3. DOMAĆA ZADAĆA – AK. GOD. 2017/18

Domaća zadaća

U okviru ove domaće zadaće potrebno je implementirati sljedeće metode optimiranja:

Postupak najbržeg spusta (gradijentni spust)

Algoritam je objašnjen na predavanjima i u skripti na stranici 4-58. Za funkcije koje će se koristiti u zadacima s ovim postupkom dodatno definirajte analitičko računanje gradijenta. Postupak treba podržavati dva načina rada: u prvom načinu postupak se uvijek pomiče u novu točku za čitav iznos dobivenog pomaka (iznos nenormiranog vektora gradijenta), dok u drugom načinu postupak korištenjem metode zlatnog reza pronalazi optimalan iznos pomaka na pravcu. Postupak je potrebno zaustaviti kada euklidska norma gradijenta postane manja od neke zadane preciznosti. Potrebno je omogućiti da se bez prevođenja programa mogu definirati preciznost *e*, početna točka pretraživanja te hoće li postupak koristiti metodu zlatnog reza za određivanje optimalnog iznosa pomaka ili ne. Pretpostavljena vrijednost preciznosti je 10^{-6}.

Newton-Raphsonov postupak

Algoritam je objašnjen na predavanjima i u skripti na stranici 4-60. Za funkcije koje će se koristiti u zadacima s ovim postupkom dodatno definirajte analitičko računanje gradijenta te Hesseove matrice. Postupak treba podržavati dva načina rada: u prvom načinu postupak se uvijek pomiče u novu točku za čitav iznos dobivenog pomaka, dok u drugom načinu postupak korištenjem metode zlatnog reza pronalazi optimalan iznos pomaka. Postupak je potrebno zaustaviti kada euklidska norma pomaka postane manja od neke zadane preciznosti. Potrebno je omogućiti da se bez prevođenja programa mogu definirati preciznost *e*, početna točka pretraživanja te hoće li postupak koristiti metodu zlatnog reza za određivanje optimalnog iznosa pomaka ili ne. Pretpostavljena vrijednost preciznosti je 10^{-6}. (**napomena**: *iskoristite LUP dekompoziciju iz prve domaće zadaće za rješavanje linearnog sustava koji se javlja u postupku*)

Postupak po Boxu za probleme s ograničenjima

Algoritam je dan na web stranici predmeta: (http://www.fer.hr/_download/repository/box.html) i na predavanjima. Ograničenja koja su dana u zadacima ugradite u program (npr. kao posebni razred). Potrebno je omogućiti da se bez prevođenja programa mogu definirati: preciznost *e*, koeficijent refleksije *alfa* te početna točka pretraživanja (npr. učitavanjem iz datoteke). Pretpostavljene vrijednosti su $e = 10^{-6}$, *alfa* = 1.3. (**napomena**: *Boxov postupak je sličan postupku simpleksa po Nelderu i Meadu, pa probajte iskoristiti određene dijelove toga postupka pri implementaciji*)

Postupak transformacije u problem bez ograničenja na mješoviti način

Algoritam je objašnjen na predavanjima i u skripti na stranici 4-83. Algoritam se mora moći primijeniti na proizvoljnu funkciju i treba podržavati ograničenja jednakosti i nejednakosti. Postupak treba raditi na način da iterativno primjenjuje jedan od postupaka minimizacije bez ograničenja (npr. postupak Hooke-Jeeves ili postupak simpleksa po Nelderu i Meadu). Nakon svake iteracije, parametar transformacije *t* je potrebno povećavati 10 puta i ponovo pokrenuti postupak korištenjem točke minimuma iz prethodne iteracije kao početnu točku nove iteracije. Postupak se ponavlja dok udaljenost točaka dviju uzastopnih iteracija u svakoj od dimenzija nije manja od *epsilon*. Uz parametre konkretnog postupka koji se koristi za rješavanje optimizacijskog problema, potrebno je dodatno specificirati početnu vrijednost parametra transformacije *t*. Definiranje parametara postupaka potrebno je omogućiti bez ponovnog prevođenja programa. Pretpostavljena početna vrijednost parametra *t* je 1 a *epsilon* 10^{-6}. (**napomena**: *kod računanja logaritma, ako ograničenje nije zadovoljeno vratite beskonačno ili neku veliku vrijednost*). Također, implementirajte postupak pronalaženja unutarnje točke (prikazano u skripti na stranici 4-80); ako zadana početna točka ne

zadovoljava ograničenja nejednakosti, primijenite navedenu metodu kako biste odredili unutarnu točku koju ćete potom iskoristiti kao početnu točku pri traženju optimuma.

Napomena: potrebno je osigurati da postupci ne zapnu u beskonačnoj petlji uslijed divergencije, već je u tom slučaju potrebno dojaviti prikladnu poruku o pogrešci. Primjerice, to možete napraviti na način da divergencijom smatrate slučaj kada u 100 uzastopnih iteracija nema poboljšanja u najboljoj dosegnutoj vrijednosti funkcije cilja.

Svi algoritmi moraju podržavati proizvoljno veliku dimenzionalnost rješenja. *Funkciju cilja* potrebno je implementirati tako da vodi evidenciju o broju pozivanja, jer će algoritme biti potrebno usporediti na temelju broja evaluacija (preporučuje se ostvariti kao apstraktni razred). Za postupke koji koriste gradijente potrebno je voditi evidenciju o broju računanja gradijenta i Hesseove matrice.

Za svaki postupak na kraju ispišite točku minimuma koju je postupak pronašao te vrijednost funkcije cilja u toj točki. Također, za svaki postupak ispišite broj poziva funkcije cilja te broj poziva gradijenta i Hesseove matrice ako se koriste u postupku.

Prije dolaska na laboratorijsku vježbu pripremite svaki od zadataka u nastavku u obliku funkcije koju možete pokrenuti, tako da se svaki od zadataka može demonstrirati bez prevelikih promjena u programu. Alternativno, možete koristiti konfiguracijske datoteke kojima specificirate sve bitne parametre, tako da se program može izvršavati za različite postupke bez potrebe za ponovnih prevođenjem.

Funkcije cilja

- $f_1(\mathbf{x}) = 100 \cdot \left(x_2 - x_1^2\right)^2 + (1 - x_1)^2$ (*Rosenbrockova 'banana' funkcija*)
  Početna točka: $\mathbf{x}_0 = (-1.9, 2)$, minimum: $\mathbf{x}_{\min} = (1,1)$, $f_{\min} = 0$
- $f_2(\mathbf{x}) = (x_1 - 4)^2 + 4 \cdot (x_2 - 2)^2$
  Početna točka: $\mathbf{x}_0 = (0.1, 0.3)$, minimum: $\mathbf{x}_{\min} = (4,2)$, $f_{\min} = 0$
- $f_3(\mathbf{x}) = (x_1 - 2)^2 + (x_2 + 3)^2$
  Početna točka: $\mathbf{x}_0 = \mathbf{0}$ (nul vektor), minimum: $\mathbf{x}_{\min} = (2, -3)$, $f_{\min} = 0$
- $f_4(\mathbf{x}) = (x_1 - 3)^2 + (x_2)^2$
  Početna točka: $\mathbf{x}_0 = (0,0)$, minimum: $\mathbf{x}_{\min} = (3,0)$, $f_{\min} = 0$

Laboratorijska vježba

1. Primijenite postupak gradijentnog spusta na funkciju 3, uz i bez određivanja optimalnog iznosa koraka. Što možete zaključiti iz rezultata?
2. Primijenite postupak gradijentnog spusta i Newton-Raphsonov postupak na funkcije 1 i 2 s određivanjem optimalnog iznosa koraka. Kako se Newton-Raphsonov postupak ponaša na ovim funkcijama? Ispišite broj izračuna funkcije, gradijenta i Hesseove matrice.
3. Primijenite postupak po Boxu na funkcije 1 i 2 uz implicitna ograničenja: (x2-x1 >= 0), (2-x1 >= 0) i eksplicitna ograničenja prema kojima su sve varijable u intervalu [-100, 100]. Mijenja li se položaj optimuma uz nametnuta ograničenja?
4. Primijenite postupak transformacije u problem bez ograničenja na funkcije 1 i 2 s ograničenjima iz prethodnog zadatka (zanemarite eksplicitna ograničenja). Novodobiveni problem optimizacije bez ograničenja minimizirajte koristeći postupak Hooke-Jeeves ili postupak simpleksa po Nelderu i Meadu. Može li se uz zadanu početnu točku pronaći optimalno rješenje problema s ograničenjima? Ako ne, probajte odabrati početnu točku iz koje je moguće pronaći rješenje.

5. Za funkciju 4 s ograničenjima (3-x1-x2>=0), (3+1.5*x1-x2>=0) i (x2-1=0) probajte pronaći minimum koristeći postupak transformacije u problem bez ograničenja (također koristite Hooke-Jeeves ili postupak simpleksa po Nelderu i Meadu za minimizaciju). Probajte kao početnu točku postaviti neku točku koja ne zadovoljava ograničenja nejednakosti (primjerice točku (5,5)) te pomoću postupka pronalaženja unutarnje točke odredite drugu točku koja zadovoljava ograničenja nejednakosti te ju iskoristite kao početnu točku za postupak minimizacije.

Demonstracija funkcionalnosti u MATLAB-u

Ovaj dio vježbe izvodi se na predavanjima.

Potrebno je odabrati dvije optimizacijske funkcije, definirati ih kao funkcije u MATLABU i pronaći njihov minimum (bez ograničenja) uporabom ugrađenih MATLABovih funkcija. Neke od funkcija za optimiranje su sljedeće:

- `fminbnd`: minimum funkcije jedne varijable; koristi algoritam zlatnog reza i kvadratne interpolacije
- `fminsearch`: minimum funkcije više varijabli; koristi simplex postupak po Nelderu i Meadu
- `fminunc`: minimum funkcije više varijabli; ova funkcija može, ovisno o proslijeđenim parametrima, uporabiti nekoliko optimizacijskih algoritama, između ostaloga i postupak po Fletcheru i Powellu, poopćeni Newtonov postupak, metodu najbržeg spusta itd.
- `fmincon`: minimum funkcije više varijabli uz ograničenja

Funkcije se u MATLAB-u mogu definirati u posebnim .m datotekama, kao u sljedećem primjeru:

```
function f = apr_primjer(x)  % mora biti u prvom retku!
% Bezvezna funkcija dvije varijable
f = x(1) + x(2);
```

Datoteku je potrebno nazvati istim imenom kao i funkcija (apr_primjer.m). Iz MATLAB-a se funkciji tada može pristupiti s `apr_primjer([1,1])`.

Odabrana poglavlja iz MATLAB Helpa:

- DEMOS: Optimization: Minimization of the "banana function"
- MATLAB: Mathematics: Function Functions: Minimizing Functions and Finding Zeros
- Optimization Toolbox: Standard Algorithms: Unconstrained Optimization