Numerična linearna algebra 2013/2014

1. domača naloga

Rešitve stisnite v ZIP datoteko z imenom `ime-priimek-vpisna-1.zip` in jih oddajte preko spletne učilnice (`http://ucilnica.fmf.uni-lj.si`) najkasneje do 7. 5. 2014 do 23. ure. Priložite poročilo, v katerem za vsako nalogo opišete postopek reševanja, zapišete rešitev, in komentirajte rezultat. Obvezno priložite programe, s katerimi ste naloge rešili. Programi naj bodo smiselno poimenovani in razporejeni v mapah. Prav morate za vsako nalogo oddati skripto, ki vrne zahtevane rezultate. Naloge morajo biti rešene v Matlabu (uporabite lahko tudi Octave, FreeMat ali Scilab). Naj bodo $c_1c_2c_3c_4$ zadnje 4 cifre vaše vpisne številke.

Če imate kakšno vprašanje o nalogah ali Matlabu, se obrnite na asistenta ali profesorja. Če menite, da je vprašanje zanimivo tudi za ostale, uporabite forum.

Singularni razcep je močno orodje, ki se ga uporablja za regularizacijo problemov, kompakten razvoj po bazi nekega prostora, reševanje problemov najmanjših kvadratov, kompresijo podatkov, odstranjevanje šuma, ... V tej domači nalogi si bomo ogledali, kako lahko singularni razcep uporabimo za primerjavo dokumentov v različnih jezikih in glajenje signala.

Zadosti je, da naredite samo nalogo s primerjavo dokumentov, brez dodatnih vprašanj. Lahko pa naredite nalogo z glajenjem signala in vsaj eno točke naloge 2 ali dve točki v 1. Dodatna vprašanja prinesejo dodatne točke.

1. Prepoznavanje števk

Na pošti bi radi avtomatsko prepoznavali ročno napisane števke. Števke so podane kot sivinske slike dimenzije 16×16. Matrični zapis slike pretvorimo v vektor, tako da sestavimo stolpce matrike v vektor dimenzije 256. Podana je baza učnih slik, kjer je `dzip.mat` vektor števk in `azip.mat` matrika katere stolpci predstavljajo slike. Bazo za določeno števko s dobimo na naslednji način. Stolpce testne matrike, ki predstavljajo to števko, sestavimo v matriko A_s. Nato naredimo singularni razcep matrike $A_s = U_s\Sigma_s V_s^*$. Za bazo števke B_s vzamemo nekaj prvih stolpcev (5 do 20) matrike U_s.

- Poiščite bazo B_s za vsako števko s. Vzemite kar prvih 5 stolpcev U_s.
- Prepričajte se, da prvi singularni vektor v bazi dobro predstavlja števko.
- Dobljene baze uporabi za klasifikacijo števk. Števko t, ki jo predstavlja vektor t, klasificirate tako, da izračunate $\min_z ||B_s z - t||_2$ za vsak s in pogledate pri katerem s je dosežen minimum. Napišite funkcijo `klas.m`, s pomočjo katere boste lahko klasificirali števke.
- Uspešnost klasifikacije preverite na testnih podatkih `dtest.mat` in `testzip.mat`. Podatki so podani enako kot učni podatki. Za vsako števko vrnite delež uspešnosti klasifikacije.
- Ali velikost baze števk bistveno vpliva na uspešnost klasifikacije?

Pri risanju slik si pomagajte s funkcijo `ima2.m`, ki sprejme sliko v vektorski obliki. Podatke v datotekah s končnico mat naložite z ukazom `load`.

2. Primerjava dokumentov v različnih jezikih

Na voljo imate podatke, ki se nahajajo v Matlabovem polju dimenzije 3, `trlangs` (`tslangs`), ki vsebuje 91375 (1000) dokumentov, ki so zapisani po stolpcih. Elementi Matlabovega polja so po vrsti dokumenti v angleščini, španščini in nemščini. Stolpci z istimi indeksi predstavljajo isti dokument v različnih jezikih, npr. `trlangs{1}(:, i), trlangs{2}(:, i), trlangs{3}(:, i)` . Za predstavitev dokumentov je uporabljena vreča besed. To pomeni, da vsaka vrstica i predstavlja besedo, vsak stolpec j predstavlja dokument. Tako je element (i, j) število pojavitev besede i v dokumentu j. To je najbolj preprosta predstavitev, ki v splošnem ne bo najbolj učinkovita. Boljše rezultate dobimo, če damo besedam, ki se zelo pogosto pojavljajo (taki so recimo vezniki in glagol biti) nižjo utež. To lahko dosežemo z uporabo utežitve *tfidf*

$$a_{ij} \mapsto a_{ij} \times \log\left(\frac{\texttt{število dokumentov}}{\texttt{število dokumentov v katerih se pojavi beseda i}}\right)$$

Naslednja izboljšava, ki jo uporabimo je normalizacija dokumentov v posameznem jeziku. Tako se lahko izognemo prevelikemu vplivu enciklopedičnih dokumentov, ki so po malem podobni kar vsemu.

Za izračun podobnosti med dokumenti bomo uporabili kar kosinusno podobnost:

$$\mathrm{Cos}(x, y) = \frac{\langle x, y\rangle}{||x||_2||y||_2}.$$

Opišimo glavno idejo naše metode, poravnane matrike korpusov dokumentov D_1, D_2, D_3 sestavimo v bločno matriko D in poiščemo njeno aproksimacijo nizkega ranga k, s $^{(k)}$ označimo, da smo vzeli samo prvih k stolpcev matrike, $S^{(k)}$ pa je dimenzije $k \times k$.

$$D = \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} \quad S \quad V^T \qquad \approx \begin{bmatrix} U_1^{(k)} \\ U_2^{(k)} \\ U_3^{(k)} \end{bmatrix} \quad S^{(k)} \quad V^{(k)^T}$$

Zdaj lahko vsak dokument v jeziku i preslikamo v skupni prostor topikov V na naslednji način:

$$x_i \mapsto V \cdot S^+ \cdot U_i^+ \cdot x_i$$

Ker je baza V ortogonalna in iščemo samo aproksimacijo, je naša končna preslikava enaka:

$$x_i \mapsto {S^{(k)}}^+ \cdot {U_i^{(k)}}^+ \cdot x_i$$

Najprej preizkusite pripravljena orodja in podatke. Najprej izvedite ukaz `load raw_data` s katerim naložite vaše podatke. Oglejte si vse spremenljivke. Na voljo imate besede in imena učnih in testnih dokumentov. V resnici gre za angleške, španske in nemške wikipedijine članke. To pomeni, da lahko dokument z imenom Slovenia, najdete na naslovu `http://en.wikipedia.org/wiki/Slovenia`. Lahko uporabite tudi funkcijo `show_doc`.

- Funkcija `gb` vam priročno vrne porabo pomnilnika za spremenljivko v GB.
- Funkcija `retrieve_sparse` vam vrne razpršen vektor, vrečo besed.
- Funkcija `show_words` vam hitro vrne besede, ki se pojavijo v dokumentu.
- Funkcijo `compose_seperated_string` lahko uporabite, da iz cell polja naredite niz ločen s presledki.
- Skripto `create_Hash` uporabite, da zgenerirate HashMap za iskanje besed.

Podrobnejša navodila z zgledi uporabe se nahajajo v skripti `nalozi_Podatke`.

(i). Za izračun podobnosti bomo dokumente najprej preslikali v nižjedimenzionalni prostor. To bomo storili s pomočjo odrezanega singularnega razcepa dimenzije 500. Pravimo, da smo dokument preslikali v prostor topikov. V tem prostoru bomo zdaj računali podobnost.
Na voljo imate več že izračunanih singularnih razcepov.
V datoteki `Baza_frekvence_en_es_de.mat` je izračun singularnega razcepa za originalne neobdelane podatke `trlangs`.

V datoteki `Baza_Tfidf_en_es_de.mat` je izračun singularnega razcepa za podatke `trlangs`, kjer je bila uporabljena utežitev tfidf na posameznem jeziku, vendar brez normalizacije.
V datoteki `Baza_Tfidf_Normal_en_es_de.mat` je izračun singularnega razcepa za podatke `trlangs`, kjer je bil najprej uporabljen tfidf na vsakem jeziku in nato še normalizacija bločnih stolpcev.

Učinkovito implementiraj preslikavo v skupni prostor za vsak primer. Preslikavo bo potrebno uporabiti večkrat, tako da naj bo kar se da učinkovita. Ne pozabi najprej narediti tfidf utežitve in normalizacije, kjer je le to potrebno.

(ii). Za testno množico preveri, kako se napovedi podobnosti, ki jih dobite z uporabo preslikave v skupni prostor, ujemajo z dejanskim stanjem. Izvorni dokument q je iz jezika A, med tem ko so testni dokumenti v jeziku B. Primerjavo naredimo z uporabo naše preslikave.

(a) Prvi kriterij za kvaliteto je povprečni inverzni rang podobnosti (MAPMR).
Naj bo $t(q)$ - vektor podobnosti med izvornim dokumentom q in vsemi ciljnimi testnimi dokumenti. Kako visoko je rangiran prijateljski dokument (poravnan z izvornim) v $t(q)$, idealno bi bilo, da bi bil na prvem mestu? To je rang dokumenta, inverzni rang dokumenta pa je recipročna vrednost. Za vsak par jezikov, vrstni red je pomemben, zgenerirajte seznam inverznih rangov. MAPMR za par jezikov dobite tako, da izračunate povprečje inverznih rangov.

(b) Ocenjevanje po prvem kriteriju lahko zavaja, saj so lahko sami testni dokumenti med sabo precej podobni. Najprej izračunamo vektor podobnosti v_1 med q_1 in vsemi testnimi dokumenti v jeziku A. Poiščemo najbližji testni dokument r v jeziku B. Nato izračunamo vektor podobnosti v_2 med r in vsemi testnimi dokumenti v jeziku B. Končna ocena je korelacijski koeficient med r_1 in r_2. V Matlabu ga dobite z ukazom `corr`. Podobno kot prej za vsak par jezikov izračunamo povprečje teh ocen.

(c) Primerjaj kvaliteto preslikav, glede na zgornji meri, kjer aproksimiraš s preslikavo ranga $50, 100, 200, 300, 400, 450$ in 500. Če ti dela težave hitrost izračuna, naredi primerjavo samo za nekaj prvih dimenzij.

Dodatno S pomočjo ukaza `svds` lahko izračunaš singularni razcep tudi sam. Namesto na treh jezikih, izračunaj zdaj preslikave samo paroma. Ali se kvaliteta preslikav poveča?

(iii).
- Preveri, če metoda deluje na realnih dokumentih, ki so prevodi. Uporabiš lahko kar Google translate.
- Napiši še funkcijo, ki poišče 10 besed v dokumentu 1, ki so najbolj podobne celemu dokumentu 2. To lahko uporabite za ilustracijo podobnosti.
- Napiši funkcijo, ki poišče n besed z največjimi utežmi v topiku (vektorju). Na tak način ponazorite nekaj vodilnih singularnih vektorjev (topikov).

3. Signal

Večina metod zajemanja signala zgladi podatke do neke mere. Naloga bo modelirati proces in z uporabo metode najmanjših kvadratov poiskati signal pred glajenjem.

Podana je diskretna časovna vrsta podatkov $x \in \mathbb{R}^n$. Proces glajenja bomo modelirali z Gaussovim filtrom. Konvolucijski koeficienti so

$$d_k = \frac{1}{b} e^{-k^2/2\gamma^2} \quad \text{za} \quad -h \leq k \leq h,$$

kjer je

$$h = c_1 + c_2 + c_3 + c_4, \quad \gamma = c_1 + c_4, \quad n = 200.$$

Konstanto b izberemo tako, da velja $\sum_{k=-h}^{h} d_k = 1$. Zglajen izhod $y \in \mathbb{R}^n$ je potem podan kot konvolucija x z d :

$$y_i = \sum_{k=-h}^{h} d_k x_{i+k} \quad \text{za} \quad i = 1, \ldots, n,$$

kjer je $x_i = 0$ za $i < 1$ in $i > n$.

(i). V Matlabu konstruirajte $n \times n$ matriko, da velja $y = Ax$. Narišite graf koeficientov c_k v odvisnosti od k.

(ii). Za vhod

$$x_k^{\text{pulse}} = \begin{cases} 1 & \text{za } 50 \leq k \leq 100, \\ 0 & \text{drugače.} \end{cases}$$

Narišite x^{pulse} in Ax^{pulse} na isti graf in preverite, da množenje z A zgladi podatke.

(iii). Narišite graf singularnih vrednosti σ_k v odvisnosti od k. Kaj to pomeni za naš problem?

(iv). Narišite prvih šest singularnih vektorjev matrike A. Kakšne so slike?

(v). Signal x konstruirajte kot

$$x_k = \begin{cases} 1 & \text{za } 50 \leq k < 100, \\ 4 & \text{za } 100 \leq k < 150, \\ 0 & \text{drugače.} \end{cases}$$

Naj bo $y_{\text{iz}} = Ax + w$, kjer je w naključni šum podan kot

`randn('state', vpisna); w=0.1*randn(n, 1).`

Poizkusite rešiti problem najmanjših kvadratov. Kaj opazite?

(vi). Veliko singularnih vrednosti A je zelo majhnih. To pomeni, da meritev v smeri pripadajočih desnih singularnih vektorjev preglasi šum. Uporabite odrezan singularni razcep matrike A, $A_{\text{est}} = \widetilde{V}\widetilde{\Sigma}^{-1}U^*$, za regularizacijo problema, kjer obdržimo samo r komponent.

Za vrednosti $r = 5, 10, 15, 30, 50$ izračunajte $x_{\text{est}} = A_{\text{est}} y_{\text{iz}}$ in narišite x_{est}. Kaj opazite?

(vii). Za $1 \leq r \leq 35$ izračunajte normo napake $||x - x_{\text{est}}||$ in narišite graf. Kaj se dogaja z napako?

(viii). Določite *najboljši* r in narišite ocenjen x_{est}.