Octave-програма dist2p для розділення двох поліедрів

Стецюк П.І., Стовба В.О., Мартинюк І.С.
stetsyukp@gmail.com

Інститут кібернетики ім. В.М. Глушкова НАН України

XIII Міжнародна науково-практична конференція "Теоретичні та прикладні аспекти побудови програмних систем" (TAAPSD'2016), 5 – 9 грудня 2016 року, м. Київ, Україна

План доповіді (Outline)

Outline

Два поліедри

Поліедру $P_1 = \{x \in R^n : \ A_1 x \leq b_1\}$ відповідають $n \times m$-матриця $A_1 = \{a_{ij}^1\}$ та m-вектор $b_1 = \{b_j^1\}$,

Поліедру $P_2 = \{x \in R^n : \ A_2 x \leq b_2\}$ відповідають $n \times k$-матриця $A_2 = \{a_{ij}^2\}$ та k-вектор $b_2 = \{b_j^2\}$.

Задача знаходження мінімальної відстані

між поліедрами P_1 та P_2 може бути представлена

задачею квадратичного програмування:

$$f^* = f(x^*, y^*) = \min_{x,y \in R^n} \{f(x,y) = \|x - y\|^2\} \quad (1)$$

при обмеженнях

$$A_1 x \leq b_1, \quad (2)$$

$$A_2 y \leq b_2, \quad (3)$$

де A_1 – $n \times m$-матриця, b_1 – m-вектор,
A_2 – $n \times k$-матриця, b_2 – k-вектор.

Властивості задачі

Якщо виконується умова Слейтера, (x^*, y^*) – оптимальний розв'язок задачі (1) – (3), не обов'язково єдиний,

тоді

- якщо поліедри розділяються, то мінімальна відстань між ними дорівнює $\rho^* = \sqrt{f^*} = \|x^* - y^*\| > 0$;
- якщо не розділяються – $\rho^* = 0$, а $y^* = x^*$.

Outline

Позначення

Вектори $u^* = \{u_1^*, ..., u_m^*\}$ та $v^* = \{v_1^*, ..., v_k^*\}$ містять оптимальні множники Лагранжа для обмежень (2) та (3).

Вектор $P = \{P_1, P_2\}$ містить два штрафних коефіцієнти.

Теорема 1 ([1], Стецюк, 2014)

Якщо $P_1 > \max\{u_1^*, ..., u_m^*\}$ *і* $P_2 > \max\{v_1^*, ..., v_k^*\}$, *то задача мінімізації негладкої опуклої функції*

$$F_P(x,y) = \sum_{i=1}^{n}(x_i - y_i)^2 + P_1 \sum_{i=1}^{m} \max\{0, \sum_{j=1}^{n} a_{ij}^{(1)} x_j - b_i^{(1)}\} +$$

$$+ P_2 \sum_{i=1}^{k} \max\{0, \sum_{j=1}^{n} a_{ij}^{(2)} y_j - b_i^{(2)}\} \qquad (4)$$

еквівалентна задачі (1)–(3) *в тому сенсі, що множина мінімумів функції* $F_P(x,y)$ *співпадає з множиною оптимальних розв'язків задачі* (1)–(3).

Про доведення теореми 1

Теорема 1 випливає з теореми 4.2 ([2], Шор, 1979), якщо вибрати найпростіший варіант негладкої функції штрафу

$$p_i(t) = \begin{cases} 0, \text{ якщо } t \leq 0 \\ c_i t, \text{ якщо } t > 0 \end{cases} \quad i = 1, ..., m + k,$$

а коефіцієнти c_i апроксимувати зверху максимальними значеннями оптимальних множників Лагранжа для обмежень (2) та (3), відповідно.

Що дає теорема 1?

Теорема 1 дозволяє будувати різноманітні алгоритми для розв'язання задачі розділення двох поліедрів.

Алгоритм буде визначатися методом мінімізації негладкої опуклої функції $F_P(x, y)$ вигляду (4).

Outline

Про програму *dist2p*

Програма **dist2p** (the distance between two polyhedra) написана на некомерційній мові Octave.

В її основу покладено octave-програму ralgb5 [3, с. 384-385], яка реалізує модифікацію r-алгоритму Шора з постійним коефіцієнтом розтягу простору α та адаптивним способом регулюванням кроку в напрямі нормованого антисубградієнта в перетвореному просторі змінних.

Код програми *dist2p* (коментарі)

```
function [fr,xr,yr,ist,itn,ncalls]=dist2p(A1,b1,A2,b2,P1,P2,
                                          alpha,h0,q1,epsg,epsz,maxitn);
#  Вхідні параметри:
#     A1,b1,A2,b2 - дані для поліедрів
#     P1,P2 - коефіцієнти для штрафної негладкої функції
#     alpha,h0,q1 - управляючі параметри r-алгоритму
#     epsg,epsz,maxitn - параметри зупинки методу
#  Вихідні параметри:
#     fr,xr,yr - значення функції f(xr,yr) в точці (xr,yr)
#     ist - код зупинки (2=epsg, 3=epsz, 4=maxitn)
#     itn - кількість ітерацій
#     ncalls - кількість обчислень значення функції та її субградієнта
itn=0; hs=h0; n=columns(A1); B=eye(2*n); xr=yr=zeros(n,1); z=[xr;yr];
```

Код програми *dist2p* (без коментарів)

```
function [fr,xr,yr,ist,itn,ncalls]=dist2p(A1,b1,A2,b2,P1,P2,
                                          alpha,h0,q1,epsg,epsz,maxitn);
itn=0; hs=h0; n=columns(A1); B=eye(2*n); xr=yr=zeros(n,1); z=[xr;yr];
ncalls = 1; b1p=-b1>0; b2p=-b2>0; fr=P1*sum(b1p)+P2*sum(b1p);
g0 = [P1*(sum(diag(sign(b1p))*A1))'; P2*(sum(diag(sign(b2p))*A2))'];
printf("itn %4d f %14.6e fr %14.6e ls %2d ncalls %4d\n",itn,fr,fr,0,ncalls);
if(norm(g0) < epsg) ist = 2;  return; endif
for (itn = 1:maxitn)
   dz = B * (g1 = B' * g0)/norm(g1);
   d = 1; ls = 0; ddz = 0;
   while (d > 0)
      z -= hs * dz; ddz += hs * norm(dz);
      ncalls ++; x=z(1:n,1); y=z(n+1:2*n,1); b1p=A1*x-b1>0; b2p=A2*y-b2>0;
      f=(x-y)'*(x-y)+P1*sum(b1p)+P2*sum(b2p); b1p=sign(b1p); b2p=sign(b2p);
      g1=[2*(x-y)+P1*(sum(diag(b1p)*A1))';2*(y-x)+P2*(sum(diag(b2p)*A2))'];
      if (f < fr)  fr = f; xr = x; yr = y; endif
      if(norm(g1) < epsg) ist = 2;  return; endif
      ls ++; (mod(ls,3)==0) && (hs *= 1.1);
      if(ls > 500) istop = 5;  return; endif
      d = dz' * g1;
   endwhile
   (ls == 1) && (hs *= q1);
   printf("itn %4d f %14.6e fr %14.6e ls %2d ncalls %4d\n",itn,f,fr,ls,ncalls);
   if(ddz < epsz) ist = 3;  return; endif
   xi = (dg = B' * (g1 - g0) )/norm(dg);
   B += (1 / alpha - 1) * B * xi * xi';
   g0 = g1;
endfor
ist = 4;
endfunction
```

Характеристика програми *dist2p*

Для адаптації кроку в програмі **dist2p** використовуються тільки параметри h_0 та q_1, а параметри $q_2 = 1.1$ та $n_h = 3$ вбудовані в тіло програми. Коефіцієнт розтягу простору та параметри регулювання кроку рекомендується вибирати наступними: $\alpha = 2 \div 4$, $h_0 = 1.0$, $q_1 = 0.95 \div 1.0$.

Параметри зупинки $r(\alpha)$-алгоритму в програмі **dist2p** такі ж самі, як і в програмі ralgb5: тобто $\varepsilon_z = \varepsilon_{(x,y)}$, ε_g та $maxitn$. Виходом із програми **dist2p** є рекордні значення f_r, x_r, y_r та код зупинки ist, який вказує по якому із критеріїв програма закінчила свою роботу.

Outline

Тестова задача $(n \geq 2)$

Розглянемо задачу (1)–(3), коли поліедри є симплексами

$$P_1 = \Delta_1 = \left\{ x \in R^n : \sum_{i=1}^{n} x_i \leq n+1,\ x_i \geq 1,\ i=1,\ldots,n \right\}$$

та

$$P_2 = \Delta_2 = \left\{ y \in R^n : \sum_{i=1}^{n} y_i \geq -n-1,\ y_i \leq -1,\ i=1,\ldots,n \right\}.$$

Аналітичний розв'язок задачі:

$$\rho^* = 2\sqrt{n}, \quad x^* = (1,...,1)^T, \quad y^* = (-1,...,-1)^T.$$

Про тестові експерименти

Досліджується величина $dist = \sqrt{\|x_r - x^*\|^2 + \|y_r - y^*\|^2}$: наскільки знайдене програмою **dist2p** наближення (x_r, y_r) відрізняється від точного розв'язку (x^*, y^*).

При $n = 5 \div 50$ для трьох значень $\varepsilon_z = \{10^{-6}, 10^{-7}, 10^{-8}\}$ у таблиці наведені **itnr** – кількість ітерацій та **nfg** – кількість обчислень значення функції $F_P(x, y)$ та її субградієнта.

Результати тестування програми dist2p

n	ε_z = 1.00e-06			ε_z = 1.00e-007			ε_z = 1.00e-08		
	itnr	nfg	*dist*	itnr	nfg	*dist*	itnr	nfg	*dist*
5	86	126	1.35e-06	100	141	9.38e-08	113	159	1.21
10	150	222	7.05e-06	178	267	9.65e-07	204	313	7.40
15	209	323	2.28e-05	260	426	1.29e-06	292	475	4.98
20	260	418	2.37e-05	315	519	2.19e-06	377	646	1.67
25	238	341	3.42e-05	359	596	2.14e-06	417	703	1.96
30	224	315	5.53e-05	475	863	1.81e-06	526	935	3.52
35	380	705	2.57e-05	573	1074	8.01e-07	616	1134	1.85
40	556	1025	1.17e-05	608	1107	2.13e-06	681	1222	1.35
45	456	785	3.48e-05	681	1236	1.46e-06	764	1369	1.40
50	369	537	6.82e-05	415	598	6.82e-05	632	1046	4.80

Про параметри тестування

При розрахунках були використані такі параметри:
$P_1 = P_2 = 100$, $\alpha = 2$, $h_0 = 1.0$, $q_1 = 0.95$.

Обчислення проводились

на комп'ютері Pentium 3GHz у системі Windows7/32
з використанням GNU Octave версії 3.6.4.

Список використаних джерел

- Стецюк П.І. Субградієнтний алгоритм з розтягом простору для задачі розділення двох поліедрів // Праці VII-ої міжнародної школи-семінару "Теорія прийняття рішень", Ужгород, 29 вересня-4 жовтня 2014 р. – Ужгород: УжНУ, 2014. – С. 244–245.
- Шор Н.З. Методы минимизации недифференцируемых функций и их приложения. – К.: Наук. думка, 1979. – 199 с.
- Стецюк П.И. Методы эллипсоидов и r-алгоритмы. – Кишинэу: Эврика, 2014. – 488 с.

Запитання?

ДЯКУЮ ЗА УВАГУ!