Projekti iz Matematičnega modeliranja

Peter Kink

17. februar 2021

Kazalo

1 Prepoznavanje ročno zapisanih črk

Črke slovenske abecede A, B, C, Č, D, ..., Z, Ž predstavimo s sivinskimi slikami velikosti 16×16. V prvem – *učnem* naboru imamo slike znanih črk, v drugem – *testnem* naboru pa slike črk, ki jih želimo prepoznati.

Ideja prepoznavanja

Recimo, da testiramo, ali neka slika iz testnega nabora predstavlja črko K. Iz učnega nabora vzamemo vse slike, ki predstavljajo K in jih predstavimo z vektorjem dolžine 256 (vse stolpce slike zložimo po vrsti enega pod drugega). Te vektorje nato staknemo skupaj enega ob drugega v matriko A_{K}. Recimo, da je $\mathbf{b}$ vektor, ki predstavlja našo testno sliko.

Sedaj si ogledamo sistem $A_{\mathsf{K}}\mathbf{x} = \mathbf{b}$. V splošnem ta sistem nima rešitev, znamo pa poiskati rešitev z minimalno normo oz. najboljši 'približek rešitve' $\mathbf{x}_{\mathsf{K}} = A_{\mathsf{K}}^{+}\mathbf{b}$. Stvar ponovimo za vse črke iz učnega nabora, tj. poračunamo $\mathbf{x}_i = A_i^{+}\mathbf{b}$ za vse $i = \mathsf{A}, \ldots, \check{\mathsf{Z}}$. Nato izberemo tisti i, za katerega je $\|\mathbf{b} - A_i\mathbf{x}_i\|$ najmanjše. Če smo dobili $i = \mathsf{K}$, smo (najbrž) prepoznali črko K.

Implementacija

Sestavite primerno velik učni nabor velikih tiskanih črk (20 ali več). Tako direktna metoda, kot je opisana zgoraj, običajno ni dovolj učinkovita. Primerneje je, če vnaprej poračunamo singularne razcepe matrik A_i, $A_i = U_i S_i V_i^{\mathsf{T}}$, nato pa poiščemo rešitve sistemov $U_i S_i \mathbf{y}_i = \mathbf{b}$, kjer je $\mathbf{y}_i = V^{\mathsf{T}}\mathbf{x}_i$.

1. Utemeljite, da velja $\|\mathbf{x}_i\| = \|\mathbf{y}_i\|$ in $\|\mathbf{b} - A_i\mathbf{x}_i\| = \|U_i^{\mathsf{T}}\mathbf{b} - S_i\mathbf{y}_i\|$.

2. Namesto učnega nabora si torej zapomnimo le singularne vrednosti S_i in singularne vektorje U_i. Ali si moramo res zapomniti vse te vrednosti? Kako učinkovit/zanesljiv je postopek, če bi si zapomnili le prvih k singularnih vrednosti in singularnih vektorjev?

2 Prepoznavanje ročno zapisanih števk

Števke 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 predstavimo s sivinskimi slikami velikosti 16×16. V prvem – *učnem* naboru imamo slike znanih števk, v drugem – *testnem* naboru pa slike števk, ki jih želimo prepoznati.

Ideja prepoznavanja

Recimo, da testiramo, ali neka slika iz testnega nabora predstavlja števko 5. Iz učnega nabora vzamemo vse slike, ki predstavljajo 5 in jih predstavimo z vektorjem dolžine 256 (vse stolpce slike zložimo po vrsti enega pod drugega). Te vektorje nato staknemo skupaj enega ob drugega v matriko A_5. Recimo, da je $\mathbf{b}$ vektor, ki predstavlja našo testno sliko.

Sedaj si ogledamo sistem $A_5\mathbf{x} = \mathbf{b}$. V splošnem ta sistem nima rešitev, znamo pa poiskati rešitev z minimalno normo oz. najboljši 'približek rešitve' $\mathbf{x}_5 = A_5^+\mathbf{b}$. Stvar ponovimo za vse števke iz učnega nabora, tj. poračunamo $\mathbf{x}_i = A_i^+\mathbf{b}$ za vse $i = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$. Nato izberemo tisti i, za katerega je $\|\mathbf{b} - A_i\mathbf{x}_i\|$ najmanjše. Če smo dobili $i = 5$, smo (najbrž) prepoznali števko 5.

Implementacija

Sestavite primerno velik učni nabor števk (20 ali več). Tako direktna metoda, kot je opisana zgoraj, običajno ni dovolj učinkovita. Primerneje je, če vnaprej poračunamo singularne razcepe matrik A_i, $A_i = U_iS_iV_i^\mathsf{T}$, nato pa poiščemo rešitve sistemov $U_iS_i\mathbf{y}_i = \mathbf{b}$, kjer je $\mathbf{y}_i = V^\mathsf{T}\mathbf{x}_i$.

1. Utemeljite, da velja $\|\mathbf{x}_i\| = \|\mathbf{y}_i\|$ in $\|\mathbf{b} - A_i\mathbf{x}_i\| = \|U_i^\mathsf{T}\mathbf{b} - S_i\mathbf{y}_i\|$.

2. Namesto učnega nabora si torej zapomnimo le singularne vrednosti S_i in singularne vektorje U_i. Ali si moramo res zapomniti vse te vrednosti? Kako učinkovit/zanesljiv je postopek, če bi si zapomnili le prvih k singularnih vrednosti in singularnih vektorjev?

3 Prepoznavanje uporabnika tipkovnice

Različni ljudje razvijemo različne stile tipkanja. Naloga je, napišete program, ki bo na podlagi količin, ki jih lahko izmerimo pri tipkanju (kot je časovni zamik med posameznimi pari znakov), prepoznal uporabnika.

- Definirajte in sestavite učno množico. To lahko dobite npr. tako, da različnim uporabnikom naložite tipkanje določenega besedila in pri tem merite časovne zamike med pritiski na tipke za vse možne pare znakov (črke, presledki, vejice itd). Na ta način dobite recimo n krat 25×25 (oziroma več, če upoštevamo ostale znake) veliko matriko podatkov za vsakega uporabnika.

- Za vsakega uporabnika poskusite analizirati dobljene podatke s pomočjo SVD razcepa ali PCA ali kakšno drugo metodo, ki bi lahko zajela (upamo) bistvene značilnosti uporabnikovega tipkanja.

- Premislite, kako lahko analizo podatkov, ki naj bi bila značilna za uporabnika (npr. največji singularni vektorji ali glavne smeri), uporabite za primerjavo s podatki, ki jih dobite, ko neznan uporabnik tipka poljubno besedilo.

Naloga sicer ni tehnično zahtevna, bo pa potrebno precej preiskušanja različnih zamisli, uspeh pa ni zagotovljen. Tako bomo ob morebitnem pomanjkljivem končnem rezultatu ocenjevali predvsem znanje, trud in iznajdljivost.

4 Premiki togega telesa

Premik togega telesa je sestavljen iz rotacije in paralelnega premika. Koordinate x točk na telesu se preslikajo v nove koordinate $y = Ax + b$, pri cemer matrika A opisuje rotacijo telesa, vektor b pa premik težišča.

Naša naloga je poiskati matriko A in vektor b, če poznamo koordinate $x_1, \ldots, x_n$ nekaj značilnih točk telesa pred premikom in koordinate $y_1, \ldots, y_n$ istih točk po premiku. Pravzaprav je to zelo znan problem, ki se pogosto rešuje na primer v robotiki.

Približno rekonstrukcijo vektorja translacije b dobimo tako, da izračunamo premik težišča izbranih točk. Točke x_i in y_i najprej prestavimo tako, da imajo težišče v koordinatnem izhodišču:

$$x'_i = x_i - \bar{x}, \quad y_i = y_i - \bar{y}, \quad \text{kjer je} \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{in} \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i,$$

in nato izračunamo razliko $b = \bar{y} - \bar{x}$.

Matrika rotacije A je ortogonalna matrika, ki zadošča pogoju $AX = Y$. Pri tem sta X in Y matriki dimenzije $3 \times n$, X ima v stolpcih (premaknjene) točke x'_i, Y pa točke y'_i. Zaradi napak pri meritvah enakost ne bo natančno izpolnjena, zato bomo poiskali matriko A tako, da bodo odstopanja, ki so zajeta v matriki $AX - Y$ najmanjša možna.

Problem lahko rešimo s pomočjo SVD razcepa z naslednjim algoritmom:

- Izračunamo *kovariančno matriko* $C = YX^T$, ki ima dimenzijo 3×3.
- Poiščemo SVD razcep $C = USV^T$, kjer je S diagonalna matrika s singularnimi vrednostmi matrike C.
- Matriko S nadomestimo z diagonalno matriko

  $$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & d \end{bmatrix},$$

  kjer je $d = \pm 1$ predznak determinante $\det(C)$.
- Iskana matrika A je enaka $A = UDV^T$.

Vaša naloga je implementirati opisani algoritem in ga preizkusiti na nekaj konkretnih primerih:

1. na umetnih podatkih, ki jih zgenerirate sami,
2. na realnih, ki jih nekako pridobite sami; lahko na primer z GPS napravo izmerite koordinate nekaj značilnih točk na vašem avtomobilu ali kolesu, potem pa ga prestavite kam drugam in ponovno izmerite koordinate istih točk (uspeh poskusa bo odvisen od natančnosti vaše GPS naprave).

5 Analiza slik

Cilj projekta je prikazati preprost primer uporabe Metode glavnih komponent (PCA) pri analizi slik.

- S fotoaparatom naredite nekaj (recimo 20) slik svojega najljubšega predmeta ali scene ob različnih časih $t_1, \ldots, t_{20}$ ali iz različnih zornik kotov. Pazite na to, da se razmere sicer spremenijo, vendar ne preveč, zajeti želite podobne slike. Dobljene digitalne slike pretvorite v črnobele slike v bitmap formatu, tj. v matrike z vrednostmi med 0 (črna) in 255 (bela). Poskrbite za to, da bo velikost vseh slik enaka.

- Vsako sliko predstavite kot vektor dimenzije N, kjer je N število slikovnih elementov (pixlov) na sliki. Če je resolucija slike $n \times m$, potem ima vektor dimenzijo nm, kjer prvih m komponent ustreza prvi vrstici na sliki, drugih m drugi vrstici in tako dalje. Dobljene vektorje $v_1, \ldots, v_{20}$ centrirajte in normailizirajte:

  od vsakega odštejte povprečje μ in delite z varianco V, kjer je

  $$\mu = \frac{1}{20}\sum_{i=1}^{20} v_i \quad \text{in} \quad V = \frac{1}{20}\sum_{i=1}^{20}(v_i - \mu)^2.$$

  Dobljeni nabor 20 vektorjev $w_1, \ldots, w_{20}$ je vzorec slik vašega objekta ali scene s povprečjem 0 in varianco 1.

- Sestavite matriko X, ki ima v vrsticah vektorje $w_1, \ldots, w_{20}$ in poiščite prvi dve lastni smeri e_1, e_2, tj. lastna vektorja, ki pripadata največjima dvema lastnima vrednostma kovariančne matrike $A = XX^T$. Uporabite potenčno metodo za hkratno iskanje prvih dveh glavnih smeri:

  Poiščite dva poljubna ortogonalna vektorja in ju postavite kot stolpca v matriko C. Zaporedje potenc $C, AC, A^2C, \ldots$ konvergrira proti matriki, ki ima v stolpcih iskana dva lastna vektorja. Metoda bo konvergirala hitreje, če začnete s prvima dvema stopcema matrike X, ki ju ortogonalizirate po Gramm Schmidtovem postopku.

- S pomočjo SVD razcepa matrike X poiščite ortogonalne projekcije $Pw_1, \ldots, Pw_{20}$ vektorjev $w_1, \ldots, w_{20}$ na podprostor, razpet na e_1 in e_2, tako da v diagonalni matriki Σ vse lastne vrednosti razen prvih dveh nadomestite z 0. Narišite digitalne slike, ki jih predstavljajo vektorji $Pw_1, \ldots Pw_{20}$ (ne pozabite vrednosti nazaj pretvoriti na interval $[0, 255]$ in zaokrožiti na cela števila). To so aproksimacije vaših začetnih slik – verjetno bolj slabe, ker ste uporabili samo dve glavni smeri. Poskusite še s tremi ali več glavnimi smermi.

6 Hitri/odrezani SVD razcep

Ena od nerodnosti pri direktni uporabi SVD za kompresijo slik je časovna zahtevnost izračuna vseh singularnih vrednosti. Če se vnaprej odločimo, da bomo poiskali le prvih k singularnih vrednosti, lahko prihranimo kar nekaj časa pri izračunu.

Odrezani SVD

Pri odrezanem singularnem razcepu izračunamo le prvih k največjih singularnih vrednosti matrike A (in prav tako le prvih k stolpcev matrik U in V).

Napišite funkcijo/rutino, ki poišče le prvih k singularnih vrednosti matrike A in pripadajoča ortogonalna nabora prvih k singularnih vektorjev. Poskusite recimo s poenostavljeno Jacobijevo iteracijo za iskanje lastnih vrednosti matrike

$$\begin{bmatrix} 0 & A^\mathsf{T} \\ A & 0 \end{bmatrix}.$$

Malce bolj specifično:

1. Poiščite natančno zvezo med lastnimi vrednostmi in lastnimi vektorji matrike $\left[\begin{smallmatrix} 0 & A^\mathsf{T} \\ A & 0 \end{smallmatrix}\right]$ ter singularnimi vrednostmi in singularnimi vektorji matrik A in A^T.

2. Opišite postopek, ki zanesljivo poišče prvih k največjih singularnih vrednosti matrike A ter pripadajoče singularne vektorje.

3. Implementirajte ta postopek.

Testiranje in uporaba/uporabnost

Preverite, na poljubnem naboru matrik, hitrost in učinkovitost take metode. Primerjajte rezultate z že dostopnimi metodami (recimo `svd` iz `octave`-a).

7 Filter zamegljenosti

Bitna slika ja lahko zamegljena zaradi inherentnih napak naprave pri zajemu slike (npr. rentgenske slike). Te napake so prisotne v vsaki zajeti sliki in so vedno enako zastopane.

Zameglitev slike

Bitno sliko predstavimo z $m \times n$ matriko A, v kateri ima vsak element vrednost med (recimo) 0 in 255. Zameglitev v navpični smeri lahko predstavimo kot množenje z matriko

$$T = \begin{bmatrix} t_1 & t_2 & \cdots & t_k & 0 & 0 & \cdots & 0 \\ 0 & t_1 & t_2 & \cdots & t_k & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & & \ddots & & 0 \\ 0 & \cdots & 0 & 0 & t_1 & t_2 & \cdots & t_k \end{bmatrix}$$

velikosti $(m - k + 1) \times m$, kjer velja $t_1 + t_2 + \cdots + t_k = 1$.

V idealiziranem primeru nam torej naprava/skener vrne sliko TA, poleg tega pa vnaprej poznamo matriko T. Če ima matrika T inverz, lahko A izrazimo kot $T^{-1}(TA)$.

1. Kdaj ima T inverz?

2. Kaj storiti, če T nima inverza?

Filter zamegljenosti in rekonstrukcija A

Dober približek za A oziroma dobro izboljšavo za zamegljeno sliko TA dobimo, če poračunamo $G(TA)$, kjer je G posplošen inverz za T.

1. Testirajte smiselnost postopka. Izberite si sliko A, konstruirajte matriko T (za poljubne $t_1, \ldots, t_k$) in zamegljeno sliko TA, nato pa primerjajte A in GTA. (Za G lahko izberete kar T^+.) Ali je rezultat odvisen od tega, kateri posplošen inverz izberete?

2. Napišite *učinkovit* algoritem za izračun $G(TA)$. Zavedajte se, da nas na koncu zanima le GTA ne pa tudi posplošen inverz G.

8 Bločna SVD kompresija

Uporaba SVD za kompresijo kompleksne slike po eni strani ni primerna, ker si moramo zapomniti precej singularnih vrednosti, če želimo ohraniti podrobnosti slike. (Kompleksne slike imajo precej bolj razpršene singularne vrednosti kot enostavne slike.) Primerneje bi bilo sliko razrezati na manjše kose – *bloke*, nato pa stisniti sliko z uporabo singularnega razcepa na posameznih blokih.

Bločni SVD

Izbrane slike razrežite na bloke različnih velikosti $k \times k$ (recimo 8×8, 16×16, 32×32, 64×64) in primerjajte kvaliteto bločne SVD kompresije, če shranite od 1 do $k/2$ največjih singularnih vrednosti pri vsakem bloku.

Bločni SVD z adaptivno izbiro ranga

Če se vnaprej odločimo za določeno stopnjo kakovosti stisnjene slike $q \in [0, 1]$, lahko rang na posameznih blokih prilagajamo. Na blokih, ki nimajo veliko detajlov si zapomnimo le malo največjih singularnih vrednosti (manjši rang), na blokih z veliko detajli pa več singularnih vrednosti (večji rang).

Enostaven način prilagajanja ranga je naslednji. Zapomnimo si le tiste singularne vrednosti $\sigma_1, \ldots, \sigma_r$ iz nabora $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$, da je

$$q \doteq \frac{\sigma_1 + \sigma_2 + \cdots + \sigma_r}{\sigma_1 + \sigma_2 + \cdots + \sigma_k}.$$

Preverite stopnjo kakovosti in kompresije svojega nabora slik pri različnih q in različnih velikostih blokov $k \times k$. Kako bi shranili tako stisnjeno sliko? Kaj bi še lahko storili za dodatno izboljšavo razmerja med kakovostjo in kompresijo?

9 Povzetek dokumenta

Povzetek

Naloga je izdelati program, ki za dani dokument poišce primeren povzetek. Uporabite metodo *latentnega semanticnega indeksiranja* (LSI) in med povedmni v besedilu poišcete tiste, ki najbolje predstavljajo celoten dokument.

Izdelajte program, ki bo v danem dokumentu poiskal povedi, ki najvec povedo o temi dokumenta. Nalogo rešite v vec korakih. Glejte tudi [Steinberger].

1. Iz dokumenta zgradite matriko A, ki povezuje besede in povedi v dokumentu. Vsaka poved naj ima v matriki svoj stolpec, vsaka beseda pa svojo vrstico. Element a_{ij} naj bo frekvenca i-te besede v j-ti povedi.

2. Matriko A razcepite z odrezanim SVD razcepom $A = U_k S_k V_k^T$, ki obdrži le k najvecjih singularnih vrednosti. Razmislite kaj predstavljajo stolpci matrike U_k in matrike V_k. Odrezan SVD zmanjša t. i. "overfitting" (preveliko prilagoditev modela podatkom, kar povzroci povecan vpliv šuma).

3. Za vsako singularno vrednost iz S izberite poved, ki ima najvecjo ustrezno komponento. Povzetek sestavite iz tako izbranih povedi za nekaj najvecjih singularnih vrednosti.

4. Stavke za povzetek lahko izberete tudi na podlagi celotne "utežene dolžine", ki upošteva tudi singularne vrednosti s_i:

   $$\|x\|_s = \sqrt{(x_1 s_1)^2 + (x_2 s_2)^2 + \ldots + (x_k s_k)^2}.$$

   Primerjajte povzetek, ki ga dobite na ta nacin s povzetkom iz prejšnje tocke.

5. Metodo je mogoce izboljšati, ce frekvence v matriki A nadomestimo z bolj kompleksnimi merami. V splošnem lahko element matrike zapišemo kot produkt

   $$a_{ij} = L_{ij} \cdot G_i,$$

   kjer je L_{ij} lokalna mera za pomembnost besede v posamezni povedi, G_i pa globalna mera pomembnosti posamezne besede. Preiskusite shemo, pri kateri je lokalna mera dana z logaritmom frekvence f_{ij} i-te besede v j-ti povedi:

   $$L_{ij} = \log(f_{ij} + 1).$$

Globalna mera pa je izracunana s pomocjo entropije

$$G_i = 1 - \sum_j \frac{p_{ij} \log(p_{ij})}{\log n},$$

kjer je n število povedi v dokumentu,

$$p_{ij} = \frac{f_{ij}}{gf_i}$$

in gf_i frekvenca besede v celotnem dokumentu. Podrobnosti so v [Dumais]. Preverite ali zgoraj opisana mera izboljša kvaliteto abstrakta.

Literatura

[Steinberger] Josef Steinberger and Karel Jezek. Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM'04*, pages 93-100, 2004.

[Dumais] Susan T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23(2):229- 236, 1991.

10 Iskanje po zbirki dokumentov

Povzetek

Izdelajte iskalnik relevantnih dokumentov po kljucnih besedah z metodo *latentnega semanticnega indeksiranja* (LSI). Metode, ki izberejo le dokumente, ki vsebujejo natanko iskane besede, so precej nenatancne. Ljudje namrec uporabljamo veliko sopomenk, ki jih preproste metode ne povežejo. Metoda LSI zgradi model, ki združuje vec besed v pojme in zato najde tudi dokumente, ki so relevantni, pa ne vsebujejo iskalne besede.

Izdelajte program, ki bo v dani zbirki za dane kljucne besede poiskal najbolj relevantne dokumente. Nalogo rešite v vec korakih. Glejte tudi [Berry].

1. Iz zbirke dokumentov zgradite matriko A povezav med besedami in dokumenti. Vsak dokument naj ima v matriki svojo stolpec, vsaka beseda pa svojo vrstico. Element a_{ij} naj bo frekvenca i-te besede v j-tem dokumentu.

2. Matriko A razcepite z odrezanim SVD razcepom $A = U_k S_k V_k^T$, ki obdrži le k najvecjih singularnih vrednosti. Razmislite kaj predstavljajo stolpci matrike U_k in matrike V_k. Odrezan SVD zmanjša t. i. "overfitting" (preveliko prilagoditev modela podatkom, kar povzroci povecan vpliv šuma).

3. Iskani niz besed (poizvedbo) zapišite z vektorjem q. Iz poizvedbe q generirajte vektor v prostoru dokumentov s formulo

$$\hat{q} = q^T U_k S_k^{-1}$$

   Iskanim dokumentom ustrezajo stolpci V_k, ki so dovolj blizu vektorju $\hat{q}$. Za razdaljo uporabite kosinus kota med dvema vektorjema in ne Evklidske razdalje med njima. Poizvedba naj vrne dokumente, za katere je kosinus vecji od izbrane mejne vrednosti. Preskusite razlicne mejne vrednosti kosinusa, pri kateri izberemo dokument (0.9, 0.7, 0.6, ...).

4. Metodo je mogoce izboljšati, ce frekvence v matriki A nadomestimo z bolj kompleksnimi merami. V splošnem lahko element matrike zapišemo kot produkt

$$a_{ij} = L_{ij} \cdot G_i,$$

kjer je L_{ij} lokalna mera za pomembnost besede v posameznem dokumentu, G_i pa globalna mera pomembnosti posamezne besede. Preiskusite shemo, pri kateri je lokalna mera dana z logaritmom frekvence f_{ij} i-te besede v j-tem dokumentu:

$$L_{ij} = \log(f_{ij} + 1).$$

Globalna mera pa je izracunana s pomocjo entropije

$$G_i = 1 - \sum_j \frac{p_{ij} \log(p_{ij})}{\log n},$$

kjer je n število dokumentov v zbirki,

$$p_{ij} = \frac{f_{ij}}{gf_i}$$

in gf_i frekvenca besede v celotni zbirki. Glejte tudi [Dumais2].

5. *Dodajanje dokumentov in besed.* Razmislite, kako bi v model dodali nove dokument ali besede, ne da bi bilo treba ponovno izracunati SVD razcep matrike A.

Program preiskusite na podatkih, ki jih dobite npr. na Classic SMART datasets. Lahko pa zudi sami zgradite zbirko iz prosto dostopnih dokumentov.

Literatura

[Berry] M. W. Berry, S.T. Dumais, G.W. O'Brien, Michael W. Berry, Susan T. Dumais, and Gavin. Using linear algebra for intelligent information retrieval. SIAM Review, 37:573-595, 1995.

[Dumais2] Susan T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23(2):229- 236, 1991.

11 Trilateracija

Radi bi določili lokacijo objekta v ravnini ali prostoru, na voljo pa imamo le nenatančne merilnike razdalj, s katerimi lahko ocenimo razdaljo od znanih položajev do objekta.

Za pričakovati je, da bomo položaj lahko bolj natančno določili, če opravimo veliko meritev. Potem lahko predpostavimo, da imamo n znanih položajev $\mathbf{x}_i = (x_i, y_i)$ (ali (x_i, y_i, z_i)), $i = 1, \ldots, n$ in izmerjene razdalje r_i do neznane neznane točke $\mathbf{x} = (x, y)$ (ali (x, y, z)), kjer je n večji od teoretično zadostnega števila meritev, ki bi jih morali opraviti, če bi imeli natančne merilnike.

1. Zapišite enačbe za neznane koordinate $\mathbf{x}$, ki jih narekujejo izmerjene razdalje.

Sistem enačb, ki ga dobite, je predoločen in je treba iskati rešitev v smislu najmanjših kvadratov. Žal pa so enačbe tudi kvadratne v neznankah, kar pomeni, da gre za nelinearen problem najmanjših kvadratov. To težavo lahko poizkusite rešiti na več načinov.

(a) V vseh enačbah nastopajo kvadrati neznank povsod s faktorjem 1. Z odštevanjem enačb lahko potem (na več načinov) sistem predelate v predoločen sistem linearnih enačb, ki ga znate rešiti. Premislite in preiskusite, kako dobro se rešitev takega sistem ujema z rešitvijo za originalen sistem.

(b) Sistem lahko rešite s kakšno metodo za reševanje nelinearnega problem njamanjših kvadratov kot je Gauss-Newtonova metoda (ampak samo če se ne bojite parcialnih odvodov).

Vaše rešitve lahko preiskusite na simuliranih podatkih, zaželeno pa je, da bi jih preiskusili tudi na realnih meritvah, ki jih dobite s napravami po lastni izbiri.

12 Vojna barv

Imamo populacijo $n \times n$ točk razporejenih v kvadratno mrežo v ravnini, ki lahko izbirajo med m barvami (mislimo si kvadrat pikslov na ekranu).

Na vsakem koraku vsaka točka izbere (ne nujno) novo barvo b glede na delež točk v svoji okolici, ki so barve b.

Naloga: Napišite **učinkovit** program, ki simulira in izrisuje opisan (markovski) proces.

Če bi morali napisati matriko prehodnih stanj takšne markovske verige, kako velika bi bila?

Dokazano je, da prej ali slej ena barva prevlada, torej da ima ta markovska veriga m absorbirajočih stanj. Eksperimentalno preverite trditev, da je verjetnost, da določena barva zmaga, enaka trenutnemu deležu točk, ki so te barve.

Da potek procesa ne bo dolgočasen, ga lahko prilagodite recimo tako, da točka z majhno verjetnostjo spontano spremeni barvo v primeru, da je vsa okolica iste barve.

Opomba : Ker je naloga bolj programerska in matematično ni zahtevna, se bo cenila učinkovitost in hitrost izvedbe. Problem je zelo primeren za paralelizacijo, tako da bi bilo zaželjeno, da se naloge loti skupina, ki zna paralelno programirati, po možnosti na GPU. Nikakor pa program ne sme biti napisan v octaveu.

13 Spam generator

S pomočjo simulacije markovske verige lahko napišemo preprost generator teksta, ki v grobem oponaša dano besedilo oz. jezik.

Analiza besedila: Vzemite besedilo (ali več besedil) in

1. Identificirajte vse besede, ki se pojavijo in jih oštevilčite oziroma shranite v seznam *besede*.

2. Za vsak i preštejte, kolikrat se neposredno za besedo $besede(i)$ v besedilu pojavi beseda $besede(j)$, vključno z ločili. Rezultat shranite v matriko $frekvence$ na (i, j)-tem mestu.

Generiranje besedila: Normirajte vrstice matrike $frekvence$, tako da bo vsota vsake vrstice enaka 1. Dobljena matrika predstavlja matriko prehodnih verjetnosti za markovsko verigo, s katero lahko generirate zaporedja besed.

14 Generiranje zaporedja akordov

S pomočjo simulacije markovske verige lahko napišemo preprost program za generiranje glasbe kot zaporedja akordov.

Analiza tablatur: Vzemite več glasbenih tablatur, recimo za kitaro.

1. Identificirajte vse akorde, ki se pojavijo in jih oštevilčite oziroma shranite v seznam *akordi*. Pri tem pazite na dolžino trajanja posameznega akorda.

2. Za vsak i preštejte, kolikrat se neposredno za akordom $akordi(i)$ v tablaturah pojavi akord $akordi(j)$. Rezultat shranite v (i, j)-to komponento matrike $frekvence$.

Generiranje novega zaporedja akordov: Normirajte vrstice matrike $frekvence$, tako da bo vsota vsake vrstice enaka 1. Dobljena matrika predstavlja matriko prehodnih verjetnosti za markovsko verigo, s katero lahko generirate nove pesmi, torej zaporedja akordov.

15 Brownovo gibanje

Brownovo gibanje je najpomembnejši primer markovskega procesa, ki poteka v zveznem času in ima za prostor stanj $\mathbb{R}^d$. V natančno definicijo se ne bomo spuščali, lahko pa približke za poti Brownovega gibanja zelo enostavno simuliramo:

Z $W(t)$ označimo stanje ob času t (torej $W(t) \in \mathbb{R}^d$). Potem lahko dobimo stanje ob času $W(t+h)$ z

$$W(t+h) = W(t) + X,$$

kjer $X \sim N(0, h)$ normalno porazdeljena slučajna spremenljivka z varianco h, oziroma vektor neodvisnih normalnih spremenljivk z isto varianco h.

1.del Napišite funkcije, ki simulirajo in narišejo potek Brownovega gibanja v dani dimenziji in za dano začetno stanje in korak h. Nekaj slik primerov poti za $d = 1, 2, 3$ vključite tudi v poročilo.

Potem boste obravnavali nekaj vprašanj v zvezi z Brownovim gibanjem, za katere poznamo teoretične odgovore, ki pa jih boste primerjali z rezultati simulacij.

1. **Izstopna porazdelitev iz množice:** Recimo, da začnemo Brownovo gibanje v neki začetni točki (x, y), ki leži znotraj nekega pravokotnika v ravnini. Brownovo gibanje gotovo prej ali slej uide iz pravokotnika, vprašanje pa je, kje. S pomočjo simulacije za dano začetno točko in pravkotnik ocenite verjetnost, da Brownovo gibanje uide iz pravokotnika na določeni stranici.

   Verjetnost takega dogodka je možno teoretično izračunati, če se rešijo ustrezne enačbe. Natančnejša navodila za to dobite na govorilnih urah oz. vajah.

2. **Minljivost in povrnljivost:** Začnimo Brownovo gibanje W_t v točki $x \in \mathbb{R}^d$, za katero velja $r < \|x\| < R$ za neki razdalji r in R.

   Z S_r in S_R označimo prvi čas, ko se W_t približa na razdaljo r od izhodišča oz. oddalji na razdaljo R. S simulacijo ocenite verjetnost

   $$\mathbb{P}(S_r < S_R)$$

   pri dani začetni točki x za dimenzije $d = 1, 2, 3$.

   To verjetnost je možno izračunati teoretično:

- Za $d = 1$
$$\mathbb{P}(S_r < S_R) = \frac{R - x}{R - r}$$
- Za $d = 2$
$$\mathbb{P}(S_r < S_R) = \frac{\log(R) - \log(\|x\|)}{\log(R) - \log(r)}$$
- Za $d \geq 3$
$$\mathbb{P}(S_r < S_R) = \frac{R^{2-d} - \|x\|^{2-d}}{R^{2-d} - r^{2-d}}$$

Izračunajte limite zgornjih verjetnosti, ko $R \to \infty$. Kaj to pove o minljivosti in povrnljivosti za Brownovo gibanje v različnih dimenzijah?

3. **Izhodni čas iz dane množice:** Začnete enako kot pri prvi nalogi, le da tokrat merite povprečen čas, ki je potreben, da W_t doseže rob pravokotnika.

   Tudi tu je možno ta povprečen čas glede na začetno točko teoretično izračunati, če se rešijo prave enačbe, ki pa jih bomo naknadno razložili.

16 Štiri v vrsto

Radi bi napisali umetno inteligenco za igranje igre 4 v vrsto s kombinacijo markovskih verig in evolucijskega algoritma.

Ideja

Ideja je, da taktiko igralca opišemo z markovsko verigo in pustimo, da računalnik igra sam proti sebi. Na začetku so markovske verige takšne, da kroglice spuščamo bolj ali manj enakomerno naključno, nato pa porazdelitve markovskih verig spreminjamo in v naslednjo generacijo vzamemo takšne, ki se bolj obnesejo.

Problemi

Težava je, da če za stanje markovske verige vzamemo vsako možno postavitev kroglic v recimo 5×7 veliki tabeli, je vseh možnih stanj (vključno s končanimi igrami) 3^{35} in bi tako imeli $3^{35} \times 3^{35}$ veliko markovsko matriko.

Nujno je, da število stanj, ki jih upoštevamo, zmanjšamo z nekaj klasične umetne inteligence in uporabo raznih hevristik za opis stanja (kot je število kroglic v stolpcih, število nizov dolžine 2 in 3 ene in druge barve ali kaj podobnega).

Druga posebnost take markovske verige je, da so vsa stanja prehodna (kroglic ne moremo jemati nazaj) in da imamo na vsakem koraku kvečjemu (recimo) 7 možnosti, se pravi vsaka vrstica prehodne matrike ima največ 7 neničelnih elementov. To pomeni, da opis markovske verige z običajno markovsko matriko nima smisla, ker je preveč potratna.

Torej, glavni izziv naloge je, da se pametno definira stanja igre in učinkovito opiše prehodne verjetnosti med stanji.

17 Raketa

Raketa na vzletišču ima maso 300kg, kar vključuje tudi 180kg goriva. Raketni motor porabi 10kg goriva vsako sekundo. Tok izgorelih plinov brizga skozi potisne šobe s tako hitrostjo, da zagotavlja stalen potisk 5000N, dokler goriva ne zmanjka. Ko raketa vzleti, nanjo deluje sila upora, ki je odvisna od hitrosti v in je enaka $(0.1v^2)$N (v merimo v m/s), ter težnostni pospešek, ki je konstanten $g = 9.81$m/s.

1. Zapiši diferencialno enačbo in začetne pogoje, iz katerih bomo lahko ugotovili, kako se bo gibala raketa (v vertikalni smeri).

2. Izračunaj, v kolikšnem času bo raketa porabila vse gorivo.

3. Izračunaj višino in hitrost rakete v trenutku, ko zmanjka goriva.

4. Zaradi vztrajnosti se bo raketa tudi brez goriva še nekaj časa vzpenjala. Zapiši diferencialno enačbo in začetne pogoje, ki opisujejo to fazo poleta.

5. Koliko časa se bo raketa še vzpenjala in kakšna bo maksimalna višina, ki jo bo dosegla?

6. Ali je potrebno spremeniti diferencialno enačbo, da bo ta dobro opisala obnašanje rakete med fazo prostega padanja?

7. Naša raketa je preveč dragocena, da bi se lahko razbila pri padcu na tla, zato smo vanjo vgadili padalo, ki se bo odprlo, ko bo raketa padla do polovice maksimalne višine. Koliko časa bo raketa prosto padala in kakšno hitrost bo imela, ko se bo odprlo padalo?

8. Za spust s padalom bo potrebno ponovno spremeniti diferencialno enačbo. Kako, če veš, da je koeficient zračnega upora padala stokrat večji od koeficienta zračnega upora rakete same?

9. Koliko časa bo trajala faza leta s padalom in s kakšno hitrostjo bo raketa padla na tla?

Po uspešnem pristanku si konstruktorska ekipa najprej malo oddahne, nato pa se v njihove glave zarije črv dvoma: ali smo vse izpeljali, kot bi bilo treba?

1. Predpostavka o konstantnem gravitacijskem pospešku za namene te naloge ni smiselna. Popravite enačbo tako, da upoštevate gravitacijski pospešek v odvisnost od oddaljenosti rakete od Zemlje. Upoštevaj Newtonov gravitacijski zakon, privzemite, da je vsa masa Zemlje koncentrirana v njenem središču in od nekod izbrskajte podatek o premeru Zemlje. Izračunajte rešitev v tem primeru in jo primerjajte z rezultati osnovne naloge.

2. Zračni upor se spreminja z višino x. Recimo, da koeficient zračnega upora z višino pada tako kot funkcija ke^{-x}, kjer je k nek začetni koeficient (poiščite vrednost za koeficient v linearnem zakonu upora za gibanje skozi zrak). Zapišite enačbo v tem primeru, izračunajte (numerično, analitično ne bo šlo) rešitev in jo primerjajte z rešitvijo osnovne naloge.

3. Ali se rezultati, ki jih doseže naša raketa, odvisni od geografske širine našega izstrelišča (pomisli na vrtenje Zemlje okoli osi). Kakšne rezultate bi dosegla raketa, če bi bilo izstrelišče an ekvatorju? ali pa na Južnem polu? ali v Zgornjem Kašlju?

Ali imamo še kakšen pomislek?

18 Nacrtovanje poti robota

Povzetek

Robotsko vozilce se nahaja v sobi polni škatel, miz in drugega pohištva. Z uporabo parametricnih zlepkov sestavi pot, ki bo robota varno in hitro pripeljala iz tocke A v tocko B.

Opis nalog

1. Izpeljite matematicni model, ki bo opisal sobo in ovire, ki se nahajajo v sobi. Model dovolj poenostavite, da ne bo prevelikih težav z izvedbo (izlocite ukrivljene objekte, soba naj ima samo pravokotne vogale, ...).

2. Podobno naredite za robota. Matematicno opišite gibanje robotskega vozila. V matematicni obliki zapišite omejitve gibanja, ki jih ima robot. Hitrost in kot obracanja koles je omejen, obracanje na mestu ni mogoce, ... Model dovolj poenostavite, da ne boste imeli prevec težav z implementacijo.

3. Za dani tocki A in B, sestavite pot, ki bo robota pripeljala iz ene v drugo tocko. Krivulja, ki jo boste sestavili, bo tir težišca robota. Ce ste predpostavili, da robot ni tockast, morate prej ovire ustrezno odebeliti. Pot naj upošteva omejitve gibanja. Lahko najprej poišcete odsekoma linearno pot (npr. kot pot v ustreznem grafu), ki jo nato zamenjate s kubicnimi zlepki. Ko dobite gladko pot, morate preveriti, da še vedno ne seka ovir in da ima navzdol omejen krivinski radij (tj. ovinki niso preostri, tako da jih robot res lahko izpelje)

4. Izracunajte dolžino poti in najmanjši cas, ki ga robot porabi, da prevozi pot.

5. Rezultate graficno prikažite.

19 Simulacija igre tenisa

Pri tenisu sodelujeta dva igralca, ki z loparji poskusita žogico odbiti cez mrežo v nasprotnikovo polje. Igra se konca, ko enemu od igralcev ne uspe zadeti žogice ali pa žogica pristane v mreži ali izven igrišca. Napisali bi radi simulacijo teniške igre, pri cemer predpostavimo, da so dejanja igralcev vhodni podatki, vse ostale dogodke pa mora opisati naš program.

Osnovne enačbe

Let objekta z maso m pod vplivom gravitacije in zracnega upora lahko opišemo z drugim Newtonovim zakonom v obliki diferencialne enacbe

$$m\ddot{\mathbf{x}} = F_g + F_u,$$

kjer je $F_g = m \cdot [0, 0, g]^\mathsf{T}$ sila teže, F_u pa zracni upor. Le–tega lahko ocenimo s pomocjo kvadratnega zakona za zracni upor

$$F_u = -\frac{1}{2}\rho S C_u \dot{\mathbf{x}}|\dot{\mathbf{x}}|,$$

kjer je ρ gostota medija, S površina presecne ploskve, C_u pa koeficient upora, ki je odvisen od oblike predmeta. (Za kroglo je $C_u = 0.47$.)

Odboj žogice od tal lahko zelo poenostavljeno opišemo z odbojnim zakonom:

(a) komponenta hitrosti v normalni smeri zamenja predznak in se zmanjša za faktor med 0 in 1 (restituicijski koeficient),

(b) komponenta hitrosti, ki je pravokotna na normalno smer se v celoti ohrani.

Naloga

1. S katerimi matematičnimi objekti lahko opišemo igralca, loparja, žogico in igrišče, ki se pojavijo v igri tenisa?

2. Razcleni teniško igro na posamezne dogodke in sestavi matematicni model za vsak posamezen del.

3. Za vsak posamezen dogodek razmisli, katere metode (numericne, analiticne) boš uporabil, da boš poiskal rešitev.

4. Napiši podprograme, ki simulirajo posamezne elemente teniške igre in jih sestavi v koncni program. Koncni program naj pri danem gibanju igralcev izracuna let teniške žogice in odloci, kdaj je igra koncana.

20 Trasiranje ceste

Namen projekta je trasirati lepo speljano pot med pisarno 3.26 in menzo FRI (ali drugima dvema zanimivima lokacijama).

Lepo trasirane poti so sestavljene iz ravnih odsekov, iz odsekov *klotoid* (ali *Cornujevih krivulj*) in odsekov krožnih lokov. Klotoide so krivulje, ki imajo to lastnost, da ukrivljenost narašča ali pada linearno v odvisnosti od dolžine (tj. od naravnega parametra). Parametrizacija klotoide je:

$$x(t) = x_0 + \int_0^t \cos \beta u^2 \, du, \quad y(t) = y_0 + \int_0^t \sin \beta u^2 \, du.$$

Vaša naloga je, da zberete koordinate točk na najkrajši poti med obema zgradbama in jih smiselno razdelite na ravne odseke in ovinke. Ravne dele opišite z daljicami, ovinke pa zlepite s klotoidami do krožnih lokov. Pri tem mora biti ukrivljenost poti odsekoma linearna.

- **Konstrukcija poti.** Določiti morate točke, kjer ravni odseki preidejo v ovinke. Ravni odseki so preprosto daljice med krajnima točkama. Ovinke med dvema ravnima odsekoma pa sestavite kot zlepek odseka klotoide, odseka krožnega loka in odseka klotoide. Parametra x_0 in y_0 sta koordinati začetka ovinka, parametra in β pa določite tako, da bo krivulja, ki jo sestavlja odsek klotoide med ravnim odsekom in krožnim lokom, zvezna in gladka, njena ukrivljenost pa bo odsekoma linearna.

- **Izračun dolžine poti.** Dolžino izračunajte tako, da numerično izračunate natančno dolžino posameznih odsekov klotoid in dobljeno vrednost primerjajte z dolžino linearnega interpolanta.

21 Krožni omejen problem treh teles

Dve nebesni telesi, recimo zvezda in njen planet, v idealiziranem primeru krožita okrog skupnega masnega središča. Zanima nas gibanje satelita pod vplivom sile teže teh dveh teles. Satelit ima (zelo) majhno maso in nima vpliva na gibanje zvezde in planeta. Čeprav se tiri gibanja sprva zdijo nepravilni in kaotični, lahko v takem sistemu poičemo periodične tire, ki so se izkazali za uporabne za astronomska opazovanja.

Sistem Sonce–Zemlja je zelo blizu idealiziranemu primeru kroženja. Najbolj znan primer umetnega satelita, ki se giblje po natančno določenem periodičnem tiru približno 5 svetlobnih sekund stran od Zemlje, je najbrž SOHO (Solar and Heliospheric Observatory), ki že od leta 1995 redno spremlja dogajanje na Soncu. Naslednji planiran vesoljski teleskop (JWST, James Webb Space Telescope) bo utirjen v podobno orbito na nasprotni strani Zemlje.

Enačbe gibanja satelita

Ker predpostavljamo, da zvezda z maso M in planet z maso m krožita okrog skupnega masnega središča, bomo enačbe gibanja zapisali v vrtečem koordinatnem sistemu, kjer masi M in m mirujeta. Označimo

$$\mu = \frac{m}{M+m} \quad \text{ter} \quad \bar{\mu} = 1 - \mu = \frac{M}{M+m}.$$

V brezdimenzijskih koordinatah (dolžinska enota je kar razdalja med masama M in m) postavimo maso M v točko $(-\mu, 0, 0)$, maso m pa v točko $(\bar{\mu}, 0, 0)$. Označimo z R in r oddaljenost satelita s položajem (x, y, z) od mas M in m, tj.

$$R = R(x,y,z) = \sqrt{(x+\mu)^2 + y^2 + z^2},$$
$$r = r(x,y,z) = \sqrt{(x-\bar{\mu})^2 + y^2 + z^2}.$$

Enačbe gibanja satelita so potem:

$$\ddot{x} = x + 2\dot{y} - \frac{\bar{\mu}}{R^3}(x+\mu) - \frac{\mu}{r^3}(x-\bar{\mu}),$$
$$\ddot{y} = y - 2\dot{x} - \frac{\bar{\mu}}{R^3}y - \frac{\mu}{r^3}y,$$
$$\ddot{z} = -\frac{\bar{\mu}}{R^3}z - \frac{\mu}{r^3}z,$$

kjer je $\dot{x} = \frac{dx}{dt}$, $\ddot{x} = \frac{d^2x}{dt^2}$ in podobno za y ter z. Za natančno izpeljavo glej [1].

Sistem opisan z zgornjimi enačbami ima 5 stacionarnih točk, ki jih označimo z L_1, L_2, L_3, L_4, L_5 in jim pravimo *Lagrangeeve točke*. Položaje teh točk izračunamo kot ničle polinomov 5. stopnje, glej [1, stran 13].

Naloga

1. Poiščite položaje Lagrangeovih točk (v brezdimenzijskih koordinatah) za nekaj naravnih sistemov:
   - Sonce–Zemlja,
   - Zemlja–Luna,
   - Sonce–Jupiter.
2. Poiščite in narišite tire gibanja satelitov, v zgornjih treh sistemih. Kaj se zgodi, če satelit na začetku miruje v eni od Lagrangeevih točk? Kako se satelit giblje, če je na začetku v bližini Lagrangeevih točk L_4 ali L_5?
3. Narišite sliko Poincaréjeve preslikave (presek tira z ravnino $y = 0$) v bližini točke L_1 za sistem Sonce–Zemlja in poiščite periodičen tir okrog točke L_1 tega sistema. (Periodičen tir lahko dobite tako, da poiščete fiksno točko Poicaréjeve preslikave. Uporabite diskretizirano Newtonovo ali večrazsežno sekantno metodo.)

Literatura

[1] `http://www.math.rutgers.edu/~jmireles/hw4Notes.pdf`

22 Ray-tracing

Razširjena metoda v računalniški grafiki je tako imenovan *ray-tracing*, ki je uporabna zlasti kadar želimo doseči fotorealistične efekte, nimamo pa hudih omejitev za čas, ki ga lahko porabimo, da narišemo sliko (torej praviloma ni primerna za renderiranje slik v realnem času). Različic te metode je veliko in imajo različna imena, osnovna ideja vseh takšnih metod pa je, da sledimo svetlobnim žarkom (ki jih pošiljamo iz svetlobnega vira ali iz kamere, če želimo hitrejšo metodo, ali oboje), poiščemo točke, kjer svetlobni žarki zadenejo objekte v danem prostoru, izračunamo kote, pod katerim se žarki odbijejo (ali lomne kote, če je objekt vsaj deloma prepusten za svetlobo), morda sledimo odbitemu žarku do enega ali več naslednjih odbojev, in točko na zaslonu pobarvamo glede na lastnosti materiala, odbojnega kota, kje je žarek končal pot, števila odbojev (če dopuščamo več odbojev) itd. Matematično gledano, je glavni problem v tem, kako izračunati presečišča in odboje svetlobnih žarkov (ki jih praviloma predstavimo s premicami) z objekti v prostoru. Kako zahteven je ta problem, je odvisno od tipa in predstavitve objekta, na časovno zahtevnost algoritma pa vplivajo seveda tudi število objektov, njihova kompleksnost, število odbojev oz. lomov, ki jih dopuščamo, lasnosti materialov in svetlobne efekte, ki jih upoštevamo.

Opis naloge

V domači nalogi bi se osredotočili na tip objektov, ki lahko predstavljajo veliko različnih oblik, imajo pa načeloma zelo enostaven in enoten matematičen opis: ploskve podane kot rešitve enačbe oblike

$$f(x, y, z) = 0. \tag{1}$$

V tak tip objektov spadajo recimo:

1. Ravnine. Ravnine so določene z enačbo

   $$ax + by + cz = d$$

2. Krogle. Krogla s središčemo v (x_0, y_0, z_0) in polmerom r ima enačbo

   $$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = r^2$$

   Podobne enačbe določajo elipsiode in ostale ploskve drugega reda.

3. Grafi funkcij dveh spremenljivk. Če imamo funkcijo dveh spremenljivk $z = u(x, y)$, lahko to prepišemo v

   $$z - u(x, y) = 0$$

   in imamo enačbo oblike (1).

Ali bomo žarke pošiljali iz svetlobnega vira ali kamere, ali bomo sledili odbitemu žarku ali tudi lomu svetlobe, ali bomo dopuščali več odbojev ali več objektov in kakšne konkretne lastnosti materialov bomo upoštevali pri določanju barve, prepuščamo pobudi študentov, ki si lahko pri tem pomagajo s poljubnimi viri, ki jih sami najdejo. V vsakem primeru pa moramo znati izračunati presečišče žarka in objekta in izračunati odboj ali lom žarka.

Žarek z izhodiščem v točki $T_0(x_0, y_0, z_0)$ in smerjo $\vec{v} = (a, b, c)$ predstavimo z enačbami

$$\begin{aligned} x(t) &= x_0 + at \\ y(t) &= y_0 + bt \\ z(t) &= z_0 + ct, \end{aligned} \tag{2}$$

kjer je $t \geq 0$ parameter. Začnemo torej žarek v točki (x_0, y_0, z_0) (kjer je $t = 0$) in se premikamo vzdolž žarka, tako da povečujemo parameter t za določen korak, dokler ne zaznamo spremembo predznaka funkcije $f(x, y, z)$ iz enačbe (1), sprememba predznaka te funkcije namreč pomeni, da je žarek trčil ob ploskev. Ali na tak način dejansko zaznamo trk, je lahko odvisno od velikosti koraka, ki ga je zato potrebno pametno izbrati. Če je prevelik, lahko zgrešimo primer, ko premica dvakrat seka ploskev, če je premajhen, pa lahko iskanje presečišč predolgo traja. Ko smo enkrat našli presečišče na intervalu $[t_1, t_2]$, ga moramo natančneje izračunati. Predlagamo uporabo Newtonove metode, da rešimo enačbo

$$g(t) := f(x_0 + at, y_0 + bt, z_0 + ct) = 0, \tag{3}$$

z začetnim približkom recimo na sredini intervala, torej v točki, kjer je $t = (t_1 + t_2)/2$. Odvod funkcije v (3), ki ga potrebujemo za Newtonovo metodo, se izračuna po formuli

$$g'(t) = af_x(x_0+at, y_0+bt, z_0+ct)+bf_y(x_0+at, y_0+bt, z_0+ct)+cf(x_0+at, y_0+bt, z_0+ct),$$

kjer so f_x, f_y in f_z parcialni odvodi funkcije f po x, y in z.

Ko izračunamo presečišče premice s ploskvijo, recimo, da to presečišče označimo s $T_1(x_1, y_1, z_1)$, dobimo normalo na ploskev v tej točki kot

$$\vec{n} = (f_x(x_1, y_1, z_1), f_y(x_1, y_1, z_1), f_z(x_1, y_1, z_1)).$$

Smer odbitega ali ulomljenega žarka, vpadni kot in ostalo lahko potem določimo z uporabo elementarne geometrije iz vektorjev $\vec{v}$ in $\vec{n}$. Za začetek lahko preskusite dve možnosti:

1. Izračunate presečišče žarka, ki izhaja iz kamere, z objektom in barvo določite na podlagi kosinusa kota med normalo in vektorjem med presečiščem in svetlobnim virom.

2. Izračunate presečišče žarka, ki izhaja iz kamere, z objektom in barvo določite na podlagi kosinusa kota med odbitim žarkom in svetlobnim virom.

23 Bézierovi zlepki

Konstruirali bi radi gladek zlepek sestavljen iz Bézierovih krivulj, ki gre skozi dane tocke v ravnini. Za izbrani zlepek, izracunamo še dolžino zlepka in morebitna samopresecišca.

Bézierove krivulje

Naj bodo $\mathbf{b}_i$, $i = 0, 1, \ldots, n$, točke v ravnini. Bézierova ravninska krivulja stopnje n je definirana s predpisom

$$\mathbf{b}(t) = \sum_{i=0}^{n} \mathbf{b}_i \, B_i^n(t), \;\; B_i^n(t) = \binom{n}{i} t^i \, (1-t)^{n-i}, \quad t \in [0, 1].$$

Točkam $\mathbf{b}_i$ rečemo kontrolne točke Bézierove krivulje, daljicam, ki jih zaporedoma povezujejo, pa kontrolni poligon. Pri danem parametru $t_0 \in [0, 1]$, lahko točko $\mathbf{b}(t_0)$ na krivulji izračunamo direktno po formuli, ali po De Casteljauovem algoritmu takole:

$$\mathbf{b}_i^r(t_0) = (1 - t_0) \, \mathbf{b}_i^{r-1}(t_0) + t_0 \, \mathbf{b}_{i+1}^{r-1}(t_0), \quad r = 1, \ldots, n, \quad i = 0, \ldots, n - r,$$

kjer je $\mathbf{b}_i^0(t_0) = \mathbf{b}_i$ in $\mathbf{b}_0^n(t_0) = \mathbf{b}(t_0)$. Pri tem zgornji indeksi ne pomenijo potenciranje, ampak nivo, na katerem se trenutno nahajamo!

Naloga

V okviru naloge boste konstruirali zlepek sestavljen iz Bézierovih krivulj iste stopnje, ki je naceloma majhna(med 2 in 5). Pricakujemo, da boste opravili naslednje naloge.

1. Izpeljite, katerim pogojem morajo zadošcati kontrolni poligoni krivulj v zlepku, da bo zlepek zvezno odvedljiv. Ce ne gre v splošnem, pa vsaj za krivulje 3. stopnje.

2. Napišite pomožne funkcije za racunanje tock in izris Bézierove krivulje s podanim poligonom. Izracun naj bo izveden z De Casteljauovem algoritmom. Rezultate preverite z direktnim izracunom po formuli.

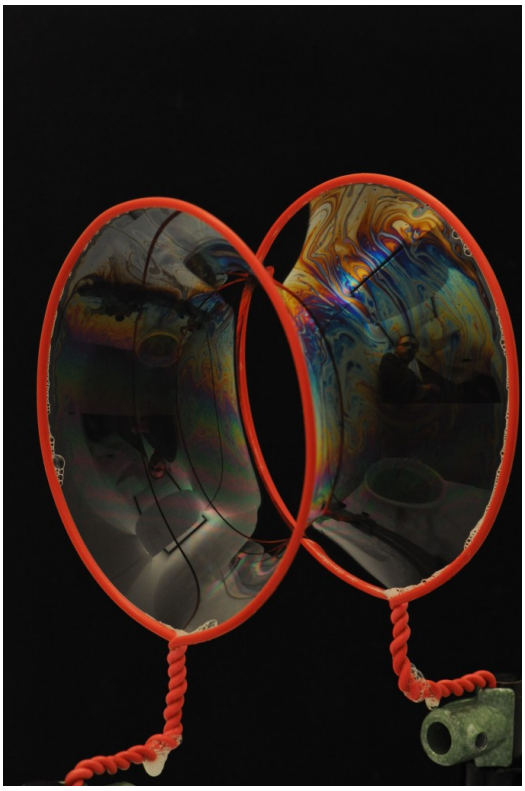
3. Za dane tocke v ravnini poišcite zvezno odvedljiv zlepek sestavljen iz Bézierovih krivulj iste stopnje.

4. Izpeljite formule in pomožne programe za izracun dolžine zlepka in morebitnega samopresecišca.

5. Napišite program, ki graficno prikaže vse rezultate iz prejšnjih tock. Program preiskusite na razlicnih naborih tock.

24 Minimalne ploskve

Poiskali bi radi obliko opne iz milnice, ki je napeta na žicno zanko dane oblike.



Slika 1: Milnica na dveh krožnicah (Foto: `http://soapbubble.dk`)

Naloga je poiskati obliko milnice, ce je podana oblika žicne zanke. Milnica zavzame ploskev, katere elasticna energija je minimalna. V primeru, da je milnica homogena, je elasticna energija sorazmerna s površino, zato je tudi površina minimalna.

1. Zapišite, kako bi elasticno opno predstavili z matematicnim objektom (*funkcija dveh spremenljivk, parametricna ploskev.*) Kako bi predstavili obliko žicne zanke?

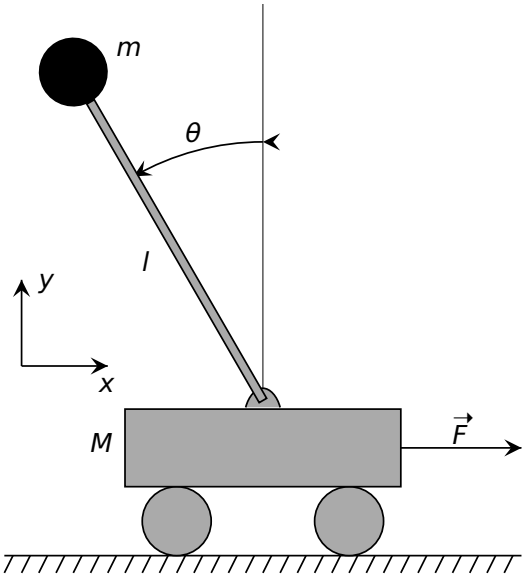
2. Matematicni objekt iz prve tocke diskretizirajte. Objekt predstavite z diskretnim objektom npr. z grafom. Zapišite pogoj minimalne elasticne energije za diskreten objekt, ki predstavlja minimalno ploskev in problem prevedite na linearni sistem enacb.

3. Sestavite algoritem, ki bo poiskal rešitev diskretiziranega problema iz prejšnje tocke. Poskrbite, da bo prostorska in casovna zahtevnost optimalna (matrike sistema ne hranite v celoti). Za reševanje linearnega sistema lahko uporabite katero od iteracijskih metod za reševanje linearnih sistemov, npr. Gauss-Seidlovo ali SOR iteracijo.

Za inspiracijo si oglejte film o minimalnih ploskvah arhitekta Freia Otto.

25 Balansiranje navpične palice

Simulirajte balansiranje palice, ki je vpeta na vodoravno ploščad.



Slika 2: Model obrnejnega nihala na vozicku

Upravljamo lahko s silo $\vec{F}$, ki deluje horizontalno na ploščad. Silo $\vec{F}$ imenujemo kontrolna sila. Za razlicne izbire kontrolne sile $\vec{F}$ obravnaj lastnosti sistema (fazni portret, narava stacionarnih rešitev).

Naloga

V sklopu projektne naloge naredite vsaj naslednje:

1. Iz fizikalnih zakonov izpeljite diferencialne enacbe, ki opisujejo sistem. Diferencialne enacbe prevedite na sistem 1. reda

$$\frac{d}{dt}\vec{x} = A(\vec{x}) + Bu(t),$$

   kjer je clen $Bu(t)$ ustreza prispevku kontrolne sile $\vec{F}$ in je $u(t)$ skalarna funkcija, s katero izvajamo kontrolo nad sistemom. Funkcijo $u(t)$ imenujemo vhod, vektor stanja $\vec{x}$ pa izhod kontrolnega sistema.

2. Najprej obravnavajte sistem brez prisotnosti kontrolne sile $\vec{F}$.

3. Obravnavajte sistem, ce je kontrolna funkcija $u(t)$ linearno odvisna od vektorja stanj $\vec{x}$

$$u(t) = -K \cdot x(t) = -(K_1 x_1 + K_2 x_2 + \ldots).$$

Ali lahko izberemo vektor K, da bo ravnovesje palice v navpicni legi stabilno?

Ce vam prejšnje tocke niso povzrocale težav, lahko poskusite razširiti nalogo, bodisi tako, da bolj podrobno preucite, kako izbrati primeren $u(t)$, ali pa izdelate simulacijo vozicka s kakšnim naprednim graficnem motorjem (npr. unity 3D).

Literatura

Priporocam ucbenik v spletnem tecaju Analysis and Design of Feedback Systems iz leta 2004.

26 Problem milijon teles

Imamo n teles v prostoru z masami m_i $(i = 1, \ldots, n)$, ki se gibljejo pod vpilvom medsebojnih sil gravitacije. Radi bi preučili dinamiko takega sistema.

Za primer $n = 2$ je možno analitično izračunati vse možne rešitve, vsaj za točkaste mase. Že za problem treh teles pa ni več mogoče analitično v vsej splošnosti zapisati rešitve, ampak je to možno samo za posebne primere.

Za primer, ko je n več tisoč ali milijionov (na primer simulacije trka galaksij), se da analitično povedati kaj o kvalitativnih lastnostih takih sistemov, sicer pa ni druge izbire kot da računamo numerične približke gibanja.

Enačbe gibanja za tak sistem ni težko zapisati. Rešiti je treba Newtonove enačbe

$$\ddot{\mathbf{x}}_i = \sum_{j \neq i} G \frac{m_j}{|\mathbf{x}_j - \mathbf{x}_i|^3} (\mathbf{x}_j - \mathbf{x}_i), \text{ za } i = 1, \ldots, n \tag{4}$$

kjer je $\mathbf{x}_i = (x_i, y_i, z_i)$ položaj i-te mase in G gravitacijska konstanta (ki je za naše potrebe lahko tudi 1). Tu smo seveda predpostavili, da so mase točkaste, kar je smiselno, če so radiji objektov zanemarljivi v primerjavi z razdaljami (in tudi v primeru, če so objekti okrogli in homogeni).

Numerične metode za reševanje tega problema lahko v grobem razdelimo na:

- **Poštene metode :** Direktno rešujemo sistem enačb (4). Problem takega načina je v tem, da moramo pri naivnem pristopu na vsakem koraku (mogoče večkrat) izračunati $n-1$ sil za vsako od od n mas, torej je časovna zahtevnost računanja sil reda n^2, kar je lahko pri večjih n-jih precej počasno.

- **Nepoštene metode :** Z različnimi peonastavitvami, kot je razdelitev točk na skupine, ki imajo drevesno strukturo, lahko časovno zahtevnost računanja sil spravimo na red $n \log(n)$. Vendar pa je pri tem potrebno narediti kompromise zaradi katerih je rezultat izračunan samo približno.

Naloga

1. Napišite funkcijo, ki računa gibanje velikega števila zvezd. Program mora biti dovolj učinkovit, da lahko v realnem času vidimo potek dogajanja vsaj za nekaj tisoč zvezd.

2. Preiskusite vsaj na dveh primerih:

   (a) galaksija, kjer na začetku zvezde krožijo okrog središča

(b) mimobežni galaksiji oz. galaksiji, ki se trčita

27 Sestavljeno nihanje

Matematično nihalo brez upoštevanja upora opisuje enačba

$$\ddot{x} = -\alpha \sin x,$$

kjer je $\alpha = lg$, l je dolžina nihala, g pa težnostni pospešek.

Izpeljite enačbo, ki opisuje vsiljeno nihanje matematičnega nihala, kjer vsiljujemo nihanje tako, da točko, kjer je nihalo pritrjeno, periodičon pomikamo navzdol in navgor z amplitudo $\alpha \geq 0$ in s frekvenco $\beta > 0$. Dobili boste enačbo

$$\ddot{x} + (1 + \alpha \sin(\beta t)) \sin x = 0.$$

Napišite izpeljavo!

- Za nekaj različnih začetnih pogojev $(x_0, v_0) = (x(t_0), \dot{x}(t_0))$ in za nekaj izbranih vrednosti α in β poiščite rešitev $(x(t), v(t))$ sistema in jo narišite v ravnini (x, v) (tj. v fazni ravnini). Uporabite algoritem za Eulerjevo metodo ali za metodo Runge-Kutta, ki ste ga izpeljali na vajah.

  Pazite, spremenljvka x predstavlja kot, ki naj zavzame vrednosti na osnovnem intervalu $[-\pi, \pi]$. To pomeni, da določata točki $(x + 2\pi, v)$ in (x, v) pravzaprav isto točko v fazni ravnini, zato morate za vrednosti x, ki so izven intervala $[-\pi, \pi]$ najprej odšteti ustrezen mnogokratnik števila 2π!

- Perioda vsiljenega nihanja je $\tau = 2\pi/\beta$, po tem času se točka, kjer je nihalo pritrjeno, vrne v začetni položaj. Zanima nas, ali se je tudi nihalo povrnilo v začetni položaj, in če ne, kaj se je z njim zgodilo. V ta namen označimo s $\phi_\tau(x_0, v_0)$ funkcijo, ki točki (x_0, v_0) priredi vrednost $(x(\tau), v(\tau))$, kjer je $(x(t), v(t))$ rešitev enačbe z začetnimi pogoji (x_0, v_0). (Tole tu je treba dobro premisliti, da razumemo, za kaj gre! Če ne razumete, pridite vprašat!).

- Napišite program, ki bo za izbrano začetno vrednost (x_0, v_0) in izbrane vrednost α in β izračunal in izrisal rekurzivno zaporedje točk $(x_0, v_0), (x_i, v_i) = \phi_\tau(x_{i-1}, v_{i-1})$. (Seveda ne morete narisati vseh neskončno mnogo točk zaporedja, naričite jih dovolj, da se bo videli, kaj se z zaporedjem dogaja).

- Kaj opazite? Ali je obnašanje dobljenega zaporedja podobno, če začetne vrednosti (x_0, v_0) spreminjate pri fiksiranih vrednostih parametrov

α in β? Narišite z različnimi barvami nekaj takih zaporedij hkrati. Kaj pa, če vrednosti α in β spreminjate, se slika zelo spreminja, ali je podobna? Poskusite za β izbrati tudi kakšno iracionalno število!

28 Pajkova mreža

Radi bi v realnem času simulirali in (morda interaktivno) prikazali dinamiko sistema masnih točk, ki so povezane prožnimi vezmi, pod vplivom zunanjih sil.

Primer je recimo nihanje pajkove mreže (kjer bi imeli zelo elastične povezave med vozlišči) ali gibanje tkanine ali plapolanje zastave (ki bi jo predstavili z veliko pravokotno mrežo, kjer so sosednja vozlišča med sabo povezana z bolj togimi povezavami).

Objekt, katerega dinamiko modeliramo, predstavimo z grafom na n točkah, kjer imamo za vsako vozlišče podatke o položaju in hitrosti. V splošnem imamo torej podatke

$$(\mathbf{x}_i, \mathbf{v}_i, P_i), \;\; i = 1, \ldots, n,$$

kjer sta

$$\mathbf{x}_i = (x_1^i, x_2^i, x_3^i) \text{ in } \mathbf{v}_i = (v_1^i, v_2^i, v_3^i)$$

vektorja trenutnega položaja in hitrosti i-tega vozlišča in

$$P_i = \{j : \; j\text{-ta točka je povezana z } i\text{-to točko}\}$$

množica indeksov vozlišč, s katerimi je i-ta točka povezana.

Predpostavljamo, da so nekatera vozlišča pripeta na neke statične objekte v prostoru (v primeru pajkove mreže so to recimo vozlišca, ki so pripeta na veje drevesa ali steno).

Drugače povedano, imamo določeno podmnožico $F \subset I = \{1, \ldots, n\}$ množice indeksov vozlišč, katerih položaj $\mathbf{x}_i$ je ves čas fiksen (in hitrost $\mathbf{v}_i$ ves čas enaka 0).

Za vsa ostala vozlišča $i \notin F$ je dinamika določena z 2. Newtonovim zakonom (masa krat pospešek je enako vsoti vseh sil, ki delujejo na vozlišče):

$$\ddot{\mathbf{x}}_i = \sum_{j \in P_i} \mathbf{f}_{ji} + \mathbf{u}_i + \mathbf{g} \tag{5}$$

kjer je $\mathbf{f}_{ji}$ sila, s katero j-to vozlišče deluje na i-to vozlišče, $\mathbf{u}_i$ sila upora oz. trenja, ki deluje na i-to vozlišče, in

$$\mathbf{g} = (0, 0, -g)$$

konstantni vektor težnega pospeška. V enačbi (5) smo Newtonovo enačbo že okrajšali z maso. Predpostavljamo tudi, da so vse mase vozlišč enake in da so mase samih povezav zanemarljive v primerjavi z masami vozlišč.

Enačba (5) predstavlja sistem $3 \times n$ diferencialnih enačb 2. reda. Vemo, da je za numerično reševanje sistema bolj primerno, če ga preoblikujemo v sistem $6 \times n$ diferencialnih enačb 1. reda

$$\begin{aligned} \dot{\mathbf{x}}_i &= \mathbf{v}_i \\ \dot{\mathbf{v}}_i &= \sum_{j \in P_i} \mathbf{f}_{ji} + \mathbf{u}_i + \mathbf{g} \end{aligned} \tag{6}$$

ki ga lahko rešujemo z običajno metodo Runge-Kutta 4. reda.

Za silo $\mathbf{f}_{ji}$ v (6) imamo več izbir, v vsakem primeru pa privzemite, da je odvisna samo od položaja sosednjih vozlišč, torej

$$\mathbf{f}_{ji} = \mathbf{f}(\mathbf{x}_j, \mathbf{x}_i)$$

Za primer popolnoma elastične povezave (naš model pajkove mreže) lahko po Hookeovem zakonu vzamemo enostavno

$$\mathbf{f}(\mathbf{x}_j, \mathbf{x}_i) = k(\mathbf{x}_j - \mathbf{x}_i) \tag{7}$$

kjer je k koeficient prožnosti (pajkove) niti. V takem primeru bi se recimo v odsotnosti pripetih točk oz. zunanjih sil nit skrčila v 0.

Bolj kompliciran primer bi bil, da predpostavimo, da ima posamezna povezava ravnovesno lego (stanje, v kateri ni sile med vozlišči) na neki dani fiksni razdalji d (za katero zaradi enostavnosti predpostavimo, da je za vsa vozlišča enaka). Silo pri odmiku od ravnovesne razdalje pa dobimo spet po Hookeovem zakonu. Potem lahko silo izračunamo po formuli

$$\mathbf{f}(\mathbf{x}_j, \mathbf{x}_i) = k(\|\mathbf{x}_j - \mathbf{x}_i\| - d)\frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|}$$

Če hočemo bolj togo povezavo, lahko to modeliramo, tako da tu vzamemo precej koeficient prožnosti k kot v primeru popolnoma elastične povezave (pri tem pa moramo paziti, da ne vzamemo prevelikega zaradi numerične stabilnosti simulacije).

Za silo upora $\mathbf{u}_i$ v (6) je najbolj enostavno privzeti, da je premosorazmeren trenutni hitrosti vozlišča

$$\mathbf{u}_i = -c\mathbf{v}_i,$$

kjer je $c \geq 0$ koeficient upora. Pri večji vrednosti za c se bo sistem hitreje umiril v stacionarni legi, kjer je totalna energija (kinetična energija plus potencialna energija vseh vozlišč) minimalna.

Za vrednost $c = 0$ bi se teoretično energija morala ves čas ohranjatti. Zaradi numeričnih napak se bo izmerjena energija sicer ves čas nekoliko spreminjala, kar pa lahko izkoristite za oceno, ali je simulacija dovolj natančna ali pa je morda potrebno zmanjšati korak v Runge-Kutta metodi.

Napišite torej še funkcijo, ki izračuna trenutno totalno energijo sistema:

$$E = \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{v}_i\|^2 + \frac{1}{4}\sum_{i=1}^{n}\sum_{j\in P_i} k\|\mathbf{x}_j - \mathbf{x}_i\|^2 + \sum_{i=1}^{n} g x_3^i \tag{8}$$

Prva vsota tu predstavlja kinetično energijo vseh vozlišč, druga vsota predstavlja prožnostno potencialno energijo vseh povezav (za primer, da smo silo izračunali po formuli (7)), tretja vsota pa težnostno potencialno energijo (v poštev pride samo tretja z koordinata i-tega vozlišča x_3^i).

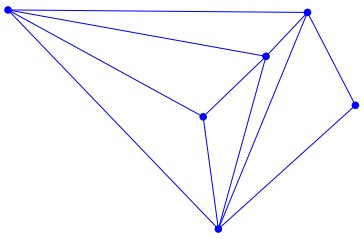
Simulacijo najprej poganjate za primer $c = 0$ in sproti merite, kaj se dogaja z totalno energijo, izračunano po formuli (8). V primeru, da je simulacija premalo natančna, se lahko hitro zgodi, da bo energija zbezljala (kar bo sicer vidno tudi iz same simulacije). Ko pa korak zmanjšate toliko, da se ta energija ne spreminja veliko, veste, da imate dovolj natančno simulacijo. Potem lahko povečate c, da bo učinek simulacije bolj realističen. Tudi v tem primeru lahko merite energijo, ki se bo zmanjševala, dokler se sistem ne ustali v ravnovesni legi.

29 Površina parametrizirane ploskve

Cilj tega projekta je preprost algoritem za izračun približne površine parametrizirane ploskve.

Približek za površino pa lahko dobimo z naslednjim postopkom:

- V območju D izberemo n točk in poiščemo triangulacijo na izbranih točkah. *Triangulacija* na točkah (u_i, v_i), $i = 1, \ldots, n$, je razrez dela ravnine, iz katerega so zajete točke, na trikotnike. Oglišča (u_i, v_i) so povezana s stranicami tako, da se nobeni dve stranici ne sekata, liki, ki pri tem nastanejo pa so vsi trikotniki. Tule je primer triangulacije na 6 točkah.

  Da dobite triangulacijo na daih točkah si lahko pomagate s kakšnih iz množice algoritmov za triangulacijo (na primer Delaunayevo triangulacijo) ravninskih območij, ki so na voljo.

  Množico oglišč triangulacije označimo z V. Stranice triangulacije bomo označevali kot pare oglišč $((u_i, v_i), (u_j, v_j))$, množico vseh stranic triangulacije pa z E. Trikotnike bomo označevali kot trojice oglišč, množico vseh trikotnikov triangulacije pa s T.

- Če je točk $(u_i, v_i) \in V$ dovolj in če so trikotniki v množici T dovolj majhni, bo triangulacija območja D določala triangulacjo ploskve $\mathbf{r}(u, v)$ z:
  - oglišči v točkah $\mathbf{r}_i = \mathbf{r}(u_i, v_i)$, $(u_i, v_i) \in V$
  - s stranicami $(\mathbf{r}_i, \mathbf{r}_j)$, kjer je $((u_i, v_i), (u_j, v_j)) \in E$
  - in trikotniki $(\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k)$, kjer je $((u_i, v_i), (u_j, v_j), (u_k, v_k)) \in T$.

  Napišite algoritem, ki bo preveril, ali je dobljena množica oglišč, stranic in trikotnikov res triangulacija ploskve v $\mathbb{R}^3$. Preveriti morate, da se nobeni dve stranici ne sekata in da nobena stranica ne prebode kakšnega od trikotnikov.

- Približek za površino ploskve je vsota ploščin vseh trikotnikov triangulacije.

- Algoritem stestirajte (vsaj) na naslednjih primerih, kjer primerjate približek, dobljen na n točkah (izberite nekaj vrednosti n) s pravo vrednostjo površine:
  - del ravnine $z = x + y$, ki leži v notranjosti valja $x^2 + y^2 = 1$ (prava vrednost je $\sqrt{3}\pi$),
  - del paraboloida $(u, v, u^2 + v^2)$, $u, v \in [0, 1] \times [0, 1]$ (prava vrednost je $1/24(24 + \arctan(44/117) - 2\arctan(2) + 14\log(5) \cong 7.44626$),
  - ploskev $(u\cos v, u\sin v, v)$, $[0, 1] \times [0, 2\pi]$ (prava vrednost je $8\pi/3$),
  - sfera s polmerom 1.

30 Določanje položajev radarjev

Radi bi onesposobili protizračno obrambo neke države in v ta namen moramo najprej odkriti položaje radarjev raketnih sistemov. Vemo, da ima sovražnik n radarjev razporejenih na lokacijah z neznanimi kordinatami (u_k, v_k), $k = 1, \ldots, n$. Na voljo imamo letala, ki lahko na varni razdaljo z neko natančnostjo merijo skupno jakost radarskih signalov, ne pa tudi smeri. Vemo, da jakost signala posameznega radarja pada s kvadratom razdalje, v dani točki (x, y) je tako jakost k-tega signala enaka

$$j_k(x, y) = \frac{c}{(x - u_k)^2 + (y - v_k)^2},$$

kjer je c konstanta, ki nam jo zagotovi proizvajalec, ki je pred leti sovražniku prodal radarje. V m točkah s znanimi koordinatami (x_i, y_i), $i = 1, \ldots, m$ opravimo meritve, ki nam dajo vrednosti z_i za skupno jakost. Približno torej velja

$$\begin{aligned} z_1 &\doteq \sum_{k=1}^{n} j_k(x_1, y_1) \\ z_2 &\doteq \sum_{k=1}^{n} j_k(x_2, y_2) \\ &\vdots \\ z_m &\doteq \sum_{k=1}^{n} j_k(x_m, y_m) \end{aligned}$$

Naloga

1. Najmanj koliko meritev je potebno opraviti, da lahko vsaj teoretično določimo položaje radarjev?

2. V smislu metode najmanjših kvadratov zapišite funkcijo, ki jo je potrebno minimizirati, da dobimo najboljšo oceno za neznane položaje (x_i, y_i).

3. Sistem enačb, ki jih dobite po metodi najmanjših kvadratov, je nelinearen. Uporabite gradientno ali Newtonovo metodo (ali za primerjavo oboje) za reševanje sistemov nelinearnih enačb, da za dane podatke izračunate oceno za neznane položaje. Preizkusite delovanje takega algoritma na simuliranih podatkih.

4. S pomočjo similuranih podatkov analizirajte vpliv števila meritev, povprečnih merskih napak in povprečnih oddaljenosti meritev od radarskih postaj na natančnost rezultatov.

31 Valovna enačba

Valovna enačba ima v splošnem obliko

$$\frac{\partial^2 u}{\partial t^2} = c^2 \triangle u,$$

kjer je $u(t, \mathbf{x})$ funkcija časa t in prostorskih koordinat $\mathbf{x}$, c hitrost valovanja in $\triangle$ Laplaceov operator. V dveh dimenzijah in v kartezičnih koordinatah dobimo

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right).$$

K tej parcialni diferencialni enačbi (PDE) spadajo še začetni pogoji (za začetno vrednost u in začetno hitrost $\partial u/\partial t$)

$$\begin{aligned} u(0, x, y) &= f(x, y) \\ \frac{\partial u}{\partial t}(0, x, y) &= g(x, y) \end{aligned}$$

in morda še pogoji na robu območja, ki ga imamo.

Valovno enačbo lahko analitično obravnavamo na različne načine in možni so tudi različni pristopi za numerično reševanje.

Ena možnost je, da to PDE prevedemo na sistem navadnih diferencialnih enačb na naslednji način: recimo da nas zanima rešitev na pravokotniku $[0, 1] \times [0, 1]$. To območje diskretiziramo, da dobimo mrežo enakomerno porazdeljenih točk (x_i, y_j), $i, j = 1, \ldots, n$ in iščemo približke rešitve

$$u_{i,j}(t) := u(t, x_i, y_j),$$

na tej mreži točk (vsak $u_{i,j}$ je potem funkcija časa). Laplaceov operator funkcije na takšni mreži lahko približno izrazimo z

$$\triangle u(x_i, y_j) \doteq \frac{1}{h^2}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j})$$

Dobimo sistem $n \times n$ (ali morda $n-1 \times n-1$, če so vrednosti na robu fiksne) navadnih diferencialnih enačb 2. reda

$$\ddot{u}_{i,j} = \frac{c^2}{h^2}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}),\ i, j = 1, \ldots n$$

Kot vemo, sisteme 2. reda pogosto prevedemo na dvakrat večji sistem 1. reda z uvedbo nove spremenljivke. V tem primeru lahko za novo spremenljivko vzamemo hitrostno polje $v_{i,j} = \frac{\partial u}{\partial t}(t, x_i, y_j)$ in ta sistem prepišemo v sistem

$$\begin{aligned} \dot{u}_{i,j} &= v_{i,j} \\ \dot{v}_{i,j} &= \frac{c^2}{h^2}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}), \end{aligned}$$

ki ga lahko rešujemo z običajnimi metodami za reševanje navadnih diferencialnih enačb (Eulerjeva metodo, Runge-Kutta ...)

Naloga

Napišite program, ki v realnem času prikazuje potek rešitve valovne enačbe po zgornji metodi na $n \times n$ mreži (za n ni potrebno vzeti števila, ki je več kot 100, tudi za recimo $n = 30$ se dobi realistično animacijo) pri različnih začetnih in robnih pogojih, ali celo pri sprotnih spremembah vrednosti funkcije u, ki jih vnese uporabnik.

Pri tem lahko uporabite Eulerjevo metodo, za natančnejše in bolj stabilno simulacijo pa lahko tudi metodo Runge-Kutta (v tem primeru potrebujete več kopij matrik $u_{i,j}$ in $v_{i,j}$).

Kot robni pogoj preizkusite vsaj naslednja dva primera:

1. Na robu je vrednost funkcije u in vrednost hitrosti v ves čas konstantno 0 (se pravi $u_{1,i} = v_{1,i} = u_{n,i} = v_{n,i} = u_{i,1} = v_{i,1} = u_{i,n} = v_{1,n} = 0$ za vse $i = 1, \ldots n$).

   Tak primer bi simuliral valovanje membrane, ki pritrjena na kvadraten okvir.

2. Robovi so prosti. V tem primeru je treba prilagoditi enačbe na robu kvadrata. Za oglišče $u_{1,1}$ se dobi enačbi

$$\begin{aligned} \dot{u}_{1,1} &= v_{1,1} \\ \dot{v}_{1,1} &= \frac{c^2}{h^2}(u_{2,1} + u_{1,2} - 2u_{1,1}), \end{aligned}$$

   za točke na robu $u_{1,j}$, kjer je $2 \leq j \leq n-1$ pa

$$\begin{aligned} \dot{u}_{1,j} &= v_{1,j} \\ \dot{v}_{1,j} &= \frac{c^2}{h^2}(u_{2,j} + u_{1,j+1} + u_{1,j-1} - 3u_{1,j}), \end{aligned}$$

   in podobno še za ostale robne primere.

   Takšen robni pogoj bi simuliral recimo valovanje vodne gladine v kvadratnem bazenu (ki ima konstantno globino in ima valovanje samo navpično komponento).

Za začetni pogoj lahko preizkusite primer, ko je na začetku $u(0, x, y) = 0$ in $v(0, x, y) = 0$, razen v eni izbrani točki v notranjosti, ki ji daš neko negativno začetno hitrost. Takšen začetni pogoj bi recimo simuliral padec

majhnega kamna v vodo ali na membrano. Potem lahko preizkusite podoben začetni pogoj za več točk hkrati, da se bo videla interferenca valovanja, odboj valov od roba itd.

Zgornje enačbe določajo sistem, kjer se energija ne izgublja in se valovanje zato nikoli se umiri. Zaradi numeričnih približkov se pri večjih vrednosti za začetne hitrosti lahko celo zgodi, da rešitev postane nestabilna, še posebej pri reševanju z Eulerjevo metodo. Za večji realizem lahko v enačbo dodate člen, ki duši valovanje, sorazmerno s trenutno hitrostjo. Druga enačba potem dobi obliko

$$\dot{v}_{i,j} = \frac{c^2}{h^2}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\right) - kv_{i,j}$$

za neko konstanto k.

32 Dinamika populacij v prehranjevalni verigi

Modeliramo dinamiko populacij različnih vrst v ekosistemu, kjer so odnosi med vrstami predstavljeni s takšnim grafom

kjer puščica označuje, da se ena vrsta prehranjuje z drugo. Označimo velikosti populacij teh šestih vrst s spremnljivkami $x_1, \ldots, x_6$. Recimo, da imamo tak model: sprememba velikosti posamezne populacije x_i po nekem kratkem času dt je sorazmerna velikosti populacije x_i in količini

$$\begin{aligned} & \text{koeficient naravnega prirastka oz. smrtnosti} \\ + \ & \text{koeficient*(količina hrane)} \\ - \ & \text{koeficient*(smrtnost zaradi ulova)}, \end{aligned}$$

kjer sta količina hrane ter delež smrti zaradi ulova spet sorazmerni velikosti populacij vrst, s katerimi se i-ta vrsta prehranjuje ter vrst, ki se prehranjujejo z njo. V splošnem lahko torej dinamiko i-te populacije opišemo z diferencialno enačbo oblike

$$\frac{dx_i}{dt} = x_i(b_i + \sum_{i \neq j} a_{ij} x_j),$$

kjer so koeficienti a_{ij} pozitivni, negativni ali enaki 0 glede na naravo interakcije z j-to vrsto.

Naloga

1. Zapišite sistem diferencialnih enačb za za narisan ekosistem. Konkretne vrednosti koeficientov določite sami. Koeficient naravnega prirastka b_i naj bo za rastline pozitiven, za ostale vrste pa negativen.

2. Poiščite neničelno stacionarno rešitev sistema.

3. Napišite funkcijo, ki izračuna numerično rešitev sistema za dano matriko koeficientov in začetni pogoj.

4. Preuzkusite numerično rešiti sistema za začetne pogoje, ki se samo malo razlikujejo od stacionarne rešitve.

5. Preučite obnašanje sistema za različne vrednosti koeficientov in začetnih pogojev in poskusite pridelati pojave kot so *cikličnо obnašanje*, *asimptotično* cikl*ično obnašanje* in *kaos*.

33 Naboji na ploskvah

Imamo n enakih točkastih nabojev, ki ležijo na (omejeni) implicitno podani ploskvi v $\mathbb{R}^3$

$$f(\mathbf{x}) = 0 \tag{9}$$

kjer je $f : \mathbb{R}^3 \to \mathbb{R}$ in smo označili $\mathbf{x} = (x, y, z)$. Naboji se med sabo odbijajo, ker pa predpostavljamo, da je ploskev omejena, obstaja ravnovesna lega, kjer je vsota vseh sil na vsak naboj na ploskvi enaka nič (pravzaprav je ravnovesna lega taka, da vsota sil na posamezen naboj deluje v smeri, ki je normalna na ploskev). Radi bi napisali funkcijo, ki za dano ploskev (npr. sfera, elipsoid, torus itd) in število nabojev n izračuna to ravnovesno lego. Število nabojev n pri tem ne bo več kot nekaj deset ali morda sto.

Če označimo položaj i-tega naboja z $\mathbf{x}_i = (x_i, y_i, z_i)$, $i = 1, \ldots, n$, lahko problem formuliramo kot problem iskanja minimuma skupne potencialne energije

$$V(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{i \neq j} \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|} \tag{10}$$

(fizikalne enote so tu izbrane tako, da pride izraz v številcih enak 1) pri pogojih

$$f(\mathbf{x}_i) = 0, \quad i = 1, \ldots, n \tag{11}$$

Tak problem imenujemo problem vezanih ekstremov. Klasična metoda za reševanje problema vezanih ekstremov so Lagrangeovi množitelji. Vendar je zaradi neenoličnosti rešitev implementacija numeričnega reševanja tega konkretnega problema s pomočjo Lagrangeovih množiteljev brez dodatnih pogojev ali predpostavk lahko težavna (poleg tega, da Lagrangeovi množitelji niso del učnega načrta).

Zato predlagamo naslednji algoritem, ki je kombinacija gradientne metode in reševanja sistemov nelinearnih enačb (za katere je tu v splošenm najbolj primerna Newtonova metoda):

1. Za dano začetno oz. trenutno konfiguracijo $\mathbf{z}_0 = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ ($\mathbf{z}_0$ je torej vektor s $3n$ koordinatami) izračunamo vrednost gradienta potenciala (10) $\mathrm{grad}V(\mathbf{z}_0)$ in sistem premaknemo v smeri nasprotno od gradienta (ker iščemo minimum potenciala)

   $$\mathbf{z}_1 = \mathbf{z}_0 - h \cdot \mathrm{grad}V(\mathbf{z}_0) \tag{12}$$

   za nek majhen korak h. Pri tem ignoriramo pogoje (11), kar pomeni, da se naboji v splošnem pri tem koraku odmaknejo stran od ploskve (13).

2. Nove položaje nabojev, ki jih dobimo po izračunu (12), popravimo tako, da vsak naboj prestavimo nazaj na ploskev (11) v smeri premic, ki so vzporedne vektorjem

$$\mathbf{n}_i = \mathrm{grad} f(\mathbf{x}_i), \quad i = 1, \ldots, n$$

ki so pravokotne na ploskev (13) (ali skoraj pravotne). Za poseben primer enotske sfere je ta korak možno preprosto izvesti tako, da vektorje $\mathbf{x}_i$ enostavno normiramo. V splošnem pa je potrebno omenjene premice parametrizirati (recimo, da tako parametrizacijo označimo z $\mathbf{r}_i(t)$) in rešiti enačbo

$$f(\mathbf{r}_i(t)) = 0$$

za vsak $i = 1, \ldots, n$, za kar bo zadostovalo že par korakov Newtonove iteracije.

3. Koraka 1. in 2. ponavljamo, dokler se položaji nabojev ne spreminjajo več oz. dokler razlike med položaji pred in po izvedenih korakih 1. in 2. ne postanejo dovolj majhne. Ko pridemo namreč do konfiguracije, kjer se po korakih 1. in 2. položaji nabojev ne spreminjajo več, to pomeni, da pri izračunu koraka (12) naboje porivamo s ploskve le še v smeri normal na ploskev in smo dosegli ravnovesno lego.

34 Geodetke na implicitno podanih ploskvah

V evklidskem prostoru (na primer v ravnini $\mathbb{R}^2$) je najkrajša pot med dvema točkama daljica. V ukrivljenih prostorih (kot so ploskve v prostoru $\mathbb{R}^3$) v splošnem ni 'ravnih' poti, najkrajša pot med dvema točkama pa je krivulja, ki jo imenujemo *geodetka*. V gladkih ukrivljenih prostorih se geodetko dobi kot rešitev določene diferencialne enačbe, za izpeljavo katere je v splošnem potrebno nekaj znanja diferencialne geometrije. Za poseben primer implicitno podane ploskve v $\mathbb{R}^3$ pa je možno izpeljati enačbe za geodetke precej enostavneje.

Cilj naloge je zapisati enačbe za geodetke za nekaj konkretnih primerov implicitno podanih ploskev (npr. sfero, elipsoid, torus itd) in napisati funkcijo, ki bo za dano ploskev numerično izračunala geodetko v dani začetni točki in smeri na ploskvi.

Izpeljava enačb

Dano imamo funkcijo $f : \mathbb{R}^3 \to \mathbb{R}$, ki implicitno določa ploskev v $\mathbb{R}^3$ z enačbo

$$f(\mathbf{x}) = 0 \tag{13}$$

kjer smo označili $\mathbf{x} = (x, y, z)$. Pogoj, da krivulja $\mathbf{x}(t) \in \mathbb{R}^3$ leži na ploskvi, je torej, da $\mathbf{x}(t)$ zadošča enačbi (13). Z odvajanjem enačbe (13) po t lahko ta pogoj zapišemo tudi kot

$$\mathrm{grad} f(\mathbf{x}(t)) \cdot \mathbf{v}(t) = 0 \tag{14}$$

kjer smo z $\mathbf{v}(t) = \dot{\mathbf{x}}(t) = (\dot{x}(t), \dot{y}(t), \dot{z}(t))$ označili tangentni vektor na krivulji. Označimo še (normiran) normalni vektor na ploskev v točki $\mathbf{x}$

$$\mathbf{n}(\mathbf{x}) = \frac{\mathrm{grad} f(\mathbf{x})}{\|\mathrm{grad} f(\mathbf{x})\|}$$

in še projekcijski matriki na normalo ter tangentno ravnino v točki $\mathbf{x}$

$$Q = \mathbf{n}\mathbf{n}^\mathsf{T},\ P = I - Q = I - \mathbf{n}\mathbf{n}^\mathsf{T} \tag{15}$$

kjer uporabljamo matrično množenje ($\mathrm{grad} f$ in s tem tudi $\mathbf{n}$ je vektor stolpec) in izpuščamo pisanje argumenta $\mathbf{x}$. Potem lahko pogoj (14), da krivulja leži na ploskvi (če malo premislimo) zapišemo tudi kot

$$\mathbf{v} = P\mathbf{v} \tag{16}$$

Enačbi (16) zadošča vsaka krivulja na ploskvi (ne samo geodetka). Če hočemo zapisati enačbo za geodetko, moramo (16) še enkrat odvajati po t:

$$\dot{\mathbf{v}} = \dot{P}\mathbf{v} + P\dot{\mathbf{v}}$$

Geometrijski pogoj, da je krivulja na ploskvi geodetka, je, da mora biti vektor pospeška $\dot{\mathbf{v}}$ enak nič v vseh smereh, ki so tangentne na ploskev. Drugače rečeno, projekcija vektorja $\dot{\mathbf{v}}$ na tangentno ravnino mora biti enaka nič:

$$P\dot{\mathbf{v}} = 0$$

To lahko razumemo tudi fizikalno: telo, ki se giblje po geodetki 'čuti' pospešek kvečjemu v smeri, ki je normalna na ploskev, saj na sami ploskvi nanj ne deluje nobena sila.

Tako lahko zapišemo enačbo, ki ji mora zadoščati geodetka kot

$$\dot{\mathbf{v}} = \dot{P}\mathbf{v} = -\dot{Q}\mathbf{v} \tag{17}$$

ki skupaj z enačbo $\dot{\mathbf{x}} = \mathbf{v}$ tvori sistem 6 diferencialnih enačb. Začetni pogoji so začetni položaj $\mathbf{x}_0$ in začetni tangentni vektor $\mathbf{v}_0$.

Enačba (17) se na pogled zdi precej enostavna, vendar je treba upoštevati, da je projekcijska matrika $Q = Q(\mathbf{x}(t))$ tu funkcija točke $\mathbf{x}$ na ploskvi in šele posredno funkcija parametra t, tako da se treba nekoliko potruditi pri izračunu odvoda $\dot{Q}$, če ga hočemo izraziti samo z odvodi funkcije f, ki definira ploskev, in tangentnega vektorja $\mathbf{v}$ v točki $\mathbf{x}$. Da bi izrazili (17) na čimbolj enostaven in ekspliciten način definiramo še Hessejevo matriko parcialnih odvodov 2. reda funkcije f:

$$\mathrm{Hess}f = \begin{bmatrix} f_{xx} & f_{xy} & f_{xz} \\ f_{yx} & f_{yy} & f_{yz} \\ f_{zx} & f_{zy} & f_{zz} \end{bmatrix} \tag{18}$$

Potem lahko z nekaj truda sistem enačb za geodetke na ploskvi zapišemo v obliki

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\frac{\mathrm{grad}f(\mathbf{x})}{\|\mathrm{grad}f(\mathbf{x})\|^2}\mathbf{v}^{\mathsf{T}}\mathrm{Hess}f(\mathbf{x})\mathbf{v} \end{aligned} \tag{19}$$

Naloga

1. Izpeljite enačbo (19) iz enačbe (17). *Namig :* Pri izpeljavi je nekajkrat treba upoštevati, da je tangentni vektor na krivulji $\mathbf{v}$ v vsaki točki pravokoten na normalo ploskve $\mathbf{n}$ oz. $\mathrm{grad}f$ (da torej velja enačba (14)).

2. Za čim več primerov implicitno podanih ploskev (torej funkcij f) napišite funkcije, ki za dano točko $\mathbf{x}$ na ploskvi vrnejo $\mathrm{grad} f$, projekcijsko matriko P iz (15) in Hessejevo matriko (18). Te funkcije boste uporabili kot argumente funkcije, ki bo numerično reševala sistem (19).

3. Napišite funkcijo, ki za dano ploskev $f = 0$, točko $\mathbf{x}_0$ na ploskvi in tangentno smer $\mathbf{v}_0$ numerično izračuna in nariše geodetko na ploskvi. Lahko se zgodi, da pri numeričnem reševanju (vsaj če koraki niso dovolj majhni) krivulja počasi 'zleze' stran s ploskve. Razmlslite o tem, kako bi se to dalo popraviti.