NUMERIČKI ALGORITMI I NUMERIČKI SOFTVER

Uvod

predavač:
Aleksandar Kovačević

Šta su numerički algoritmi?

- Skup tehnika pomoću kojih se matematički problemi
- formulišu kao problemi koji se mogu rešiti aritmetičkim i logičkim operacijama

$$\frac{\partial f}{\partial x} \qquad \int$$

$$f(x) = 0$$

Linear Algebra

\+ , -, x, /,

if then

else

> , =, <

loop

Šta su numerički algoritmi?

- To su algoritmi ili metode koje nam pružaju mogućnost da rešimo određene vrste matematičkih problema.

- To su matematički problemi za koje ne postoji analitičko rešenje ili analitičko rešenje dobijamo previše sporo.

Šta je analitičko rešenje?

- Pod analitičkim rešenjem smatramo rešenje koje se sastoji od logičkih koraka koji nam uvek daju tačno rešenje.

- Na primer, kvadratna jednačina:

$$ax^2 + bx + c = 0$$

  ima sledeće analitičko rešenje:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

- Međutim za jednačine stepena većeg ili jednakog od 5 ne postoji analitičko rešenje.

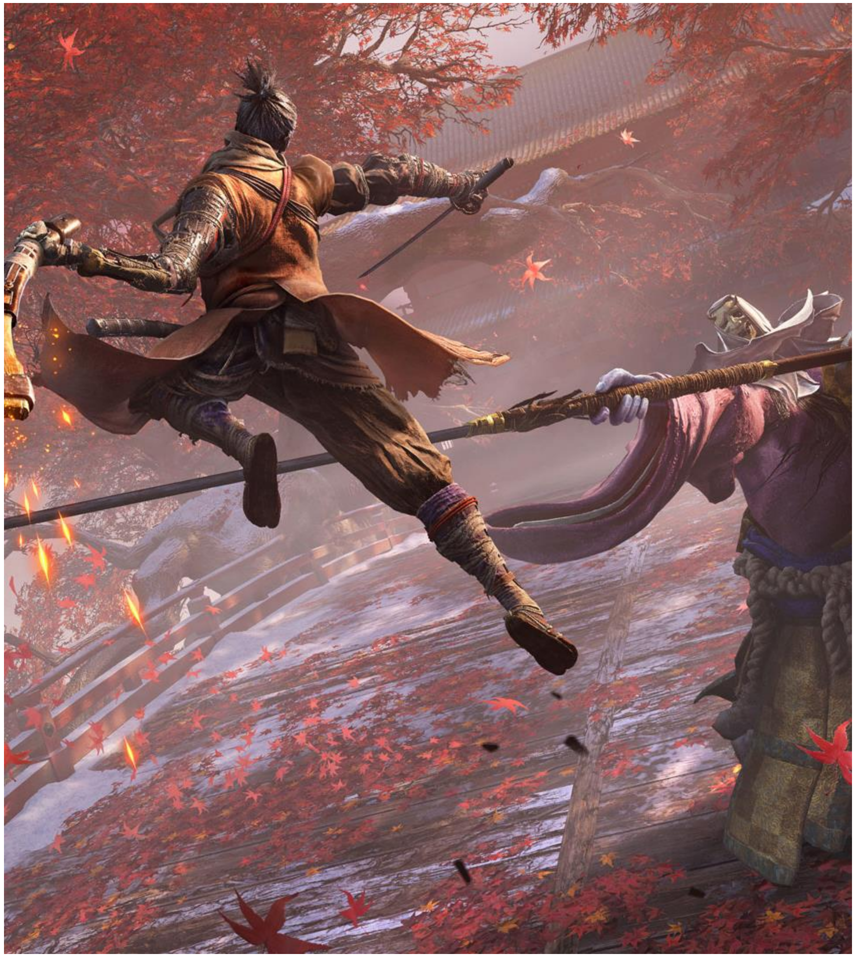
Rešenje dobijeno numeričkim algoritmom

(Često) nije tačno rešenje

već aproksimacija tačnog rešenja
(dovoljno dobra za praktičnu primenu)

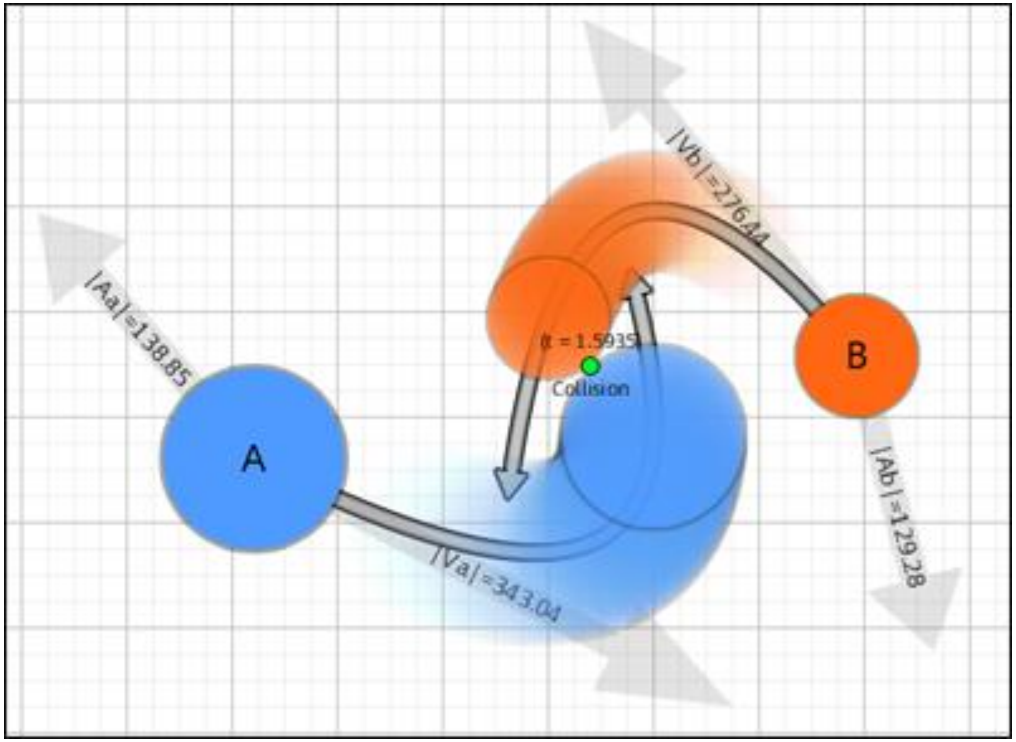
Primer upotrebe numeričkih algoritama

- Rešavanje jednačina je krucijalno za izvršavanje kompjuterskih igara.
- Detekcija kolizije dva objekta može se svesti na problem rešavanja jednačine.



Primer upotrebe numeričkih algoritama

- Određivanje trenutka kolizije dve sfere koje sre kreću sa ubrzanjem svodi se određivanje nule polinoma četvrtog stepena.



Jedan primer upotrebe numeričkih algoritama

- „Jednostavno“ analitičko rešenje problema sa prethodnog slajda.

```
t = -(b/(4 a))-1/2 Sqrt(b^2/(4 a^2)-(2 c)/(3 a)+(2^(1/3) (c^2-3 b d+12 a e))/(3 a (2 c^3-9 b c d+27 a
d^2+27 b^2 e-72 a c e+Sqrt(-4 (c^2-3 b d+12 a e)^3+(2 c^3-9 b c d+27 a d^2+27 b^2 e-72
a c e)^2))^(1/3))+(1/(3 2^(1/3) a))((2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e+Sqrt(-4 (c^
2-3 b d+12 a e)^3+(2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e)^2))^(1/3)))-1/2 Sqrt(b
^2/(2 a^2)-(4 c)/(3 a)-(2^(1/3) (c^2-3 b d+12 a e))/(3 a (2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e+Sqrt(
-4 (c^2-3 b d+12 a e)^3+(2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e)^2))^(1/3))-(1/(3 2^(1/3) a))((2 c
^3-9 b c d+27 a d^2+27 b^2 e-72 a c e+Sqrt(-4 (c^2-3 b d+12 a e)^3+(2 c^3-9 b c d+27 a d^2+
27 b^2 e-72 a c e)^2))^(1/3))-(-(b^3/a^3)+(4 b c)/a^2-(8 d)/a)/(4 Sqrt(b^2/(4 a^2)-(2 c)/(3 a)+(2^(1/3) (c^2-3 b d+12
a e))/(3 a (2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e+Sqrt(-4 (c^2-3 b d+12 a e)^3+(2 c^3-9 b c d+27 a d^2+
27 b^2 e-72 a c e)^2))^(1/3))+(1/(3 2^(1/3) a))((2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e+Sqrt(-4 (
c^2-3 b d+ 12 a e)^3+(2 c^3-9 b c d+27 a d^2+27 b^2 e-72 a c e)^2))^(1/3)))))
```

- Očigledno je da postoji potreba za numeričkim algoritmima.

Još jedan primer upotrebe numeričkih algoritama

Fast inverse square root algoritam za brzo izračunavanje $\frac{1}{\sqrt{x}}$

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y  = number;
    i  = * ( long * ) &y;                       // evil floating point bit level hacking
    i  = 0x5f3759df - ( i >> 1 );               // what the fuck?
    y  = * ( float * ) &i;
    y  = y * ( threehalfs - ( x2 * y * y ) );   // 1st iteration
//  y  = y * ( threehalfs - ( x2 * y * y ) );   // 2nd iteration, this can be removed

    return y;
}
```

https://en.wikipedia.org/wiki/Fast_inverse_square_root

Kako funkcionišu numerički algoritmi?

1. Krećemo od nekog početnog rešenja
2. Pomoću algoritma generišemo nove predloge za rešenja
3. Ponavljamo 2. dok ne odlučimo da zaustavimo algoritam
4. Vraćamo poslednji predlog rešenja kao konačno rešenje

Kako znamo da nas približava tačnom rešenju?

Generalno, ne znamo. Ali…

za većinu numeričkih algoritama postoji dokaz o konvergenciji

pre nego što primenimo algoritam, možemo da proverimo da li će on raditi za naš problem

Kako funkcionišu numerički algoritmi?

Numerički algoritmi generalno ne vraćaju tačno rešenje već njegovu aproksimaciju

Ako znamo da numerički algoritam konvergira ka tačnom rešenju zašto bismo kao rezultat vratili aproksimaciju?

Kod numeričkih algoritama postoji „nagodba“ između tačnosti rešenja i brzine izvršavanja algoritma

Primer jednostavnog numeričkog algoritma

- Problem: odrediti vrednost $\sqrt{2}$

- Ako je $\mathrm{a} = \sqrt{2}$, onda mora da važi:

$$a^2 = 2$$

$$a = \frac{2}{a}$$

Dakle, imamo način da proverimo da li smo pronašli tačno rešenje

Ali kako da generišemo predloge rešenja?

Primer jednostavnog numeričkog algoritma

1. Zadajemo, recimo na slučajan način, neko početno rešenje a_0
2. Proverimo da li za a_0 važi $a = \frac{2}{a}$
   a) Važi $\rightarrow$ pronašli smo rešenje
   b) Ne važi $(a_0 \neq \frac{2}{a_0})$ $\rightarrow$ moramo da smislimo način kako da levu i desnu stranu jednakosti približimo jednu drugoj da bi vremenom postale jednake

Za sledeće rešenje uzeti artimetičku sredinu leve i desne strane jednakosti: $a_1 = \frac{a_0 + \frac{2}{a_0}}{2}$

Primer jednostavnog numeričkog algoritma

- Na taj način došli smo do sledećeg numeričkog algoritma za određivanje korena iz 2:

$$a_0 > 0 \qquad a_{n+1} = \frac{a_n + \frac{2}{a_n}}{2} = \frac{a_n}{2} + \frac{1}{a_n}$$

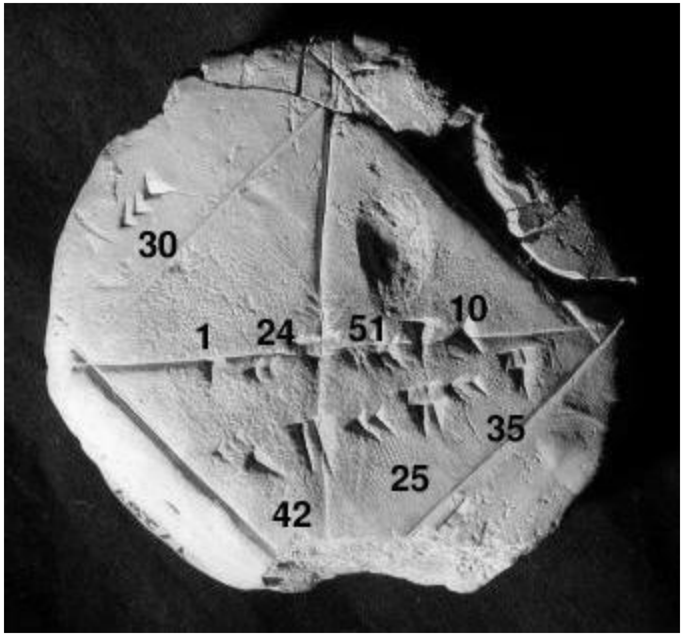
- Primer:

$$a_0 = 1 \qquad a_1 = \frac{1 + \frac{2}{1}}{2} = \frac{3}{2} \qquad a_2 = \frac{\frac{3}{2} + \frac{2}{\frac{3}{2}}}{2} = \frac{17}{12}$$

$$a_3 = \frac{\frac{3}{2} + \frac{2}{\frac{17}{12}}}{2} = \frac{577}{408} = 1.414215\ldots \qquad \sqrt{2} = 1.41421356$$

Istorija

- Pomoću vrlo jednostavnog algoritma
- posle samo 3 iteracije
- stigli smo do rešenja koje od tačne vrednosti razlikuje tek na 6. cifri posle decimalne tačke.



Prikazani algoritam za izračunavanje $\sqrt{2}$ izmislili su Vavilonci još davne 1600 B.C., i smatra prvim numeričkim algoritmom

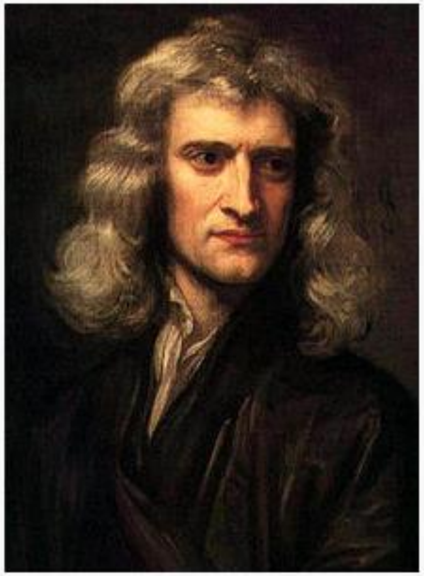
http://www.math.ubc.ca/~cass/Euclid/ybc/ybc.html

Istorija

Nakon Vavilonaca, mnogi veliki matematičari razvijali su numeričke algoritme

Carl Friedrich Gauss

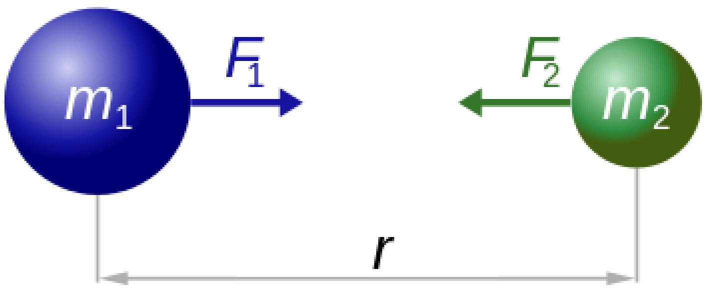
Sir Isaac Newton



Leonhard Euler

Istorija

- Njutnov rad na matematičkom modelu efekata gravitacije, tj. Univerzalni zakon gravitacije:



$$F_1 = F_2 = G\frac{m_1 \times m_2}{r^2}$$

- Ispirisao je mnoge matematičare i fizičare da modeluju pojave iz realnog sveta (dinamika fluida, elektricitet, magnetizam, kvantna mehanika itd.).

- Takvi modeli su tipično diferencijalne jednačine koje je teško ili nemoguće rešiti analitički, pa su zato rešavane numeričkim metodama.

Istorija

- Nagli rast u razvoju i upotrebi numeričkih algoritama kreće krajem 1940-tih godina, zbog razvoja računara.

- Računari su jakooooo efikasni u računskim operacijama i ponavljanju istih radnji, što je najznačajnije za numeriku.

- Jedan od najvećih umova dvadesetog veka Džon fon Nojman shvatio je značaj upotrebe računara za izvršavanje numeričkih algoritama i svojim radom:

  "Numerical Inverting of Matrices of High Order"
  (Bulletin of the AMS, Nov. 1947)

- postavio temelje moderne numeričke analize.

Istorija

- Nakon uvođenja računara primena numeričkih algoritama dobija nove sinonime:
  - *Scientific computing* i *Computational science*

- Pored efikasnosti, računari doneli su neke neželjene efekte u upotrebi računara. Zašto?

- Računari su najefikasniji kada rade se brojevima koji su reprezentovani u konačnoj preciznosti (*finite precision*).

- To znači da neki brojevi (npr. 0.1) nisu nikad potpuno tačno reprezentovani na računaru.

Istorija

- Džon fon Nojman i mnogi drugi načunici su sa pravom
- upozoravali na neželjene efekte primene računara za izvršavanje numeričkih algoritama.

- Međutim, preciznost računara u reprezentaciji brojeva (16 decimalnih cifara u *double* preciznosti po IEEE standardu) daleko premašuje preciznost koja postoji bilo gde u prirodi*.

* Pogledati sekciju o numeričkoj analizi profesora Lloyd Nicholas Trefethena
https://people.maths.ox.ac.uk/trefethen/NAessay.pdf

- Zbog toga numeričke algoritme srećete svakog dana (GPS, simulacije fizike u igrama, računarska grafika, prognoza vremena itd.).

Karakteristike numeričkih algoritama

- Ne radimo sa simbolima već numeričkim vrednostima
- Ne tražimo analitička rešenja već numerička
- Ne tražimo egzaktna rešenja već približna
  - dovoljno dobra za praktičnu primenu

Motivacija

- Numerički algoritmi se koriste za rešavanje mnogo različitih problema u računarstvu:

- Fizika u komjuterskim igrama
    - kretanje, uticaji sila, detekcija kolizija objekata, simulacija tkanine, simulacija tečnosti, kretanja čestica…
- Prediktivno modelovanje
    - predikcija ishoda u sportskim utakmicama
    - predikcija cene akcija, kursa valute…
- Razni drugi primeri
    - rangiranje u društvenim mrežama, GPS, grafika, generisanje terena...

Motivacija

- Postajete sposobni da efikasno koristite postojeći numerički softver, konkretno Python, Matlab ili Octave*.

- Poznavanjem numeričkih algoritama možete inteligentno da koristite softver

- Softver nije više *black box* za vas

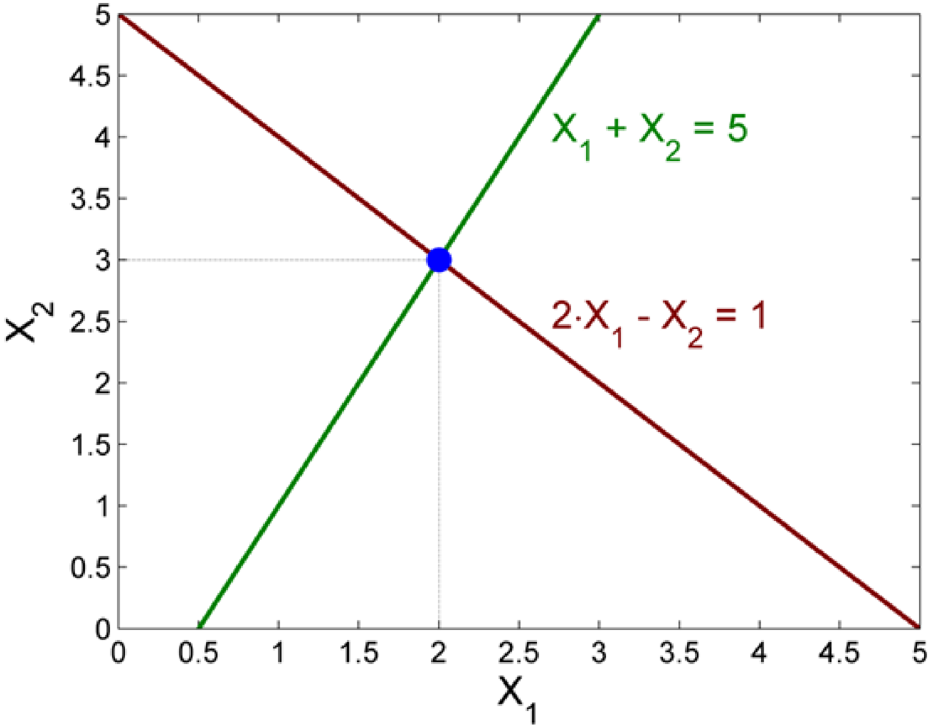
- *U zavisnosti od vašeg studijskog programa, asistenti će vas uputiti dalje.

Sadržaj predmeta

- Rešavanje sistema linearnih jednačina
- Rešavanje nelinearnih jednačina
- Interpolacija
- Aproksimacija
- Kompjuterska reprezentacija brojeva i greške
- Numerička integracija i diferenciranje
- Numeričko rešavanje diferencijalnih jednačina

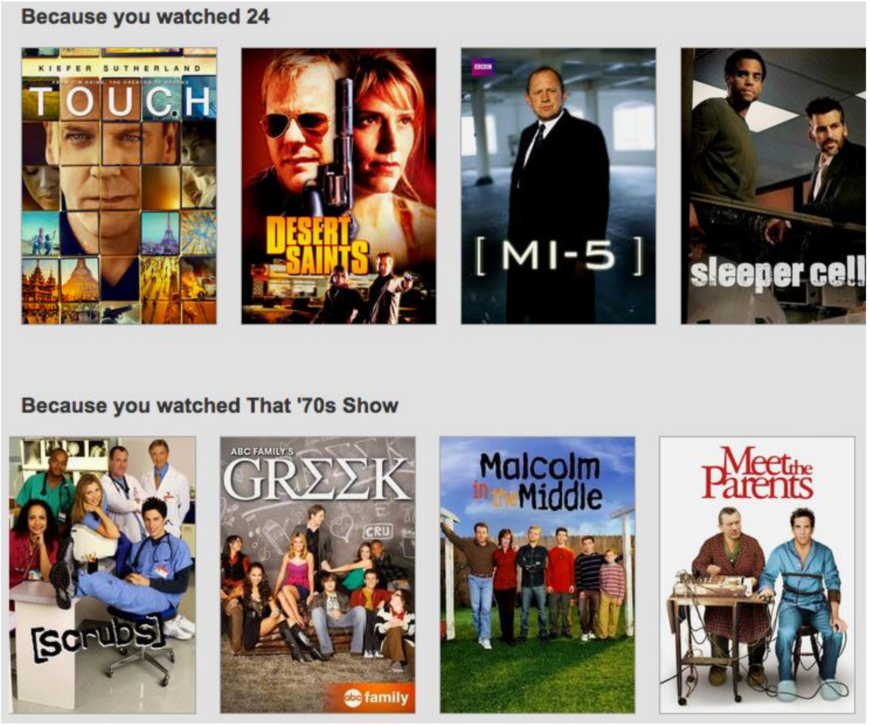
Rešavanje sistema linearnih jednačina

Numeričke metode, koje će vam omogućiti da rešite sisteme jednačina čije se rešnje ne može lako ili efikasno pronaći analitički

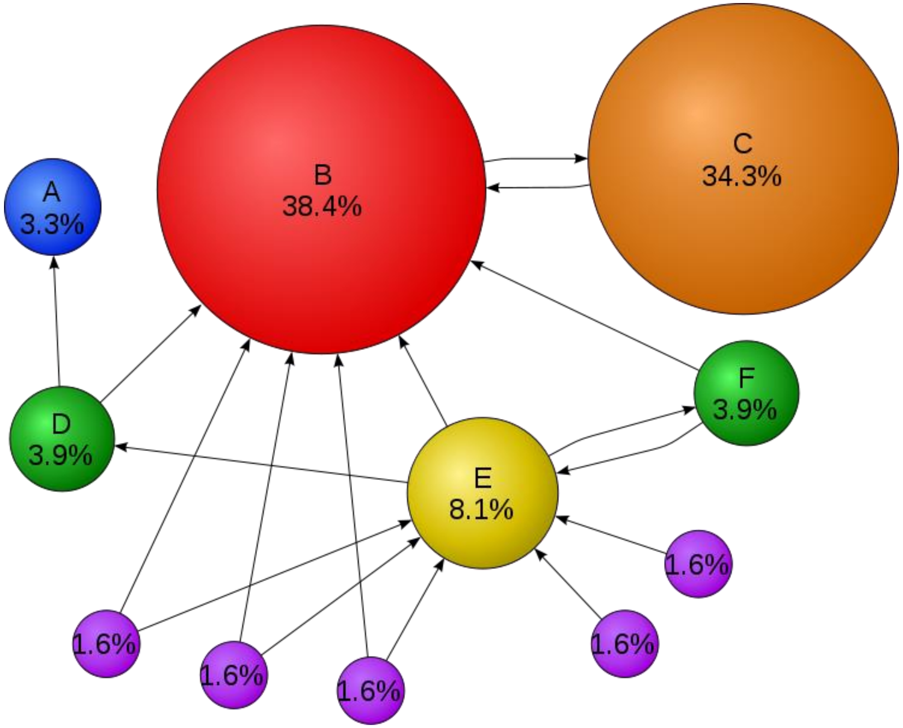


Rešavanje sistema linearnih jednačina

Kako nekome automatski preporučiti koji film da pogleda

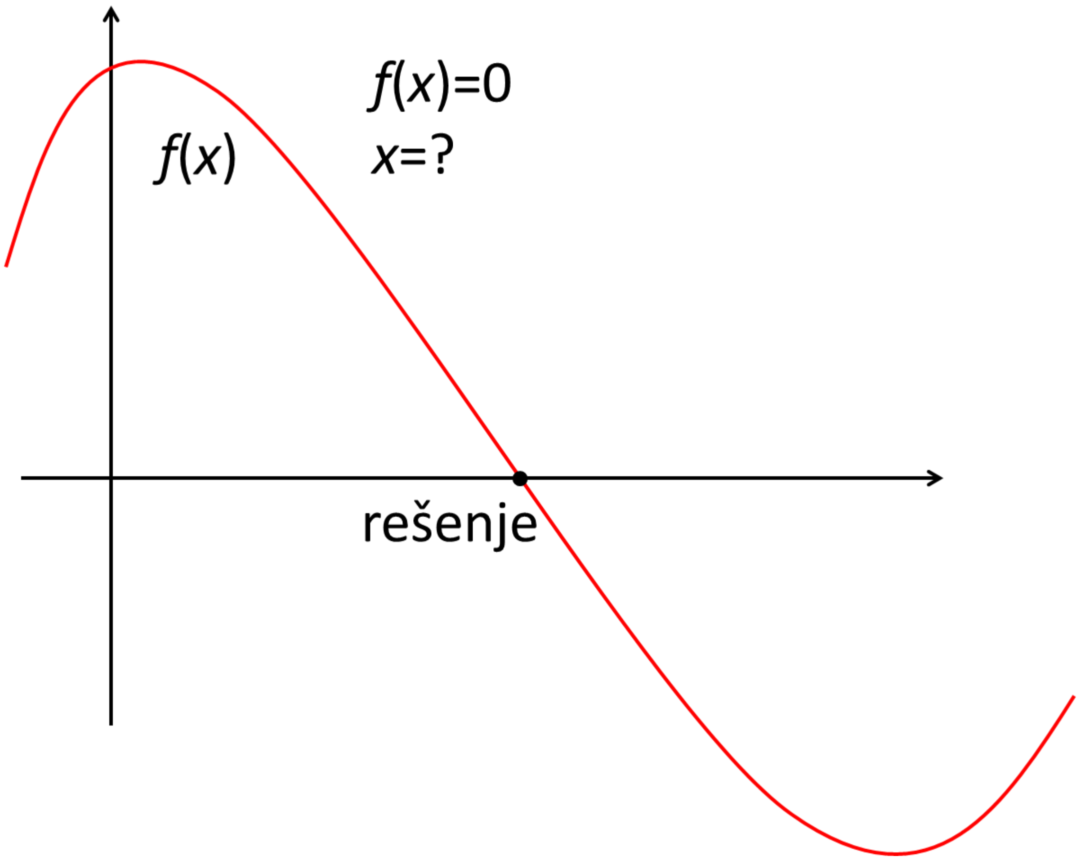


Google page rank



Rešavanje nelinearnih jednačina

Numeričke metode, koje će vam omogućiti da rešite nelineanu jednačinu čije se rešenje ne može lako ili efikasno pronaći analitički.

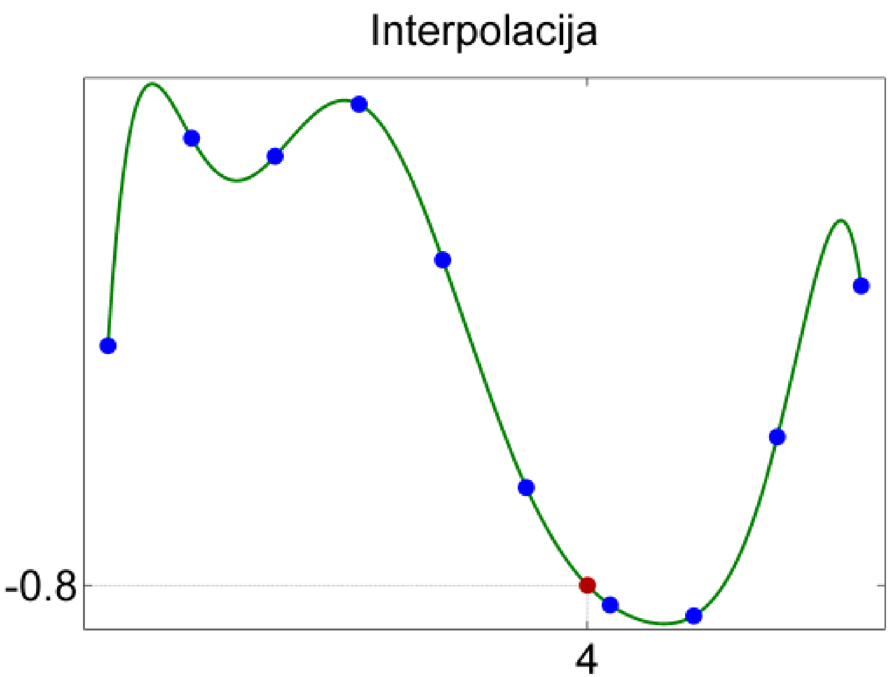


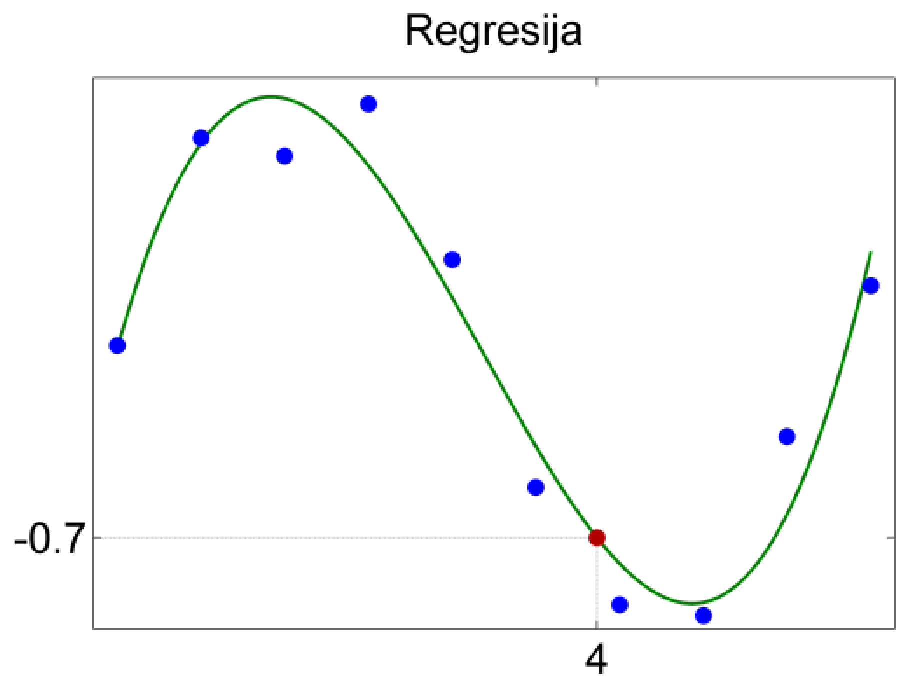
Rešavanje nelinearnih jednačina

Detekcija kolizija u
kompjuterskim igrama

Interpolacija i Aproksimacija

Kako da pomoću postojećih podataka (tačaka) kreirate matematički model i pomoću koga vršite predikcije ili dopunjujete nedostajuće vrednosti





Interpolacija i Aproksimacija

Kako da pomoću postojećih podataka (tačaka) kreirate matematički model i pomoću koga vršite predikcije ili dopunjujete nedostajuće vrednosti

Kako predvideti kurs evra za sledeću nedelju

Kako predvideti vrednost akcija *Apple* za sledeći mesec

Kako automatski predvideti koja će razlika biti u košarkaškoj utakmici

Kompjuterska reprezentacija brojeva i greške

- Kako izmeriti kvalitet numeričkog algoritma
- Koje su mogućnosti i ograničenja komjutera u radu sa brojevima (*float*, *double*, *overflow*, *underflow*...)
- Zašto je to važno?

Raketa *Ariane 5*

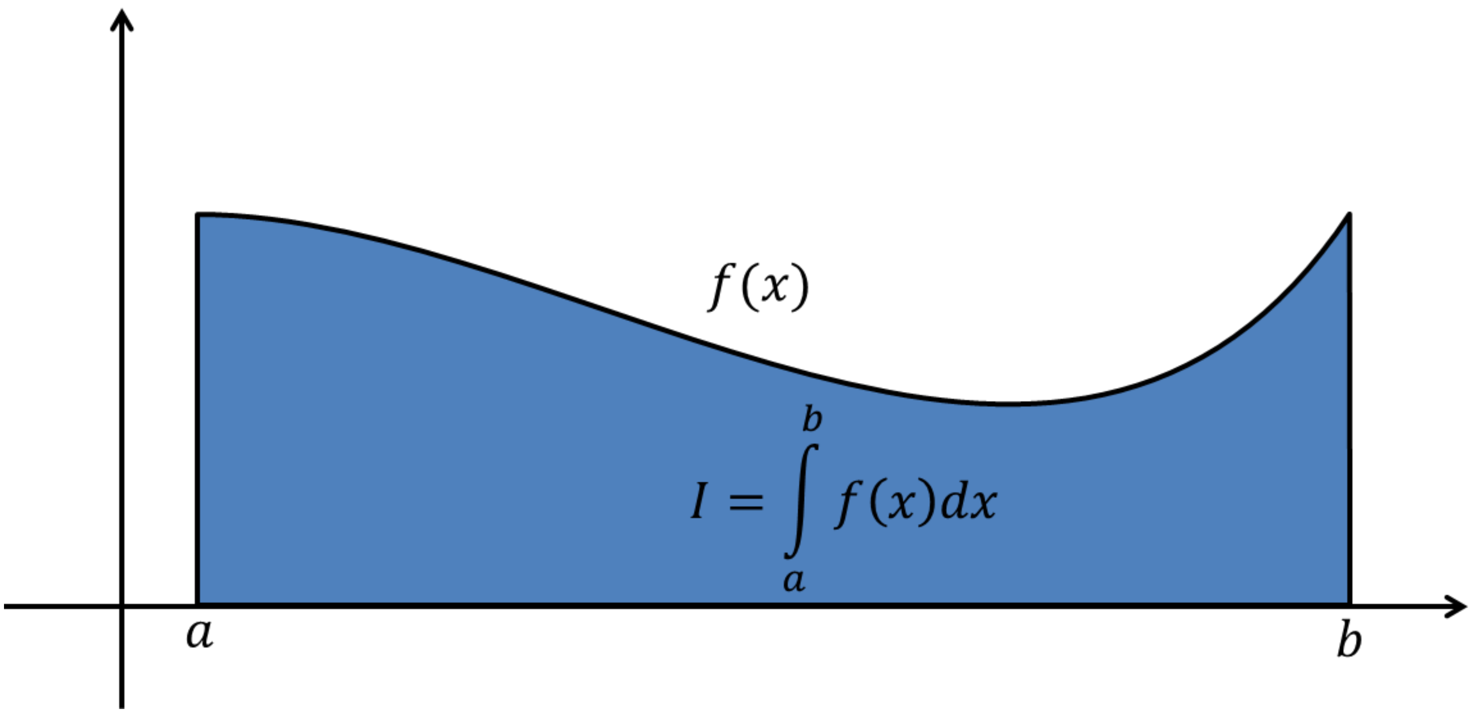
- 1996 eksplodirala samo 40s posle poletanja
- Razlog je greška u konverziji
- 64 bit floating point je konvertovan u 16 bit singed int
- došlo je do overflow-a

Irački SCUD projektil

- 1991 pogodio američku kasarnu, poginulo je 28 ljudi
- Američka patriot raketa nije uspela da ga presretne zbog
- greške u odsecanju binarne reprezentacije broja 1/10

Numerička integracija i diferenciranje

- Kako da lako rešavate integrale (i tražite izvode) čije se rešenje teško pronalazi analitičkim putem

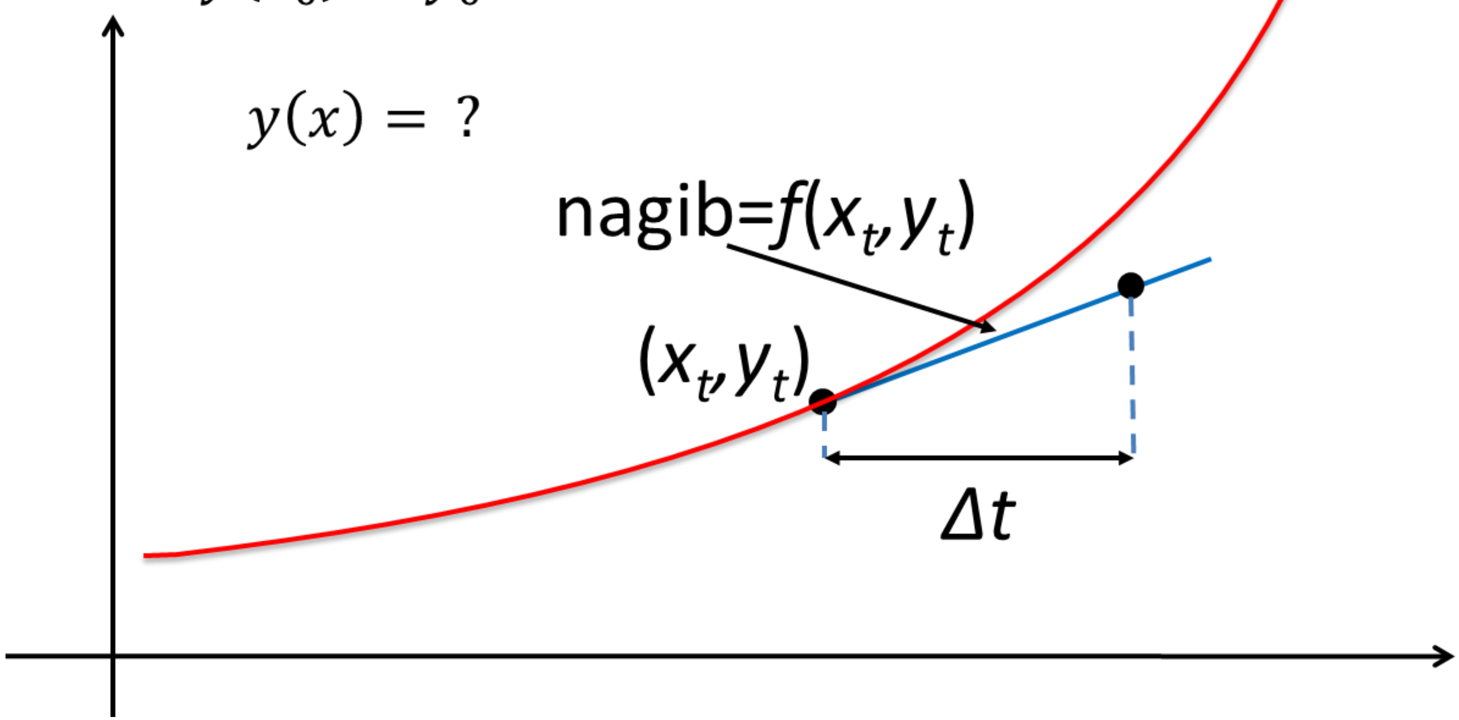


Numeričko rešavanje diferencijalnih jednačina

Kako da lako rešavate diferencijalne jednačine čije se rešenje teško pronalazi analitičkim putem

$$\frac{dy}{dx} = f(x, y)$$
$$y(x_0) = y_0$$

$$y(x) = \ ?$$



Numeričko rešavanje diferencijalnih jednačina

Fizika u kompjuterskim igrama

O predmetu

- Predavanja
  - objašnjenja numeričkih algoritama uz primere

- Vežbe
  - ideja je da sami implementirate i primenite metode koje učite na predavanjima
  - koristi se Python, Matlab ili Octave
  - svakoj celini iz teorije odgovara jedna vežba

Pravila polaganja

- Obaveze na predmetu sastoje se od dva dela:
  - Predispitne obaveze
  - Teorijski ispit

Pravila polaganja - Predispitne obaveze

- Sastoje se od dva kolokvijuma i projekta
  - kolokvijumi će biti održani samo jednom i to u toku semestra tj. pred početak ispitnog roka

- Termini za računarske kolokvijume su:
  - I kolokvijum, 11.12.2021, računarski centar FTN
  - II kolokvijum, 15.01.2022, računarski centar FTN

- Kolokvijumi počinju u 8:00, a o tačnom terminu za Vašu grupu bićete obavešteni na vreme.

Pravila polaganja - Predispitne obaveze

- Popravni kolokvijum
  - Tokom februara 2021. biće održan jedan popravni kolokovijum
  - Na popravnom kolokvijumu <u>može da se popravlja samo jedan od kolokvijuma, nikako oba</u>.
  - <u>Narednih prilika za poravni neće biti</u>.

Pravila polaganja - Projekat

- <u>Umesto</u> polaganja drugog računarskog kolokvijuma postoji mogućnost izrade projekta
- Projekat nosi više bodova od drugog kolokvijuma
- Detaljne informacije o projektima biće date na vežbama.
- Danas samo par generalnih informacija.

Pravila polaganja - Projekat

- Izrada projekta ima nekoliko faza:
  - Specifikacija predloga projekata
  - Izrada projekta
  - Odbrana projekta

Specifikacija i predloga projekata

- Studenti treba sami da osmisle temu projekta
- Okvirne informacije o temama dobićete kroz motivacione primere na predavanjima i vežbama i kroz materijale sa sajta predmeta
- Informacije o tome šta sve treba da sadrži specifikacija dobićete od asistenata
- Studenti čiji predlozi ne prođu u sledeću fazu moraju da rade drugi računarski kolokvijum

Izrada projekta

- Od momenta kada asistent odobri projekat studenti počinju sa izradom projekta
- Detalje oko načina na koji će asistenti pratiti projekte će vam oni saopštiti kad za to dođe vreme

Odbrana projekta

- Postoji samo jedan termin za odbranu projekata

- Tačan termin će objaviti asistenti i znaće se na vreme

- Način odbrane će takođe objaviti asistenti

Projekat – Važna napomena

- Kada student pristupi izradi projekta tj. kada je asistent odobrio projekat:
  - Student nema pravo polaganja drugog računarskog kolokvijuma
  - Dakle imate izbor, ali morate sami da donesete odluku šta želite da uradite

Pravila polaganja - Teorijski ispit

- Pismeno u terminima rasporeda ispita koje određuje FTN

- Sastoji se od 2 dela koji se mogu polagati odvojeno ili zajedno

Bodovanje – Teorijski deo

Bodovanje teorijskog dela ispita:

	min	max
Deo 1	50	100
Deo 2	50	100
Ukupno	100	200
Ukupno (preračunato)*	26	45

$$* \textit{Ukupno teorijski ispit} = \frac{(Deo1 + Deo2)}{200} * 38 + 7$$

Bodovanje – Predispitne obaveze

Bodovanje predispitnih obaveza:

	min	max	max sa projektom
Kolokvijum 1	14	27	27
Kolokvijum 2	11	23	0
Projekat	0	0	38
Ukupno	25	50	65

Bodovanje – Ukupno

Bodovanje teorijskog dela ispita:

	min	max
Deo 1	50	100
Deo 2	50	100
Ukupno	100	200
Ukupno (preračunato)*	26	45

Bodovanje predispitnih obaveza:

	min	max	max sa projektom
Kolokvijum 1	14	27	27
Kolokvijum 2	11	23	0
Projekat	0	0	38
Ukupno	25	50	65

Ukupno:

	min	max	max sa projektom
Ukupno teorija	26	45	45
Ukupno praktični	25	50	65
Ukupno	51	95	110

Bodovanje – Geodezija i Geoinformatika Predispitne obaveze

Bodovanje predispitnih obaveza:

	min	max
Kolokvijum 1	14	30
Kolokvijum 2	11	25
Ukupno	25	55

Bodovanje – Geodezija i Geoinformatika – Ukupno

Bodovanje teorijskog dela ispita:

	min	max
Deo 1	50	100
Deo 2	50	100
Ukupno	100	200
Ukupno (preračunato)*	26	45

Bodovanje predispitnih obaveza:

	min	max
Kolokvijum 1	14	30
Kolokvijum 2	11	25
Ukupno	25	55

Ukupno:

	min	max
Ukupno teorija	26	45
Ukupno praktični	25	55
Ukupno	51	100

Važna napomena

- Da biste imali pravo na zaključnu ocenu (položili premet), obavezno je da:

1. **Položite oba dela teorije**
   - Da osvojite minimalno 50 bodova iz svakog dela
2. **Položite prvi računarski kolokvijum**
3. **Položite drugi računarski kolokvijum ili odbranite projekat**

Literatura

- Teorijski deo
  - Slajdovi sa predavanja
  - Udžbenik:
  - „NUMERIČKE METODE U SOFTVERSKOM INŽENJERSTU“
  - Autori: Aleksandar Kovačević, Jelena Slivka
  - Izdavač: FTN, Novi Sad
- Predispitne obaveze
  - Materijali sa vežbi