Stochastické optimalizačné metódy - zadania úloh

3. decembra 2025

Poznámky ku skúške

Každý z Vás si vyberie jednu úlohu z prvej pätice a jednu úlohu z druhej pätice (pozri zadania uvedené nižšie). Jednu z úloh je možné nahradiť Vašou vlastnou optimalizačnou úlohou (niekedy majú študenti vlastný zaujímavý optimalizačný problém), pokiaľ Vám ju schválim. Za riešenie úlohy budem považovať optimalizačný/é program/y. Používať môžete `R`, `MATLAB`, `Python`, `Julia`, `C++`, ale pokiaľ nepoužijete `R`, prineste si vlastný notebook s funkčnou inštaláciou zvoleného prostredia. Vaše programy budete demonštrovať počas skúšky. Pokiaľ napíšete riešenia v `R` a budete chcieť používať môj počítač, pošlite mi ich pred skúškou na radoslav.harman@fmph.uniba.sk. Použiť môžete akúkoľvek optimalizačnú metódu, čiže aj takú, ktorú sme na prednáške nespomínali, samozrejme pokiaľ je vhodná a veľmi dobre jej rozumiete. Môžete tiež akýmkoľvek spôsobom využívať kódy programov, ktoré sme používali na prednáške. Zadania aj riešenia môžete prediskutovať so spolužiakmi, ako aj s umelou inteligenciou, budem však kontrolovať, či riešeniam detailne rozumiete. Programový kód zapisujte prehľadne, napríklad v štýle tidyverse. Pokiaľ je to v danej úlohe možné, ilustrujte názorne graficky výsledok výpočtu, prípadne aj priebeh výpočtu. Vypracovaný program si dôkladne prekontrolujte na viacerých čo najodlišnejších vstupoch. Využite pochopenie problému, teoretické vedomosti a nadhľad na overenie správnosti výstupu. Ku kontrole som Vám pri každej úlohe pripísal nejaké poznámky (viď **Test**).

Organizačné informácie ku skúške nájdete na stránke k predmetu: `http://www.iam.fmph.uniba.sk/ospm/Harman/teaching.htm`.

Pri hodnotení programov budem zohľadňovať nasledovné kritériá: vhodnosť použitej metódy, správnosť, úplnosť a efektívnosť riešenia, presnosť výsledkov (cca 0b-20b), kvalita prezentácie, presnosť vyjadrovania v súlade so zaužívanou terminológiou, schopnosť reagovať na otázky a schopnosť modifikovať program podľa požiadaviek (cca 0b-20b), originalita prístupu, kritické zhodnotenie nedostatkov, tvorivé nápady týkajúce sa možností vylepšenia a rozšírenia riešenia (cca 0b-10b)[1].

Orientačné hodnotenie: $F_x[0, 20)$, $E[20, 25)$, $D[25, 30)$, $C[30, 35)$, $B[35, 40)$, $A[40, \infty)$.

Poznámka: Ak v zadaniach nájdete nejaké chyby (alebo iné námety na vylepšenie), oznámte mi to prosím; prvému z Vás, ktorý ma chybu upozorní, udelím prémiové body.

[1] Keďže hodnotím aj originalitu vzhľadom k už prezentovaným riešeniam ostatných študentov, môžu mať výhodu tí študenti, ktorí sa prihlásia na skorší termín a tiež tí študenti, ktorí si zvolia pre ostatných menej atraktívne úlohy.

1 Turbína

Na obvod kruhovej turbíny chceme do n evidištantných pozícií umiestniť lopatky so známymi hmotnosťami $m_1, \ldots, m_n$, a to tak, aby ich ťažisko bolo čo najbližšie stredu turbíny. Napíšte dva programy implementujúce dve principálne rôzne optimalizačné metódy;[2] vstupom každého z týchto programov bude vektor hmotností lopatiek a výstupom bude poradie, v ktorom je vhodné tieto lopatky umiestniť. Môžete využiť to, že táto úloha má blízko úlohe *travelling salesman problem*.

Test: Pre $n = 10$ a $m_i = i$, $i = 1, \ldots, 10$, je možné umiestniť lopatky do evidištantných pozícií na obvode turbíny tak, že ich ťažisko je v úplne presnom strede kruhu turbíny, napríklad v poradí 6, 3, 2, 7, 10, 5, 4, 1, 8, 9. Váš program by mal byť schopný nachádzať pre tento prípad optimálne riešenie v rozmedzí maximálne niekoľkých sekúnd.

2 Drony

Roj M dronov sa nachádza vo formácii určenej pozíciami $(x_1^b, y_1^b), \ldots, (x_M^b, y_M^b)$. (Pre jednoduchosť uvažujeme len rovinnú situáciu, nie priestorovú.) Cieľom je, aby sa tento roj preskupil do novej formácie, ktorá pozostáva z pozícií $\{(x_1^f, y_1^f), \ldots, (x_M^f, y_M^f)\}$. Každý dron letí rovnakou konštantnou rýchlosťou. Tiež predpokladáme, že si drony navzájom neprekážajú, čiže všetky drony budú letieť k svojej určenej pozícii po priamke. Napíšte program, ktorého vstupom bude postupnosť $B = (x_1^b, y_1^b), \ldots, (x_M^b, y_M^b)$ iniciálnych pozícií dronov $1, \ldots, M$ a množina pozícií $F = \{(x_1^f, y_1^f), \ldots, (x_M^f, y_M^f)\}$ požadovanej finálnej formácie a výstupom bude priradenie, ktorý z dronov má letieť ku ktorému bodu z F. Chceme pritom minimalizovať L_p-normu $(\sum_{i=1}^M t_i^p)^{1/p}$, kde $t_1, \ldots, t_M$ sú časy letov dronov do dosiahnutia ich určenej pozície v F a $p \in [1, \infty)$ je ďalší vstupný parameter.[3] Táto úloha má blízko k úlohám označovaným ako *assignment problems*.

Test: Pre $M \leq 10$ by mal Váš program na bežnom počítači nachádzať (takmer) optimálne riešenia do niekoľkých sekúnd. Vyskúšajte viacero vstupov, ale aj také, v ktorých je optimálne riešenie "evidentné". Výsledky sa dajú pekne zobraziť (ako množina šípok určujúcich trasy dronov z východzích pozícií do cieľových pozícií).

3 Batoh

Do batohu s nosnosťou M kilogramov chceme povyberať predmety tak, aby sme maximalizovali ich celkovú cenu. Vyberať môžeme z n predmetov so známymi hmotnosťami $m_1, \ldots, m_n$ kilogramov a známymi cenami $c_1, \ldots, c_n$. Napíšte dva programy implementujúce dve principálne rôzne optimalizačné metódy;[4] vstupom každého z týchto programov bude vektor hmotností všetkých

[2] Maximálne jeden z týchto programov môže používať optimalizačnú procedúru, ktorú ste nepísali Vy.

[3] Všimnite si, že ak $p = 1$, tak minimalizujeme celkovú preletenú trasu, čiže cca celkovú spotrebu energie a pre $p \to \infty$ minimalizujeme čas, za ktorý roj nadobudne konfiguráciu F. Iné hodnoty parametra p môžu reprezentovať kompromis medzi "ekonomickosťou" a "rýchlosťou" dosiahnutia F.

[4] Maximálne jeden z týchto programov môže používať optimalizačnú procedúru, ktorú ste nepísali Vy.

predmetov, ktoré sú k dispozícii, vektor ich cien, nosnosť batoha a výstupom bude zoznam (podmnožina) predmetov, ktoré je vhodné vybrať. Táto úloha sa v angličtine nazýva *0-1 knapsack problem*. Berte na vedomie, že hmotnosti aj ceny môžu byť necelé čísla.

Test: Pre $n \leq 20$ by mal Váš program na bežnom počítači nachádzať optimálne, alebo takmer optimálne riešenia do niekoľkých sekúnd. Úlohy na testovanie by mali byť rozmanité, ale majte na pamäti, že ťažšie, a teda zaujímavejšie prípady tejto úlohy sú také, v ktorých sú ceny približne úmerné hmotnostiam. (Pre celočíselné vstupy môžete program otestovať v prostredí `R` napríklad pomocou funkcie `knapsack` z knižnice `adagio`.)

4 Vrtuľníky

V dvoch zadaných bodoch (x^A, y^A) a (x^B, y^B) v rovine máme k dispozícii dva vojenské vrtuľníky A, resp. B. Cieľom je, aby A a B ako tandem zlikvidovali n cieľov v bodoch so známymi pozíciami $(x_1, y_1), \ldots, (x_n, y_n)$ a vrátili sa do východzej pozície. (Na zlikvidovanie cieľa v istej pozícii sa musí vrtuľník do tejto pozície dostať.) Napíšte optimalizačný program, ktorého vstupom budú súradnice stanovíšť (x^A, y^A), (x^B, y^B) a pozície cieľov $(x_1, y_1), \ldots, (x_n, y_n)$. Výstupom programu bude zoznam cieľov určených pre vrtuľník A (usporiadaný podľa poradia likvidácie príslušných cieľov vrtuľníkom A) a zoznam cieľov určených pre vrtuľník B (usporiadaný taktiež podľa poradia likvidácie príslušných cieľov vrtuľníkom B). Výstupný plán pre vrtuľníky by mal byť čo najefektívnejší v zmysle minimalizácie $(l_1^p + l_2^p)^{1/p}$, $p > 0$, kde l_1, l_2 sú dĺžky trasy pre prvý, resp. druhý vrtuľník.[5]

Test: Túto úlohu nemám otestovanú; ak si ju vyberiete, idete do neznámeho terénu. Taktiež neviem o tom, že by toto zovšeobecnenie problému obchodného cestujúceho malo svoj názov. Odhadom by ale mal byť Váš program schopný spoľahlivo počítať optimálne trasy pre niekoľko desiatok cieľov v rozmedzí maximálne niekoľkých desiatok sekúnd.

5 Korelácia

Máme výberovú korelačnú maticu M náhodných premenných, pričom chceme zredukovať túto množinu premenných bez veľkej straty informácie. Rozhodli sme sa to riešiť tak, že z daných M premenných vybrieme len m premenných ($m \ll M$), a to tak, aby sme ku každej pôvodnej premennej mali v našom výbere nejakú vysoko korelovanú premennú. Čiže hľadáme takú m-prvkovú podmnožinu A množiny všetkých premenných, aby sme maximalizovali

$$\max_{i \notin A,\, j \in A} |r(X_i, X_j)|,$$

kde $r(X_i, X_j)$ je výberový korelačný koeficient i-tej a j-tej náhodnej premennej.

Test programu: Funkčnosť programu si môžete otestovať napríklad na korelačnej matici veľkosti 4096×4096, získanej z dátového súboru `olivetti_X.csv`. Tieto dáta reprezentujú hodnoty

[5] Interpretácia voliteľného parametra p je podobná ako to bolo pre príklad s dronmi.

úrovní šedej pre 400 fotografií tvárí veľkosti 64×64 pixlov. Zvoľte m v rádoch desiatok, vypočítajte (v zmysle zadania úlohy) optimálny výber m premenných a zobrazte príslušné indexy na obrázku rozmerov 64×64. Zobrazené body by nemali vytvárať príliš veľké "diery", pretože v týchto dierach by sa zrejme nachádzali premenné, ktorú málo korelujú aj s najkorelovanejšou premennou z nájdeného výberu.

6 Štvoruholník

Uvažujme konvexný štvoruholník $ABCD$. Napíšte programy implementujúce dve rôzne metódy,[6] ktorých vstupom sú súradnice bodov A, B, C, D a výstupom sú také súradnice bodu E vo vnútri štvoruholníka $ABCD$, aby plochy trojuholníkov ABE, BCE, CDE, DAE boli "čo najpodobnejšie". Rozumné kritérium podobnosti plôch všetkých štyroch trojuhoníkov zvoľte samostatne, určite však tak, aby optimálne riešenie bolo v bode E, pre ktorý sú všetky štyri plochy rovnaké, pokiaľ taký bod E existuje. Ak sa Vám toto zadanie bude zdať ľahké, môžete sa pokúsiť riešiť analogickú úlohu pre všeobecný konvexný n-uholník. Tento problém je trochu príbuzný skupine problémov označovaných pojmom *fair division problem*.

Test: Program musí nachádzať očividné správne riešenie pre jednoduché štvoruholníky, napríklad pre kosodĺžnik, a to na bežnom počítači v priebehu maximálne niekoľkých sekúnd.

7 Cesty

Máme n miest v rovine, ktoré reprezentujeme bodmi so súradnicami $(x_1, y_1), \ldots, (x_n, y_n)$. Napíšte program, ktorého vstupom budú tieto súradnice miest a výstupom bude mapa najkratšej cestnej komunikácie, ktorá spája každé mesto s každým. Na výpočet najlacnejšej euklidovskej kostry (angl. *minimum spanning tree*) spájajúcej $m \geq n$ uzlových bodov môžete použiť už existujúci program napísaný pre Vaše prostredie. Napríklad pre prostredie `R` môžete použiť funkciu `mst` z knižnice `ape`. Pomôcka: Tento problém je známy pod názvom "problém Steinerovho stromu" (angl. *Steiner tree problem*).

Test: Pre $n = 4$ a $(x_1, y_1) = (0, 0)$, $(x_2, y_2) = (0, 1)$, $(x_3, y_3) = (1, 1)$, $(x_4, y_4) = (1, 0)$ je najkratšia cestná komunikácia spájajúca všetky štyri body zobrazená na druhom obrázku na stránke http://en.wikipedia.org/wiki/Steiner_tree_problem. Celková dĺžka prepojení je v tomto prípade $1 + \sqrt{3}$. Váš program by mal byť schopný na bežnom počítači nachádzať toto optimálne riešenie v rozmedzí maximálne niekoľkých sekúnd. Vo všeobecnosti by mal program nájsť kvalitné riešenie pre $n \leq 10$ v rozmedzí niekoľkých desiatok sekúnd.

8 Kaviareň

Máme obdĺžnikovú kaviareň rozmerov $L_x \times L_y$ metrov, do ktorej chceme umiestniť n okrúhlych stolov s polomerom r. Napíšte program, ktorého vstupom bude L_x, L_y, r, n a výstupom bude

[6] Maximálne jeden z týchto programov môže používať optimalizačnú procedúru, ktorú ste nepísali Vy.

zoznam súradníc stredov stolov, tak, aby vzdialenosť dvoch najbližších stolov bola maximálna možná. Pomôcka: Táto úloha je veľmi príbuzná úlohe, ktorá je známa pod anglickým názvom *circle packing problem*.

Test: Výsledky programu je možné otestovať pomocou hodnôt (a obrázkov) uvedených na stránkach http://en.wikipedia.org/wiki/Circle_packing_in_a_square. Pre $n \leq 4$ by Váš program mal byť schopný na bežnom počítači nachádzať optimálne riešenia v rozmedzí maximálne niekoľkých sekúnd. Výborný výsledok je, ak program nájde do minúty optimálne rozloženie stolov pre štvorec a $n = 8$ (a dostatočne malé r).

9 Kružnice

Napíšte dva programy implementujúce dve principálne rôzne optimalizačné metódy na riešenie nasledovného problému. Vstupom programu sú tri body v rovine (x_i^C, y_i^C), $i = 1, 2, 3$ a tri kladné čísla r_1, r_2, r_3. Výstupom sú tri body (x_i^*, y_i^*), $i = 1, 2, 3$, pričom pre každé i leží bod (x_i^*, y_i^*) na kružnici[7] so stredom v (x_i^C, y_i^C) a polomerom r_i, avšak tak, aby sa minimalizoval obvod trojuholníka s vrcholmi (x_i^*, y_i^*), $i = 1, 2, 3$.

Test: Ide o výrazne nekonvexný problém.[8] Existuje však veľa špeciálnych situácií, kde je riešenie geometricky jasné. Pokúste sa vymyslieť niekoľko takých prípadov a otestovať, či ich Vaše programy nachádzajú. Ak by Vás úloha zaujala, môžete sa pokúsiť ju zovšeobecniť na viacej kružníc; môže ísť o relatívne ťažký mnohorozmerný benchmark pre optimalizačné algoritmy. Výsledok (aj priebežné riešenia) sa dajú pekne zakresliť do obrázka.

10 exGauss

Napíšte dva programy[9] využívajúce dve principálne rôzne optimalizačné metódy na riešenie nasledovného problému. Vstupom je realizácia $x_1, \ldots, x_n$ jednorozmerného náhodného výberu. Výstupom je odhad metódou maximálnej vierohodnosti pre parametre μ, σ^2, λ takzvaného "exponenciálne modifikovaného gaussovského rozdelenia", pozri
https://en.wikipedia.org/wiki/Exponentially_modified_Gaussian_distribution.[10]

Test: Bolo by pekné nájsť reálne dáta, ktoré sa dajú dobre fitovať týmto rozdelením, avšak na otestovanie Vašich procedúr je ideálna simulačná metóda: Rozdelenie exGauss totiž zodpovedá súčtu normálneho a exponenciálneho rozdelenia, takže si z tohto rozdelenia veľmi ľahko nagenerujete umelú sadu dát (s parametrami μ, σ^2, λ, ktoré si zadáte, pričom tieto paramatere by mal Váš program približne lokalizovať len z nasimulovaných dát.) Výsledok sa dá pekne zakresliť do obrázka, napríklad ako hustota určená odhadnutými parametrami preložená cez histogram dát.

[7] Pozor, nie v kruhu, ale na kružnici.

[8] Zaujímavé je ale, že limitný prípad, čiže hľadanie troch bodov A, B, C na priamkach a, b, c, aby bol minimálny obvod trojuholníka ABC, je konvexná úloha.

[9] Maximálne jeden z týchto programov môže používať optimalizačnú procedúru, ktorú ste nepísali Vy.

[10] Všimnite si, že funkcia `erfc` sa dá počítať pomocou distribučnej funkcie rozdelenia $N(0, 1)$.