Određivanje smera se pokazalo kao vrlo složeno pitanje. Potrebno je osmisliti test koji bi bio sposoban da odredi smer pravca „centar – vrh triedra“ u odnosu na rešenje sistema (1) koji se ponavlja u svakom koraku i pri tom neophodno je da taj test bude što je moguće jednostavniji, sa minimumom dodatnih operacija. Pogotovo što ga treba primeniti na sisteme sa velikim brojem nepoznatih. Kako već određujemo rastojanja dosegnutih vrhova do svih hiperravni logično je iskoristiti te veličine, jer su već određene. Iako matematički nije savršeno tačno, ipak se praktično pokazalo kao veoma zadovoljavajuće, ako za test smera koristimo sume svih odstojanja tačke centara do svih hiperravni. Logično je da, ako sistem ima rešenje hiperravni obrazuju rogalj kome se sve stranice međusobno „približavaju“ ka zajedničkoj tački. Zato se smer određuje ka tački čiji je zbir svih odstojanja do svih hiperavni manji.

Zbog toga je u algoritmu potrebno predvideti da ukoliko vrh triedra, koji je izabran, bude orijentisan u suprotnom smeru bude omogućeno ponavljanje izbora triedra sve dok se ne dobije neki koji je „orijentisan“ na željeni način. Ako se takav ne nađe bira se onaj triedar koji najmanje vraća u nazad.

C. Vrh triedra i rešenje sistema..

Za predloženu metodu dve tačke imaju centralnu i najvažniju ulogu. To su centar pomoću koje nalazimo tri nekolinearne tačke u jednoj hiperravni i vrh triedra. Na početku se polazi iz centra mase sistema (1) kroz koju pomoću normala strana triedra nalazimo tri nekolinearne tačke u jednoj od njegovoj strana. Potom određujemo vrh triedra koji postaje sledeći „centar mase“. Kako je tu tačku, u suštini, uvek moguće uzeti proizvoljno, to sve računske greške koje su do tada bile napravljene bivaju resetovane. Određivanje vrha sledećeg roglja ne sadrži kumulisanu i već stvorenu grešku. Na taj način predložena metoda pravi, odnosno može da napravi grešku, samo u poslednjem koraku. Tačno rešenje sistema (1) je zajednički vrh mnogobrojnih triedara koji kada mu se dovoljno približe mogu ga i jedinstveno odrediti. To je kao gađanje u metu. Što si bliže meti to je lakše pogoditi centar. Kako su svi vrhovi triedra algoritmom „orijentisani“ ka rešenju to je svaki sledeći vrh triedra uvek tačka koja je bliža tački rešenja i samo na osnovu toga do rešenja se stiže skoro ravnomerno po svim koordinatama. Neprekidna zamena te dve tačke i izmena triedara formiraju prostornu logaritamskom spiralu koja vodi ka rešenju sistema.

D. Kriterijum dosezanja rešenja..

Metoda je izgrađena tako da je svaki sledeći vrh triedra sve bliži rešenju sistema. Rešenje sistema je tačka čije je odstojanje od svih hiprravni nula. Svaki vrh triedra koji algoritam odredi ima upravo svojstvo da mu je zbir odstojanja od svih hiperravni sve manji i manji. Otuda, to svojstvo može biti iskorišćeno i kao kriterijum tačnosti. Predložena metoda kako je postavljena je sposobna da dopre do tačnog rešenja. Osim toga, ova metoda može zaustaviti svoj proces približavanja i dati približno rešenje ako smo unapred zadovoljni nekom zadatom tačnošću. Zbog ravnomerne konvergencije po svim koordinatama zadata greška se „deli“ na svaku koordinatu, pa kriterijum zaustavljanja koji algoritam registruje kao parametar Epsilon se može shvatiti kao $n \cdot \varepsilon$, gde je ε mera zadovoljavajuće tačnosti po svakoj komponenti, recimo $\varepsilon = 0.05$, a n broj nepoznatih.

E. Moguć, nemoguć i neodređen sistem.

Za određivanje da li je sistem moguć, nemoguć ili neodređen algoritmi u [3] ili [4] zahtevali su dodatna ispitivanja koja nisu bila u sklopu osnovnog algoritma. Algoritam koji je ovde opisan je sposoban da detektuje da li je sistem nemoguć ili neodređen tokom svog izvršenja. Sve provere se izvršavaju u toku samog algoritma. Analiza započinje formi- ranjem tri tačke u jednoj od strana triedra. Ako su te tri tačke kolinearne ili ako jedna (ili više) od njih pripada i nekoj drugoj ravni, su slučajevi čije objašnjenje daje odgovor o prirodi rešenja sistema (1). Takođe i ako prava određena sa prodornim tačkama u drugoj ravni ne seče treću ravan definiše sistem kao nemoguć. Kada se utvrdi da prave određene početno nađenim nekolinearnim tačkama ne seku drugu hiperravan već joj pripadaju to ima za posledicu da je sistem neodređen. Predloženi algoritam nije neophodno dodatno opremati da bude sposoban da razreši sve moguće slučajeve. Njih je svakako potrebno u algoritmu predvideti i ugraditi zbog korektnosti samog algoritma.

III. IMPLEMENTACIJA ALGORITMA

Pseudokod programa napisanog po upustvima iz ovog rada ima oblik:

```
UcitatiSistem();
PocetnaAproksimacija();
While(s1 < Epsilon) {
   koliko++;   //brojac koraka  izmene triedara
   jedan = najveci(); // izbor tri ravni
   dva = najmanji();
   tri = najmanji();
   triedar(jedan, dva, tri);// odredi vrh
   s1 =Odstojanje();//provera rastojanja nove dosegnute tacke
   koja = 0; // koliko puta se vrsi korekcija
      while(s1 > s) {
            tri = najmanji();
            triedar(jedan, dva, tri);
            s1 = Odstojanje();
            koja++;//broji ponovne izbore
```

```
            if(koja == 4) break;//podredjeno primeru
            }
  for(i = 0; i < 10; i++) {X0[i] = V[i]; D[i] = D1[i];}
                                        //zamena vrha i centra
  s = s1; //novo rastojanje postaje staro
}
IspisRezultata();
```

A. Primer

Neka je dat sledeći sistem linearnih jednačina:

$43x_1-11x_2+13x_3-17x_4+19x_5-23x_6+29x_7-31x_8+37x_9-41x_{10}=-496$

$41x_1-43x_2+11x_3-13x_4+17x_5-19x_6+23x_7-29x_8+31x_9-37x_{10}=-1008$

$37x_1-41x_2+43x_3-11x_4+13x_5-17x_6+19x_7-23x_8+29x_9-31x_{10}=-204$

$31x_1-37x_2+41x_3-43x_4+11x_5-13x_6+17x_7-19x_8+23x_9-29x_{10}=-864$

$29x_1-31x_2+37x_3-41x_4+43x_5-11x_6+13x_7-17x_8+19x_9-23x_{10}=0$

$23x_1-29x_2+31x_3-37x_4+41x_5-43x_6+11x_7-13x_8+17x_9-19x_{10}=-864$

$19x_1-23x_2+29x_3-31x_4+37x_5-41x_6+43x_7-11x_8+13x_9-17x_{10}=204$

$17x_1-19x_2+23x_3-29x_4+31x_5-37x_6+41x_7-43x_8+11x_9-13x_{10}=-1008$

$13x_1-17x_2+19x_3-23x_4+29x_5-31x_6+37x_7-41x_8+43x_9-11x_{10}=496$

$11x_1-13x_2+17x_3-19x_4+23x_5-29x_6+31x_7-37x_8+41x_9-43x_{10}=-1008$

Tačno rešenje ovog sistema je: **x_1=11; x_2=13; x_3=17; x_4=19; x_5=23; x_6=29; x_7=31; x_8=37; x_9=41; x_{10}=43**. U radu [3] taj primer je testiran sa postupkom koji je ovde znatno poboljšan. Tada se dobijao sledeći izveštaj:

Resenje se doseze u 205 poluiterativnih koraka. (20 punih iteracija)

10.9999 12.9998 16.9998 18.9999 23 29.0001 31.0002 37.0002 41.0001 43.

U radu [4] startovanjem odgovarajućeg programa za ovaj primer dobijao se sledeći izveštaj:

Resenje se doseze u 67 poluiterativnih koraka. (6 punih iteracija)

11 13 17 19 23 29 31 37 41 43

U ovom radu moguće je dobiti i približno i tačno rešenje.

Startovanjem predloženog algoritma sa početnim zahtevom da se rešenje dobije sa tačnošću Epsilon=0.1 dobija se sledeći izveštaj:

Resenje je: 11.021 13.0036 17.1225 19.0882 22.9866 29.0394 31.0457 37.0061 40.9853 43.0095

Resenje se doseze prolaskom kroz 33 triedra.

Ukoliko se tačnost poveća dobija se:

Resenje je: 11 13 17 19 23 29 31 37 41 43

Resenje se doseze prolaskom kroz 42 triedra.

B. Konvergencija i brzina.

Predloženi algoritam je potpuno komplementaran algoritmima datim u [3] ili [4]. Zato je pogodon za rešavanje upravo onih sistema koji kada bi se rešavali metodom unutrašnje ili spoljašnje spirale bi veoma sporo konvergirali. Svi primeri koji su u vezi sa stabilnošću rešenja, u metodama gde je stabilnost rešenja prisutna kao problem bili bi za ovu metodu najidealniji primeri za najbržu i najtačniju konvergenciju. Ako su uglovi koje hiperravni međusobno zaklapaju sve manji ova metoda je sve brža i obrnuto. Predložena metoda uvek konvergira kad god sistem (1) ima rešenje.

C. Broj potrebnih operacija i zauzeće memorije.

implementaciju ove metode, na način kako je ovde urađeno, potrebno je $n^2 + 10n$ registara za zapis sistema i elemente pomoćnih nizova, što je hardverski gledano prilično zadovoljavajuće.

Za sračunavanje svih odstojanja potrebno je $n^2 + 4n$ operacija. Za određivanje jedne tačke (prodora jedne prave) potrebno je $6n$ operacija, a za određivanje prve tri tačke $18n$ operacija. Određivanje prave zahteva $8n$ operacija, a za tri korišćene prave ukupno $24n$ operacija. Određivanje najmanjeg ili najvećeg broja potrebuje oko $2n$ operacija. Za izbor strana jednog triedra potrebno je ukupno 6n operacija. U okviru while petlje koja se izvršava konačan broj puta broj operacija je srazmeran sa n^2. Broj koraka koje je potrebno izvršiti za relativno male sisteme jednačina prevazilazi navedenu procenu s obzirom na broj nepoznatih, ali kako broj nepoznatih raste broj potrebnih algoritamskih koraka ne prati taj rast. Za velike sisteme ocena broja potrebnih operacija je upravo srazmerna broju n^2.

Predloženi algoritam treba proširiti i unaprediti i po drugim svojim komponentama. Za postavljeni primer, brzina i tačnost su za ovaj rad bile od veće važnosti od ostalih karakteristika koje su navedene, a po pravilu su pratioci svih poznatih metoda za rešavanje sistema jednačina.

IV. ZAKLJUČAK

Metoda „triedra“ nudi krajnje jednostavan postupak, sa računarske tačke gledišta. Međutim, čak i ovako postavljen problem je primer zadatka kojem je dorastao jedino super računar. Osim toga, ova metoda: 1. Zahteva broj operacija koji je za računar prihvatljiv; 2. Ne prozvodi nekontro- lisani rast međurezultata - osim ako sama veličina rešenja to ne uzrokuje. (Tada je to nemoguće izbeći bilo kojom metodom.); 3. Ne menja koeficijente početno zadatog sistema – zbog toga nema problema sa stabilnošću rešenja; 4. Ako rešenje postoji uvek konvergira. Pri tome postupak može biti započet iz bilo koje tačke prostora rešenja. Konvergencija je ravnomerna po svim koordinatama. Svaka aproksimacija podjednako i ravnomerno odstupa od stvarnog rešenja, jer se rešenje ne računa, već mu se neprekidno približavamo. 5. Veoma jednostavna je za sastavljanje algoritma. 6. Za razliku od postupka u [3], [4] sam proces je sposoban da determiniše kada je sistem moguć, nemoguć i neodređen bez dopunskih provera. 7. Greška može da se pojavi samo u poslednjem koraku – metoda ne kumulira računske greške.

Osim zahteva kako da se dodatno ubrza predložena metoda, nameće se i problem nalaženja jednostavnijeg kriterijuma za proveru tačnosti, a posebno određivanja „smera“ vrha triedra.

LITERATURA

[1] J. Stoer, R. Bulirsch: Introduction to Numerical Analysis, Texts in Applied Mathematics 12, Springer 2002.

[2] N. Higham: Accuracy and Stability of Numerical Algorithms 2000.

[3] A. Srdanov, R. Stefanović. Kako rešiti linearni sitem sa ekstremno mnogo nepoznatih. INFOTEH, Jahorina, 2017.

[4] (*)A. Srdanov, R. Stefanović, D. Milovanović, A. Jovanović. The Method of "External Spiral" for Solving Large System of Linear Equations. Vojnotehnički Glasnik, volumen 66, br. 2, april-jun 2018.

[5] Zvonimir Boht. Numericne metode. Drzavna zalozba Slovenije. Ljubljana 1978.

(*) Rad prihvaćen i biće objavljen u navedenom broju

ABSTRACT

We show that it is possible to define an algorithm for finding the exact solution of a system of linear equations with a large number of unknowns, which ends after finitely many steps without the process of solving it. Every system of linear equations is one hyperplane within the space whose dimension is equal to the number of its unknowns. The proposed algorithm solution finds using an invariant properties of "trihedron" which form three hyperplanes. If the solution exists, a method that converges relatively quickly, always leads to the exact solution of the linear system. In addition, it does not change the initial system coefficients and does not make unexpected underflow - overflow intermediate results. The proposed method can be accelerated if the solution can be allowed to be with a predetermined fault.

METHOD OF TRIHEDRALS FOR FINDING THE EXACT SOLUTION OF A LINEAR SYSTEM WITH A LARGE NUMBER OF UNKNOWNS

Aleksa Srdanov, Radiša Stefanović, Dragan Milovanović, Aleksandra Jovanović and Đorđe Marjanović