РЕШЕНИЕ ЗАДАЧ КВАДРАТИЧНОГО ПРОГРАММИРОВАНИЯ В СРЕДЕ MATLAB*

Н. А. Соловьёва — vinyo@mail.ru

Е. К. Чернэуцану — katerinache@yandex.ru

12 февраля 2011 г.

В среде MATLAB задачи квадратичного программирования решаются с помощью функции `quadprog`. Доклад посвящён краткому описанию её возможностей.

1°. Функция `quadprog` решает задачу квадратичного программирования в форме

$$\begin{aligned} \tfrac{1}{2}\,\mathtt{x}^T\cdot\,\mathtt{H}\cdot\mathtt{x} + \mathtt{f}^T\cdot\,\mathtt{x} &\to \inf,\\ \mathtt{A}\cdot\mathtt{x} &\leqslant \mathtt{b},\\ \mathtt{Aeq}\cdot\mathtt{x} &= \mathtt{beq},\\ \mathtt{lb} \leqslant \mathtt{x} &\leqslant \mathtt{ub}. \end{aligned} \tag{1}$$

Основными входными параметрами `quadprog` являются: матрица `H` и вектор `f` из целевой функции, матрица ограничений-неравенств `A`, вектор правых частей ограничений-неравенств `b`, матрица ограничений-равенств `Aeq`, вектор правых частей ограничений-равенств `beq`, вектор `lb`, ограничивающий план `x` снизу, вектор `ub`, ограничивающий план `x` сверху. На выходе функция `quadprog` выдаёт оптимальный план `x` задачи (1) и экстремальное значение целевой функции `fval`.

Если матрица `H` несимметрична, то MATLAB заменяет её на $(\mathtt{H}+\mathtt{H}^T)/2$. (При этом значение целевой функции не меняется.)

*Семинар по дискретному гармоническому анализу и геометрическому моделированию «DHA & CAGD»: `http://www.dha.spb.ru/`

ПРИМЕР 1. Решим в MATLAB задачу квадратичного программирования

$$
\begin{aligned}
f(x) = x_1^2 + x_2^2 + x_3^2 - 2x_1x_2 - x_1 - x_2 + x_3 \to \inf,\\
x_1 + x_2 + x_3 \geqslant 1,\\
2x_1 + x_2 - x_3 \geqslant -1,\\
x_1 - x_2 + x_3 = 0,\\
0 \leqslant x_1 \leqslant 1,\\
0 \leqslant x_2 \leqslant 1,\\
0 \leqslant x_3 \leqslant 1.
\end{aligned}
$$

Соответствующая программа (m-файл)* выглядит так:

```
clear all
close all
clc % удаляются все текущие переменные из памяти MATLAB,
    закрываются все графические окна, очищается экран консоли
D = [2 -2 0; -2 2 0; 0 0 2]; % строки матрицы разделяются
    точкой с запятой
C = [-1 -1 1]; % задаётся вектор длины три
G = [1 1 1; 2 1 -1];
B = [1 -1];
Aeq = [1 -1 1];
beq = [0];
lb = zeros(3,1); % задаётся нулевой вектор длины три
ub = [1 1 1];
H = D; % коэффициенты квадратичной части целевой функции
f = C; % коэффициенты линейной части целевой функции
A = -G;
b = -B; % появляются знаки «-», так как ограничения-неравенства
    Gx ⩾ B приводятся к виду −Gx ⩽ −B
[x,fval] = quadprog(H,f,A,b,Aeq,beq,lb,ub);
x
fval
```

Запустив программу, получим сообщение

```
Warning: Large-scale algorithm does not currently solve this
   problem formulation, using medium-scale algorithm instead.
Optimization terminated.
```

*Для отладки приведённых в докладе программ использовался MATLAB 7.11.0 (R2010b).

```
x =
  1.0000
  1.0000
  0.0000
fval =
  -2.0000
```

Предупреждение, содержащееся в первых двух строках, прокомментируем позже.

Дополнительно можно задать начальное приближение `x0`:

```
[x,fval] = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0).
```

Если какой-то из входных параметров отсутствует, на его место следует поставить квадратные скобки `[]`, за исключением случая, когда это последний параметр в списке. Например, если нужно решить задачу без ограничений-равенств, в которой не задано начальное приближение, то оператор вызова функции `quadprog` будет выглядеть так:

```
[x,fval] = quadprog(H,f,A,b,[],[],lb,ub).
```

(Квадратные скобки в конце списка, соответствующие начальному приближению, не ставятся.)

С помощью входного параметра `options` устанавливаются некоторые дополнительные настройки, в частности, выбирается алгоритм решения. MATLAB решает задачи квадратичного программирования двумя способами: алгоритмом внутренней точки (`Large-Scale Algorithm`) и методом перебора граней (`Medium-Scale Algorithm`). По умолчанию используется алгоритм внутренней точки. Чтобы выбрать метод перебора граней, нужно написать

```
options = optimset('LargeScale','off');
[x,fval] = quadprog(H,f,A,b,Aeq,beq,lb,ub,[],options).
```

Заметим, что MATLAB использует алгоритм внутренней точки для задач двух типов: если есть только ограничения-равенства и матрица ограничений-равенств имеет полный ранг или если есть только нижние `lb` и верхние `ub` границы. Задачу, которая не относится к этим двум типам, MATLAB решает методом перебора граней, выдавая предупреждение (см. пример 1)

```
Warning: Large-scale algorithm does not currently solve this
   problem formulation, using medium-scale algorithm instead.
```

Разберёмся с выходными данными. MATLAB позволяет выводить информацию о том, как завершилось решение задачи. За это отвечает параметр `exitflag`. Если значение `exitflag` равно 1, то найдено решение задачи, если равно 0, то превышено допустимое число итераций, если равно -2 — множество планов задачи пусто, если равно -3 — целевая функция не ограничена снизу на множестве планов. Интерпретация других значений параметра `exitflag` приведена в MATLAB Help. Для метода перебора граней допустимое число итераций (`MaxIter`) по умолчанию равно 200. Значение `MaxIter` можно изменить. Чтобы установить допустимое число итераций равным, к примеру, 10, нужно написать

```
options =
optimset('LargeScale','off','MaxIter',10);
[x,fval] = quadprog(H,f,A,b,Aeq,beq,lb,ub,[],options).
```

Если после выполнения десятой итерации решение не будет найдено, параметр `exitflag` станет нулевым и на экране появится сообщение

```
Maximum number of iterations exceeded;
increase options.MaxIter.
```

Параметр `output` содержит информацию о процессе оптимизации, в частности, число итераций (`iterations`) и используемый алгоритм (`algorithm`). Другие поля параметра `output` описаны в MATLAB Help. Запустим с данными из примера 1 следующую программу:

```
options = optimset('LargeScale','off');
[x,fval,exitflag,output] =
quadprog(H,f,A,b,Aeq,beq,lb,ub,[],options);
exitflag
output.iterations
output.algorithm
```

На выходе получим:

```
Optimization terminated.
exitflag =
  1
ans =
  3
ans =
  medium scale: active-set
```

Это означает, что метод перебора граней успешно завершил работу. Для нахождения решения потребовалось три итерации.

2°. При решении задачи квадратичного программирования возможны три выхода из процесса: найдено решение задачи, множество планов пусто, целевая функция не ограничена снизу на множестве планов. Продемонстрируем эти варианты на примерах.

ПРИМЕР 2. Решим в MATLAB задачу квадратичного программирования

$$\begin{aligned} Q(x) = {} & x_1^2 + x_1 + x_2 \to \inf, \\ & x_1 - x_2 = 2, \\ & x_1 \geqslant 0,\ x_2 \geqslant 0. \end{aligned} \tag{2}$$

Соответствующая программа будет выглядеть так:

```
clear all
close all
clc
D = [2 0; 0 0];
C = [1 1];
Aeq = [1 -1];
beq = [2];
lb = zeros(2,1);
H = D;
f = C;
options = optimset('LargeScale','off');
[x,fval,exitflag] =
quadprog(H,f,[],[],Aeq,beq,lb,[],[],options);
x
fval
exitflag
```

В результате работы программы получим:

```
Optimization terminated.
x =
  2
  0
fval =
  6
exitflag =
  1
```

Найдено решение задачи (2).

ПРИМЕР 3. Решим в MATLAB задачу квадратичного программирования

$$\begin{aligned} f(x) = {} & x_1^2 + x_2^2 \to \inf, \\ & x_1 + x_2 = 4, \\ & x_1 + 2x_2 \geqslant 10, \\ & x_1 \geqslant 0,\ x_2 \geqslant 0. \end{aligned} \tag{3}$$

Приведём результат работы соответствующей программы:

```
Exiting: The constraints are overly stringent; no feasible
starting point found.
x =
  -0.6180
  4.6180
fval =
  21.7082
exitflag =
  -2
```

Множество планов задачи (3) пусто.

ПРИМЕР 4. Решим в MATLAB задачу квадратичного программирования

$$\begin{aligned} f(x) = x_1^2 - x_1 - 4x_2 &\to \inf, \\ x_1 + x_2 &\geqslant 2, \\ x_1 \geqslant 0,\ x_2 &\geqslant 0. \end{aligned} \tag{4}$$

Запустив программу, решающую задачу (4), получим:

```
Exiting: The solution is unbounded and at infinity;
the constraints are not restrictive enough.
x =
  1.0e+016*
  0
  4.0000
fval =
  -1.6000e+017
exitflag =
  -3
```

Целевая функция задачи (4) не ограничена снизу на множестве планов.

3°. Полное описание функции `quadprog` приведено в MATLAB Help.