УДК 512.36 + 519.3

Л.М. БРЭГМАН

АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ ВЫПУКЛОГО ПРОГРАММИРОВАНИЯ МЕТОДОМ РЕЛАКСАЦИИ

Рассмотрим следующую задачу: минимизировать

$$\sum_{j=1}^{n} (c_j x_j + \varepsilon x_j \ln x_j) \tag{I}$$

при ограничениях

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \quad (i=1,2,\ldots,m), \tag{2}$$

$$x_j \geqslant 0 . \tag{3}$$

Здесь ε - заданное положительное число. Функция (I), очевидно, является выпуклой, поэтому для решения задачи (I) - (3) можно воспользоваться методом релаксации [I] . При этом ограничения (3) можно опустить, так как алгоритм не выводит точку из положительного ортанта.

При решении задачи (I) - (3) методом релаксации строится последовательность $\{x^{t,k}\}$ следующим образом:

1. В качестве $x^{0,m}$ выбирается вектор, доставляющий абсолютный минимум функции (I), т.е.

$$x_j^{0,m} = \exp(-c_j/\varepsilon - 1) .$$

2. Если известен вектор $x^{t,k}$ и $k<m$, то вектор $x^{t,k+1}$ строится следующим образом:

$$x_j^{t,k+1} = x_j^{t,k} \exp(\lambda a_{k+1,j}) , \qquad (4)$$

где λ - единственный корень уравнения :

$$\sum_{j=1}^{n} a_{k+1,j} x_j^{t,k} \exp(\lambda a_{k+1,j}) = b_{k+1} . \qquad (5)$$

Если $k = m$,то вектор $x^{t+1,1}$ строится следующим образом:

$$x_j^{t+1,1} = x_j^{t,m} \exp(\lambda a_{1j}) ,$$

где λ - единственный корень уравнения :

$$\sum_{j=1}^{n} a_{1j} x_j^{t,m} \exp(\lambda a_{1j}) = b_1 . \qquad (5')$$

Последовательность $\{x^{t,k}\}$ сходится к вектору x^* , который является решением задачи (I) - (3).

Для решения описанного процесса составлена программа на языке АЛГОЛ-60. В программе уравнения (5) и $(5')$ решаются методом Ньютона (процедура newt) с начальным приближением $\lambda = 0$.

Левые части уравнений (5) и $(5')$ (как функции от λ) и их производные вычисляются с помощью процедуры $f\ (i, lam, z)$, где формальному параметру i соответствует номер уравнения (5), формальному параметру lam - значение переменной λ , а параметру z - логическое значение, причём если это значение true , то вычисляется сама функция, а если это значение false , то её производная .

Итерации по методу Ньютона продолжаются до тех пор, пока два последовательных приближения не будут отличаться друг от друга меньше, чем на величину delta.

Информация о матрице коэффициентов a_{ij} , вводится с помощью трех массивов:

I. Массив a $[1:p]$ (p - число, не меньшее, чем количество ненулевых a_{ij}). Элементами этого массива является отличные от нуля числа a_{ij} , записанные по строкам.

2. Массив col[I:p] . Элементами этого массива являются номера столбцов, в которых находятся отличные от нуля числа a_{ij} (т.е. col[j] является номером столбца, в котором стоит элемент a[j]).

3. Массив adr[0:m]. adr[i] – номер в массиве a числа a_{ij} ,являющегося последним ненулевым элементом i-той строки. adr[0] принимается равным нулю.

Кроме этих трех массивов,вводятся числа b_i (массив b[I:m]) и числа c_j (массив x[I:n]).

Задача считается решенной,если на какой-либо итерации корни уравнений (5) и (5′) по абсолютной величине не будут превосходить заданного числа eps I.

На языке АЛГОЛ-60 алгоритм записывается следующим образом:

```
begin integer m,n,p; ввод (m,n,p);
  begin integer u,j,i; real cor,eps,epsI,delta;
    array a[I:p], b[I:m], x[I:n];
    integer array col[I:p], adr[0:m];
    real procedure f(i,lam,z);
    value i,lam,z;
    integer i; real lam; Boolean z;
      begin integer j; real s;
        s:=0;
        for j:=adr[i-I]+I step I until adr[i] do
        s:=s+exp (lam×a[j]) × x[col[j]] ×
        (if z then a[j] else a[j]↑2);
        f:=if z then s-b[i] else s
      end;
     procedure newt (lam0,lam,i,f,delta);
     value i,lam0,delta;
     integer i; real lam0,lam,delta;
      real procedure f;
        begin real s;
          mk1: s:=f(i,lam0,true)/f(i,lam0,false);
```

```
        if abs (s)<delta then lam:=lam0 else
          begin lam0:=lam0-s; go to mk1
           end
        end;
    ввод (a,b,x,col,adr,eps,eps1,delta);
    for j:=I step I until n do
    x[j]:=exp(-x[j]/ε-1);
    mk2: u:=0;
    for i:=I step I until m do
      begin newt (0,cor,i,f,delta);
       if abs (cor)>eps I then
        begin u:=1;
          for j:=adr[i-I]+I step 1 until adr[i] do
          x[col[j]]×exp(cor×a[j])
        end
      end;
   if u=I then go to mk2;
    вывод (x);
   for i:=I step I until m do
   b[i]:=f(i,0,true); вывод (b);
   stop
  end
end
```

Литература

I. Л.М. Брэгман. "Релаксационный метод нахождения общей точки выпуклых множеств и его применение для решения задач выпуклого программирования". Ж. выч. математ. и мат. физики, 1967, т.7, № 3.

Поступила в редакцию
15.XI. 1967 г.