Matlab 在最优化问题中的应用

优化理论是一门实践性很强的学科，广泛应用于生产管理、军事指挥和科学试验等各种领域，Matlab 优化工具箱提供了对各种优化问题的一个完整的解决方案。

在数学上，所谓**优化问题**，就是求解如下形式的**最优解**：

Min fun (x)
Sub. to [C.E.]
[B.C.]

其中 fun (x)称为**目标函数**，“Sub. to”为“subject to”的缩写，由其引导的部分称为**约束条件**。[C.E.]表示 Condition Equations，即**条件方程**，可为等式方程，也可为不等式方程。[B.C.]表示 Boundary Conditions，即**边界条件**，用来约束自变量的求解域，以 lb≤x≤ub 的形式给出。当[C.E.]为空时，此优化问题称为自由优化或无约束优化问题；当[C.E.]不空时，称为有约束优化或强约束优化问题。

在优化问题中，根据变量、目标函数和约束函数的不同，可以将问题大致分为：

- **线性优化** 目标函数和约束函数均为线性函数。
- **二次优化** 目标函数为二次函数，而约束条件为线性方程。

线性优化和二次优化统称为**简单优化**。

- **非线性优化** 目标函数为非二次的非线性函数，或约束条件为非线性方程。
- **多目标优化** 目标函数并非一个时，称为多目标优化问题。

本章将对以上几类优化问题在 Matlab 中的实现作比较详细的讲解。另外还将介绍两个利用优化方法解非线性方程的函数。

通过本章的介绍，用户可以不必掌握艰涩的各种优化算法而轻易地解决一些常用的最优化问题了。

3.1 线性规划问题

线性规划问题即**目标函数和约束条件**均为线性函数的问题。

其标准形式为：

min C’x
sub. To Ax = b
x≥0

其中 C, b, 0∈R^n，A∈$R^{m\times n}$，均为数值矩阵，x∈R^n。

若目标函数为：**max C’x，则转换成：min –C’x**。

标准形式的线性规划问题简称为 LP（Linear Programming）问题。其它形式的线性规划问题经过适当的变换均可以化为此种标准形。线性规划问题虽然简单，但在工农业及其他生产部门中应用十分广泛。

在 Matlab 中，线性规划问题由 **linprog** 函数求解。

函数：linprog %求解如下形式的线性规划问题：

$$\min_x f^T x$$

such that $A\cdot x \le b$

$Aeq\cdot x = beq$

$lb \le x \le ub$

其中 *f, x, b, beq, lb, ub* 为向量，*A, Aeq* 为矩阵。

格式：x = linprog(f,A,b)
x = linprog(f,A,b,Aeq,beq)
x = linprog(f,A,b,Aeq,beq,lb,ub)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0,options)

[x,fval] = linprog(...)
[x,fval,exitflag] = linprog(...)
[x,fval,exitflag,output] = linprog(...)
[x,fval,exitflag,output,lambda] = linprog(...)

说明：

x = linprog(f,A,b) 求解问题 min f'*x，约束条件为 A*x<=b。**不等式约束**

x = linprog(f,A,b,Aeq,beq) 求解上面的问题，但增加**等式约束**，即 Aeq*x = beq。若没有不等式存在，则令 A = [ ]、b = [ ]。

x = linprog(f,A,b,Aeq,beq,lb,ub) 定义设计变量 x 的**下界 lb 和上界 ub**，使得 x 始终在该范围内。若没有等式约束，令 Aeq = [ ]、beq = [ ]。

x = linprog(f,A,b,Aeq,beq,lb,ub,x0) 设置**初值为 x0**。该选项只适用于中型问题，默认时大型算法将忽略初值。

x = linprog(f,A,b,Aeq,beq,lb,ub,x0,options) 用 options 指定的优化参数进行最小化。

[x,fval] = linprog(...) 返回**解 x** 处的**目标函数值 fval**。

[x,fval,exitflag] = linprog(...) 返回 exitflag 值，描述函数计算的退出条件。

[x,fval,exitflag,output] = linprog(...) 返回包含优化信息的输出变量 output。

[x,fval,exitflag,output,lambda] = linprog(...) 将解 x 处的 Lagrange 乘子返回到 lambda 参数中。

exitflag 参数

描述退出条件：

- >0 表示目标函数**收敛**于解 x 处；
- =0 表示已经**达到函数评价或迭代的最大次数**；
- <0 表示目标函数**不收敛**。

output 参数

该参数包含下列优化信息：

- output .iterations 迭代次数；
- output .cgiterations PCG 迭代次数(只适用于大型规划问题)；
- output .algorithm 所采用的算法。

lambda 参数

该参数是解 x 处的 Lagrange 乘子。它有以下一些属性：

- lambda.lower—lambda 的下界；
- lambda.upper—lambda 的上界；
- lambda.ineqlin—lambda 的线性不等式；
- lambda.eqlin—lambda 的线性等式。

例 3-1 求解下列优化问题：

$$\begin{aligned} \min \quad & f(x) = -5x_1 - 4x_2 - 6x_3 \\ \text{sub.to} \quad & x_1 - x_2 + x_3 \le 20 \\ & 3x_1 + 2x_2 + 4x_3 \le 42 \\ & 3x_1 + 2x_2 \le 30 \\ & 0 \le x_1, \quad 0 \le x_2, \quad 0 \le x_3 \end{aligned}$$

解：在 Matlab 命令窗口键入：

```
f=[-5;-4;-6];
A=[1 -1 1;3 2 4;3 2 0];
b=[20;42;30];
lb=zeros(3,1);
[x,fval,exitflag,output,lambda]=linprog(f,A,b,[],[],lb)
Optimization terminated.
x =
     0.0000
    15.0000
     3.0000
fval =
```

```
   -78.0000
exitflag =
     1
output =
            iterations: 6
             algorithm: 'interior-point-legacy'
          cgiterations: 0
               message: 'Optimization terminated.'
       constrviolation: 0
         firstorderopt: 5.8705e-10
lambda =
       ineqlin: [3x1 double]
         eqlin: [0x1 double]
         upper: [3x1 double]
         lower: [3x1 double]
>> lambda.ineqlin
ans =
       0.0000
       1.5000
       0.5000
>> lambda.lower
ans =
       1.0000
       0.0000
       0.0000
```

lambda 向量中的非零元素表示哪些约束是主动约束。本例中，第 2 个和第 3 个不等式约束，第 1 个下界约束是主动约束（如这些解位于约束边界上）。exitflag = 1 表示过程正常收敛于解 x 处。

例 3-2 生产决策问题。

某厂生产甲乙两种产品，已知制成一吨产品甲需资源 A 3 吨，资源 B 4m^3；制成一吨产品乙需资源 A 2 吨，资源 B 6 m^3；资源 C 7 个单位。若一吨产品甲和乙的经济价值分别为 7 万元和 5 万元，三种资源的限制量分别为 90 吨、200 m^3 和 210 个单位，试决定应生产这两种产品各多少吨才能使创造的总经济价值最高？

解：令生产产品甲的数量为 x_1，生产产品乙的数量为 x_2。由题意可以建立下面的数学模型：

$$\begin{aligned} \max \quad & z = 7x_1 + 5x_2 \\ \text{sub. to} \quad & 3x_1 + 2x_2 \le 90 \\ & 4x_1 + 6x_2 \le 200 \\ & 7x_2 \le 210 \\ & x_1 \ge 0, \quad x_2 \ge 0 \end{aligned}$$

该模型中要求目标函数最大化，需要按照 Matlab 的要求进行转换，即目标函数为

$$\min \quad z = -7x_1 - 5x_2$$

在 Matlab 中实现：

```
f=[-7;-5];
A=[3 2;4 6;0 7];
b=[90;200;210];
lb=[0;0];
[x,fval,exitflag,output,lambda]=linprog(f,A,b,[],[],lb)
Optimization terminated.
x =
    14.0000
    24.0000
fval =
-218.0000
```

```
exitflag =
     1
output =
          iterations: 5
           algorithm: 'interior-point-legacy'
        cgiterations: 0
             message: 'Optimization terminated.'
    constrviolation: 0
      firstorderopt: 4.3909e-07
lambda =
    ineqlin: [3x1 double]
      eqlin: [0x1 double]
      upper: [2x1 double]
      lower: [2x1 double]
```

由上可知，生产甲种产品 14 吨、乙种产品 24 吨可使创造的总经济价值最高为 218 万元。exitflag = 1 表示过程正常收敛于解 x 处。

例 3-3 厂址选择问题。

考虑 A、B、C 三地，每地都出产一定数量的原料也消耗一定数量的产品（见下表）。已知制成每吨产品需 3 吨原料，各地之间的距离为：A—B：150km，A—C：100km，B—C：200km。假定每万吨原料运输 1km 的运价是 5000 元，每万吨产品运输 1km 的运价是 6000 元。由于地区条件的差异，在不同地点设厂的生产费用也不同。问究竟在哪些地方设厂，规模多大，才能使总费用最小？另外，由于其它条件限制，在 B 处建厂的规模（生产的产品数量）不能超过 5 万吨。

A、B、C 三地出产原料、消耗产品情况表

地点	年产原料（万吨）	年销产品（万吨）	生产费用（万元/万吨）
A	20	7	150
B	16	13	120
C	24	0	100

解：令 x_{ij} 为由 i 地运到 j 地的原料数量（万吨），y_{ij} 为由 i 地生产后运到 j 地的产品数量（万吨），i，$j=1$，2，3（分别对应 A、B、C 三地）。根据题意，可以建立问题的数学模型（其中目标函数包括原料运输费、产品运输费和生产费用（万元））：

$$\begin{aligned}
\min\quad & z = 75x_{12}+75x_{21}+50x_{13}+50x_{31}+100x_{23}+100x_{32}+150y_{11}+240y_{12}+210y_{21} \\
& \quad +120y_{22}+160y_{31}+220y_{32} \\
\text{sub.to}\quad & 3y_{11}+3y_{12}+x_{12}+x_{13}-x_{21}-x_{31}\le 20 \\
& 3y_{21}+3y_{22}-x_{12}+x_{21}+x_{23}-x_{32}\le 16 \\
& 3y_{31}+3y_{32}-x_{13}-x_{23}+x_{31}+x_{32}\le 24 \\
& y_{11}+y_{21}+y_{31}=7 \\
& y_{12}+y_{22}+y_{32}=13 \\
& y_{21}+y_{22}\le 5 \\
& x_{ij}\ge 0, i, j=1,2,3; i\ne j \\
& y_{ij}\ge 0, i=1,2,3;\ \ j=1,2
\end{aligned}$$

在 Matlab 中实现：

```
f=[75;75;50;50;100;100;150;240;210;120;160;220];
A=[1 -1 1 -1 0 0 3 3 0 0 0 0
   -1 1 0 0 1 -1 0 0 3 3 0 0
    0 0 -1 1 -1 1 0 0 0 0 3 3
    0 0 0 0 0 0 0 0 1 1 0 0];
b=[20;16;24;5];
Aeq=[0 0 0 0 0 0 1 0 1 0 1 0
     0 0 0 0 0 0 0 1 0 1 0 1];
beq=[7;13];
```

```
lb=zeros(12,1);
[x,fval,exitflag,output,lambda]=linprog(f,A,b,Aeq,beq,lb)
Optimization terminated.
x =
    0.0000
    1.0000
    0.0000
    0.0000
    0.0000
    0.0000
    7.0000
    0.0000
    0.0000
    5.0000
    0.0000
    8.0000
fval =
   3.4850e+03
exitflag =
     1
output =
          iterations: 8
           algorithm: 'interior-point-legacy'
      cgiterations: 0
             message: 'Optimization terminated.'
    constrviolation: 4.9738e-14
       firstorderopt: 4.2980e-11
lambda =
    ineqlin: [4x1 double]
       eqlin: [2x1 double]
       upper: [12x1 double]
       lower: [12x1 double]
```

因此，要使总费用最小，需要 B 地向 A 地运送 1 万吨原料，A、B、C 三地的建厂规模分别为 7 万吨、5 万吨、8 万吨。最小总费用为 3485 万元。

3.2 非线性规划问题

3.2.1 非线性无约束规划问题

无约束规划由 3 个功能函数 fminbnd、fminsearch、fminunc 实现。

3.2.1.1 fminbnd 函数

函数：fminbnd

功能：求取固定区间内**单变量**函数的最小值，也就是一元函数最小值问题。

数学模型：

$$\min_x f(x)$$

$$x_1 < x < x_2$$

式中，x, x_1 和 x_2 为标量，$f(x)$ 为函数，返回标量。

格式：
```
x = fminbnd(fun,x1,x2)
x = fminbnd(fun,x1,x2,options)
x = fminbnd(fun,x1,x2,options,P1,P2,...)
[x,fval] = fminbnd(...)
[x,fval,exitflag] = fminbnd(...)
[x,fval,exitflag,output] = fminbnd(...)
```

说明：

fminbnd **求取固定区间内单变量函数的最小值**

x = fminbnd(fun,x1,x2) 返回[x1, x2]区间上 fun 参数描述的标量函数的最小值点 x。

x = fminbnd(fun,x1,x2,options) 用 options 参数指定的优化参数进行最小化。

x = fminbnd(fun,x1,x2,options,P1,P2,...) 提供另外的参数 P1，P2 等，传输给目标函数 fun。如果没有设置 options 选项，则令 options = [ ]。

[x,fval] = fminbnd(...) 返回解 x 处目标函数的值 fval。

[x,fval,exitflag] = fminbnd(...) 返回 exitflag 值描述 fminbnd 函数的退出条件。

[x,fval,exitflag,output] = fminbnd(...) 返回包含优化信息的结构输出。

·fun：需要**最小化的目标函数**。fun 函数需要输入标量参数 x，返回 x 处的目标函数标量值 f。可以将 fun 函数指定为命令行，如

x = fminbnd (inline ('sin (x*x)'), x0)

inline 函数可以直接定义函数的表达式，不需要使用函数的 m 文件。

同样，fun 参数可以是一个包含函数名的字符串。对应的函数可以是 M 文件、内部函数或 MEX 文件。若 fun = 'myfun'， 则

x = fminbnd(**@myfun**,x0)

其中 M 文件函数 myfun.m 必须为下面的形式

```
function f = myfun (x)
f = …            %计算 x 处的函数值。
```

·options：优化参数选项。你可以用 optimset 函数设置或改变这些参数的值。Options 参数有以下几个选项：

- Display 显示的水平。选择**'off'**，不显示输出；选择**'iter'**，显示每一步迭代过程的输出；选择**'final'**，显示最终结果；
- MaxFunEvals 函数评价的最大允许次数；
- MaxIter 最大允许迭代次数；
- TolX x 处的终止容限。

·exitflag：描述退出条件：

- >0 表示目标函数**收敛**于解 x 处；
- 0 表示已经**达到函数评价或迭代的最大次数**；
- <0 表示目标函数**不收敛**。

·output：该参数包含下列优化信息：

- output .iteratiions 迭代次数；
- output .algorithm 所采用的算法；
- output .funcCount 函数评价次数。

注意：

（1）目标函数必须是连续的；

（2）fminbnd 函数可能只给出局部最优解；

（3）当问题的解位于区间边界上时，fminbnd 函数的收敛速度常常很慢。此时，**fmincon** 函数的计算速度更快，计算精度更高；

（4）fminbnd 函数只用于实数变量。

例 3-4 在(0, 2π) 上求函数 sinx 的最小值。

解：Matlab 中实现：

```
>> [x,y_min]=fminbnd('sin(x)',0,2*pi)
x =
    4.7124
y_min =
   -1.0000
```

或

```
>> [x,y_min]=fminbnd(@sin,0,2*pi)
x =
    4.7124
y_min =
   -1.0000
```

例 3-5 对边长为 3m 的正方形铁板，在四个角处剪去相等的小正方形以制成方形无盖盒子，问如何剪法使盒子容积最大？

解：设剪去的正方形的边长为 x，则盒子容积为

$$f(x) = (3-2x)^2 x$$

现在要求在区间(0, 1.5)上确定 x，使 f (x)最大化。因为优化工具箱中要求目标函数最小化，所以需要对目标函数进行转换，即要求−f (x)最小化。

在 Matlab 中实现：

```
>> [x,f_min]=fminbnd('-(3-2*x)^2*x',0,1.5)
x =
    0.5000
f_min =
   -2.0000
```

或编写 M 文件 Ex1005.m

```
function y=Ex1005(x)
y = -(3-2*x)^2*x;
```

```
>> [x,f_min]=fminbnd(@Ex1005,0,1.5)
x =
    0.5000
f_min =
   -2.0000
```

即剪去边长为 0.5 m 的正方形，最大容积为 2 m^3。

3.2.1.2 fminsearch 函数

函数：fminsearch

功能：求解**多变量**无约束函数的最小值。

数学模型：

$$\min_x f(x)$$

其中，x 为向量，$f(x)$ 为一函数，返回标量。

格式：
```
x = fminsearch(fun,x0)
x = fminsearch(fun,x0,options)
x = fminsearch(fun,x0,options,P1,P2,...)
[x,fval] = fminsearch(...)
[x,fval,exitflag] = fminsearch(...)
[x,fval,exitflag,output] = fminsearch(...)
```

说明：

fminsearch 求解多变量无约束函数的最小值。该函数常用于无约束非线性最优化问题。

x = fminsearch(fun,x0) 初值为 x0，求 fun 函数的局部极小点 x。x0 可以是标量、向量或矩阵。

x = fminsearch(fun,x0,options) 用 options 参数指定的优化参数进行最小化。

x = fminsearch(fun,x0,options,P1,P2,...) 将问题参数 P1、P2 等直接输给目标函数 fun，将 options 参数设置为空矩阵，作为 options 参数的默认值。

[x,fval] = fminsearch(...) 将 x 处的目标函数值返回到 fval 参数中。

[x,fval,exitflag] = fminsearch(...) 返回 exitflag 值，描述函数的退出条件。

[x,fval,exitflag,output] = fminsearch(...) 返回包含优化信息参数 output 的结构输出。

各变量的意义同前及下面 fminunc 函数。

注意：

（1）应用 fminsearch 函数可能会得到局部最优解；

（2）fminsearch 函数只对实数进行最小化，即 x 必须由实数组成，f (x)函数必须返回实数。如果 x 为复数，则必须将它分为实数部和虚数部两部分；

（3）对于求解二次以上的问题，fminunc 函数比 fminsearch 函数有效，但对于高度非线性不连续问题时，fminsearch 函数更具稳键性。

（4）fminsearch 函数不适合求解平方和问题，用 lsqnonlin 函数更好一些。

例 3-6 求 $2x_1^3+4x_1x_2^3-10x_1x_2+x_2^2$ 的最小值。

解：在 Matlab 中实现如下：

```
f='2*x(1)^3+4*x(1)*x(2)^3-10*x(1)*x(2)+x(2)^2';
x0=[0,0];
[x,f_min]=fminsearch(f,x0)
x =
    1.0016    0.8335
f_min =
   -3.3241
```

或在 Matlab 编辑器中编辑 M 文件 Ex1006.m：

```
function f=Ex1006(x)
f=2*x(1)^3+4*x(1)*x(2)^3-10*x(1)*x(2)+x(2)^2;
```

命令窗口运行：

```
x0=[0,0];
[x,f_min]=fminsearch(@Ex1006,x0)
x =
    1.0016    0.8335
f_min =
   -3.3241
```

运行后结果一致。

3.2.1.3 fminunc 函数

函数：fminunc

功能：求**多变量无约束函数**的最小值。

数学模型：

$$\min_x f(x)$$

其中，x为向量，$f(x)$为一函数，返回标量。

格式：

```
x = fminunc(fun,x0)
x = fminunc(fun,x0,options)
x = fminunc(fun,x0,options,P1,P2,...)
[x,fval] = fminunc(...)
[x,fval,exitflag] = fminunc(...)
[x,fval,exitflag,output] = fminunc(...)
[x,fval,exitflag,output,grad] = fminunc(...)
[x,fval,exitflag,output,grad,hessian] = fminunc(...)
```

说明：

fminunc 给定初值，**求多变量标量函数的最小值**。常用于无约束非线性最优化问题。

x = fminunc(fun,x0) 给定初值 x0，求 fun 函数的局部极小点 x。x0 可以是标量、向量或矩阵。

x = fminunc(fun,x0,options) 用 options 参数指定的优化参数进行最小化。

x = fminunc(fun,x0,options,P1,P2,...) 将问题参数 P1、P2 等直接输给目标函数 fun，将 options 参数设置为空矩阵，作为 options 参数的默认值。

[x,fval] = fminunc(...) 将 x 处的目标函数值返回到 fval 参数中。

[x,fval,exitflag] = fminunc(...) 返回 exitflag 值，描述函数的退出条件。

[x,fval,exitflag,output] = fminunc(...) 返回包含优化信息参数 output 的结构输出。

[x,fval,exitflag,output,grad] = fminunc(...) 将解 x 处 fun 函数的梯度值返回到 grad 参数中。

[x,fval,exitflag,output,grad,hessian] = fminunc(...) 将解 x 处目标函数的 Hessian 矩阵信息返回到 hessian 参数中。

·fun 变量：为**目标函数**。需要最小化的目标函数。fun 函数需要输入向量参数 x，返回 x 处的目标函数标量值 f。可以将 fun 函数指定为命令行，如

```
x = fminunc(inline('norm(x)^2'),x0)
```

同样，fun 函数可以是一个包含函数名的字符串。对应的函数可以是 M 文件、内部函数或 MEX 文件。若 fun = ‘myfun’，则

```
x = fminunc(@myfun,x0)
```

其中 M 文件函数 myfun.m 必须有下面的形式：

```
function f = myfun (x)
f = …                %计算 x 处的函数值。
```

若 fun 函数的梯度可以算得，且 options.GradObj 设为’on’（用下式设定）

```
options = optimset (‘GradObj’, ‘on’)
```

则 fun 函数必须返回解 x 处的梯度向量 g 到第二个输出变量中去。注意，当被调用的 fun 函数只需要一个输出变量时（如算法只需要目标函数的值而不需要其梯度值时），可以通过核对 nargout 的值来避免计算梯度值

```
function[f, g] = myfun (x)
f = …                %计算 x 处的函数值
if nargout > 1       %调用 fun 函数并要求有两个输出变量
    g = …            %计算 x 处的梯度值
end
```

若 Hessian 矩阵可以求得，并且 options. Hessian 设为’on’，即

```
options = optimset (‘Hessian’, ‘on’)
```

则 fun 函数必须返回解 x 处的 Hessian 对称矩阵 H 到第三个输出变量中去。注意，当被调用的 fun 函数只需要一个或两个输出变量时（如算法只需要目标函数的值 f 和梯度值 g 而不需要 Hessian 矩阵 H 时），可以通过核对 nargout 的值来避免计算 Hessian 矩阵

```
function[f, g] = myfun (x)
f = …                %计算 x 处的函数值
if nargout > 1       %调用 fun 函数并要求有两个输出变量
    g = …            %计算 x 处的梯度值
    if nargout > 2
    H = …            %计算 x 处的 Hessian 矩阵
end
```

·options 变量：优化参数选项。可以通过 optimset 函数设置或改变这些参数。其中有的参数适用于所有的优化算法，有的则只适用于大型优化问题，另外一些则只适用于中型问题。

首先描述适用于大型问题的选项。这仅仅是一个参考，因为使用大型问题算法有一些条件。对于 fminunc 函数来说，必须提供梯度信息

- LargeScale　当设为’on’时，使用大型算法，若设为’off’则使用中型问题的算法

适用于大型和中型算法的参数：

- Diagnostics　打印最小化函数的诊断信息
- Display　显示水平。选择’off’，不显示输出；选择’iter’，显示每一步迭代过程的输出；选择’final’，显示最终结果
- GradObj　用户定义的目标函数的梯度。对于大型问题此参数是必选的，对于中型问题则是可选项
- MaxFunEvals　函数评价的最大次数
- MaxIter　最大允许迭代次数
- TolFun　函数值的终止容限
- TolX　x 处的终止容限

只适用于大型算法的参数：

- Hessian　用户定义的目标函数的 Hessian 矩阵
- HessPattern　用于有限差分的 Hessian 矩阵的稀疏形式。若不方便求 fun 函数的稀疏 Hessian 矩阵 H，可以通过用梯度的有限差分获得的 H 的稀疏结构（如非零值的位置等）来得到近似的 Hessian 矩阵 H。若连矩阵的稀疏结构都不知道，则可以将 HessPattern 设为密集矩阵，在每一次迭代过程中，都将进行密集矩阵的有限差分近似（这是默认设置）。这将非常麻烦，所以花一些力气得到 Hessian 矩阵的稀疏结

构还是值得的

- MaxPCGIter　　PCG 迭代的最大次数
- PrecondBandWidth　　PCG 前处理的上带宽，默认时为零。对于有些问题，增加带宽可以减少迭代次数
- TolPCG　　PCG 迭代的终止容限
- TypicalX　　典型 x 值

只适用于中型算法的参数：

- DerivativeCheck　　对用户提供的导数和有限差分求出的导数进行对比
- DiffMaxChange　　变量有限差分梯度的最大变化
- DiffMixChange　　变量有限差分梯度的最小变化
- LineSearchType　　一维搜索算法的选择

- **exitflag 变量：**描述退出条件：
  - >0　　表示目标函数**收敛**于解 x 处
  - 0　　表示已经**达到函数评价或迭代的最大次数**
  - <0　　表示目标函数**不收敛**
- **output 变量：**该参数包含下列优化信息：
  - output.iterations　　迭代次数
  - output.algorithm　　所采用的算法
  - output.funCount　　函数评价次数
  - output.cgiterations　　PCG 迭代次数（只适用于大型规划问题）
  - output.stepsize　　最终步长的大小（只适用于中型问题）
  - output.firstorderopt　　一阶优化的度量，解 x 处梯度的范数

注意：

（1）目标函数必须是连续的。fminunc 函数有时会给出局部最优解；

（2）fminunc 函数只对实数进行优化，即 x 必须为实数，而且 f (x)必须返回实数。当 x 为复数时，必须将它分解为实部和虚部；

（3）在使用大型算法时，用户必须在 fun 函数中提供梯度(options 参数中 GradObj 属性必须设置为'on')，否则将给出警告信息；

（4）目前，若在 fun 函数中提供了解析梯度，则 options 参数 DerivativeCheck 不能用于大型算法以比较解析梯度和有限差分梯度。通过将 options 参数的 MaxIter 属性设置为 0 来用中型方法核对导数，然后重新用大型方法求解问题；

（5）对于求解平方和问题，fminunc 函数不是最好的选择，用 lsqnonlin 函数效果更佳。

例 3-7 最小化下列函数：

$$f(x) = 3x_1^2+2x_1x_2+x_2^2$$

解：使用 M 文件，创建文件 Ex10071.m：

```
function f=Ex10071(x)
f=3*x(1)^2+2*x(1)*x(2)+x(2)^2;
```

然后调用 fminunc 函数求[1，1]附近 f (x)函数的最小值：

```
x0=[1,1];
[x,fval]=fminunc(@Ex10071,x0)
Warning: Gradient must be provided for trust-region method; using line-search method
instead.
> In fminunc at 241
Optimization terminated: relative infinity-norm of gradient less than options.TolFun.
x =
   1.0e-006 *
     0.2541    -0.2029
fval =
   1.3173e-013
```

下面用提供的梯度 g 最小化函数，修改 M 文件为 Ex10072.m：

```
function [f,g]=Ex10072(x)
f=3*x(1)^2+2*x(1)*x(2)+x(2)^2;
if nargout>1
```

```
    g(1)=6*x(1)+2*x(2);
    g(2)=2*x(1)+2*x(2);
end
```

下面通过将优化选项结构 options.GradObj 设置为'on'来得到梯度值。

```
options=optimset('GradObj','on');
x0=[1,1];
[x,fval]=fminunc(@Ex10072,x0,options)
Optimization completed because the size of the gradient is less than
the default value of the function tolerance.
x =
    1.0e-15 *
     0.1110      -0.8882
fval =
    6.2862e-31
```

例 3-9 求无约束非线性问题

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \qquad x0 = [-1.2, 1]$$

解：在 Matlab 中实现：

```
x0=[-1.2,1];
[x,fval]=fminunc('100*(x(2)-x(1)^2)^2+(1-x(1))^2',x0)
Warning: Gradient must be provided for trust-region method;
   using line-search method instead.
> In fminunc at 241
Optimization terminated: relative infinity-norm of gradient less than options.TolFun.
x =
     1.0000       1.0000
fval =
   2.8372e-011
```

3.2.2 二次规划

数学模型：如果某非线性规划的**目标函数为自变量的二次函数，约束条件全是线性函数**，就称这种规划为**二次规划**。其数学模型为

$$\min_x \frac{1}{2} x^T H x + f^T x$$
$$A \cdot x \le b$$
$$Aeq \cdot x = beq$$
$$lb \le x \le ub$$

其中，H，A 和 Aeq 为矩阵，f，b，beq，lb，ub 和 x 为向量。

函数：quadprog

功能：求解二次规划问题。

格式：

```
x = quadprog(H,f,A,b)
x = quadprog(H,f,A,b,Aeq,beq)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options,p1,p2,...)
[x,fval] = quadprog(...)
[x,fval,exitflag] = quadprog(...)
[x,fval,exitflag,output] = quadprog(...)
[x,fval,exitflag,output,lambda] = quadprog(...)
```

说明：

x = quadprog(H,f,A,b) 返回向量 x，最小化函数 1/2*x'*H*x+f'*x，其约束条件为 A*x<=b。

x = quadprog(H,f,A,b,Aeq,beq) 仍求上面的解，但添加了**等式约束**条件 Aeq*x = beq。

x = quadprog(H,f,A,b,Aeq,beq,lb,ub) 定义设计变量的**下界 lb 和上界 ub**，使得 lb<=x<=ub。

x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0) 同上，并设置**初值** x0。

x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options) 根据 options 参数指定的优化参数进行最小化。

x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options,p1,p2,...) 将参数 P1，P2 等直接输给 Hessian 乘子函数，如果存在，用 options 参数中的 HessMult 属性指定。

[x,fval] = quadprog(...) 返回解 x 处的目标函数 fval = 1/2*x'*H*x+f'*x。

[x,fval,exitflag] = quadprog(...) 返回 exitflag 参数，描述计算的退出条件。

[x,fval,exitflag,output] = quadprog(...) 返回包含优化信息的结构输出 output。

[x,fval,exitflag,output,lambda] = quadprog(...) 返回解 x 处包含 Lagrange 乘子的 lambda 参数。

各变量的意义同前。

注意：

（1）一般地，如果问题不是严格凸性的，用 quadprog 函数得到的可能是局部最优解；

（2）如果用 Aeq 和 Beq 明确地指定等式约束，而不是用 lb 和 ub 指定，则可以得到更好的数值解；

（3）若 x 的组分没有上限或下限，则 quadprog 函数希望将对应的组分设置为 Inf（对于上限）或-Inf（对于下限），而不是强制性地给予上限一个很大的数或给予下限一个很小的负数；

（4）对于大型优化问题，若没有提供初值 x0，或 x0 不是严格可行，则 quadprog 函数会选择一个新的初始可行点；

（5）若为等式约束，且 quadprog 函数发现负曲度(negative curvature)，则优化过程终止，exitflag 的值等于-1；

（6）此时，显示水平只能选择'off'和'final'，迭代参数'iter'不可用；

（7）当问题不定或负定时，常常无解（此时 exitflag 参数给出一个负值，表示优化过程不收敛）。若正定解存在，则 quadprog 函数可能只给出局部极小值，因为问题可能是非凸的；

（8）对于大型问题，不能依靠线性等式，因为 Aeq 必须是行满秩的，即 Aeq 的行数必须不多于列数。若不满足要求，必须调用中型算法进行计算；

（9）大型化问题　　大型化问题不允许约束上限和下限相等，如若 lb (2)= =ub (2)，则给出以下出错信息：

Equal upper and lower bounds not permitted in this large-scale method.Use equality constraints and the medium-scale method instead.

若优化模型中只有等式约束，仍然可以使用大型算法；如果模型中既有等式约束又有边界约束，则必须使用中型方法。

中型优化问题　　当解不可行时，quadprog 函数给出以下警告：

Warning: The constraints are overly stringent; there is no feasible solution.

这里，quadprog 函数生成使约束条件矛盾最坏程度最小的结果。

当等式约束不连续时，给出下面的警告信息：

Warning: The equality constraints are overly stringent; there is no feasible solution.

当 Hessian 矩阵为负半定时，生成无边界解，给出下面的警告信息：

Warning: The solution is unbounded and at infinity; the constraints are not restrictive enough.

这里，quadprog 函数返回满足约束条件的 x 值。

例 3-10 求解下面的最优化问题：

目标函数

$$f(x)=\frac{1}{2}x_1^2+x_2^2-x_1x_2-2x_1-6x_2$$

约束条件

$$\begin{aligned}&x_1+x_2\le 2\\&-x_1+2x_2\le 2\\&2x_1+x_2\le 3\\&0\le x_1,\quad 0\le x_2\end{aligned}$$

解：首先，目标函数写成下面的矩阵形式：

$$H=\begin{bmatrix}1 & -1\\ -1 & 2\end{bmatrix},\quad f=\begin{bmatrix}-2\\ -6\end{bmatrix},\quad x=\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

在 Matlab 中实现：

```
H=[1 -1;-1 2];
f=[-2;-6];
A=[1 1;-1 2;2 1];
b=[2;2;3];
lb=zeros(2,1);
[x,fval,exitflag,output,lambda]=quadprog(H,f,A,b,[],[],lb)
```

```
Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.
x =
    0.6667
    1.3333
fval =
   -8.2222
exitflag =
     1
output =
                message: 'Minimum found that satisfies the constraints.…'
              algorithm: 'interior-point-convex'
          firstorderopt: 4.4208e-11
       constrviolation: 0
             iterations: 4
          cgiterations: []
lambda =
       ineqlin: [3x1 double]
         eqlin: [0x1 double]
         lower: [2x1 double]
         upper: [2x1 double]
>> lambda.lower
ans =
   1.0e-11 * 0.4486
    0.0003
>> lambda.ineqlin
ans =
    3.1111
    0.4444
    0.0000
```

3.2.3 有约束规划

函数：fmincon

功能：求**多变量有约束非线性函数**的最小值。

数学模型：

$$\begin{aligned}&\min_x f(x)\\ &c(x)\le 0\\ &ceq(x)=0\\ &A\cdot x\le b\\ &Aeq\cdot x=beq\\ &lb\le x\le ub\end{aligned}$$

其中，x，b，beq，lb 和 ub 为向量，A 和 Aeq 为矩阵，$c(x)$ 和 $ceq(x)$ 为函数，返回标量。

$f(x)$， $c(x)$ 和 $ceq(x)$ 可以是非线性函数。

格式： x = fmincon(fun,x0,A,b)

x = fmincon(fun,x0,A,b,Aeq,beq)

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub)

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options)

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,P1,P2, ...)

[x,fval] = fmincon(...)

[x,fval,exitflag] = fmincon(...)

[x,fval,exitflag,output] = fmincon(...)

[x,fval,exitflag,output,lambda] = fmincon(...)

[x,fval,exitflag,output,lambda,grad] = fmincon(...)

[x,fval,exitflag,output,lambda,grad,hessian] = fmincon(...)

说明：

fmincon 求多变量有约束非线性函数的最小值。该函数常用于有约束非线性优化问题。

x = fmincon(fun,x0,A,b) 给定初值 x0，求解 fun 函数的最小值点 x。fun 函数的约束条件为 A*x<=b，x0 可以是标量、向量或矩阵。

x = fmincon(fun,x0,A,b,Aeq,beq) 最小化 fun 函数，约束条件为 A*x<=b 和 Aeq*x = beq。若没有不等式存在，则设置 A = [ ]，b = [ ]。

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub) 定义设计变量 x 的下界 lb 和上界 ub，使得 lb<=x<= ub。若无等式存在，则令 Aeq = [ ]，beq = [ ]。

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon) 在上面的基础上，在 **nonlcon 参数中提供非线性不等式 c (x)或等式 ceq (x)**。fmincon 函数要求 c (x)<=0 且 ceq (x) = 0。当无边界存在时，令 lb = [ ]和（或）ub = [ ]。

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options) 用 options 参数指定的参数进行最小化。

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,P1,P2, ...) 将问题参数 P1，P2 等直接传递给函数 fun 和 nonlcon。若不需要参数 A，b，Aeq，beq，lb，ub，nonlcon 和 options，将它们设置为空矩阵。

[x,fval] = fmincon(...) 返回解 x 处的目标函数值。

[x,fval,exitflag] = fmincon(...) 返回 exitflag 参数，描述计算的退出条件。

[x,fval,exitflag,output] = fmincon(...) 返回包含优化信息的结构输出 output。

[x,fval,exitflag,output,lambda] = fmincon(...) 返回解 x 处包含 Lagrange 乘子的 lambda 参数。

[x,fval,exitflag,output,lambda,grad] = fmincon(...) 返回解 x 处 fun 函数的梯度。

[x,fval,exitflag,output,lambda,grad,hessian] = fmincon(...) 返回解 x 处 fun 函数的 Hessian 矩阵。

·nonlcon 参数

该参数计算非线性不等式约束 c (x)<=0 和非线性等式约束 ceq (x) = 0。nonlcon 参数是一个包含函数名的字符串。该函数可以是 M 文件、内部文件或 MEX 文件。它要求输入一个向量 x，返回两个变量——解 x 处的非线性不等式向量 c 和非线性等式向量 ceq。例如，若 nonlcon = ‘mycon’，则 M 文件 mycon.m 具有下面的形式：

```
function [c, ceq] = mycon (x)
c = …              %计算 x 处的非线性不等式
ceq =…             %计算 x 处的非线性等式
```

若还计算了约束的梯度，即

```
options = optimset (‘GradConstr’, ‘on’)
```

则 nonlcon 函数必须在第三个和第四个输出变量中返回 c (x)的梯度 GC 和 ceq (x)的梯度 GCeq。当被调用的 nonlcon 函数只需要两个输出变量（此时优化算法只需要 c 和 ceq 的值，而不需要 GC 和 GCeq）时，可以通过查看 nargout 的值来避免计算 GC 和 GCeq 的值。

```
function [c, ceq, GC, GCeq] = mycon (x)
c = …              %解 x 处的非线性不等式
```

ceq =…　　%解 x 处的非线性等式
if nargout>2　　%被调用的 nonlcon 函数，要求有 4 个输出变量
　　GC = …　　%不等式的梯度
　　GCeq = …　%等式的梯度
end

若 nonlcon 函数返回 m 元素的向量 c 和长度为 n 的 x，则 c (x)的梯度 GC 是一个 n×m 的矩阵，其中 GC(i, j)是 c (j)对 x (i)的偏导数。同样，若 ceq 是一个 p 元素的向量，则 ceq (x)的梯度 GCeq 是一个 n×p 的矩阵，其中 GCeq(i, j)是 ceq (j)对 x (i)的偏导数。

其它参数意义同前。

注意：

（1）大型优化问题：

（1.1）使用大型算法，必须在 fun 函数中提供梯度信息（options.GradObj 设置为'on'）。如果没有梯度信息，则将给出警告信息。

Fmincon 函数允许 g (x)为一近似梯度，但使用真正的梯度将使优化过程更具稳键性。

（1.2）当对矩阵的二阶导数（即 Hessian 矩阵）进行计算后，用该函数求解大型问题将更有效。但不需要求得真正的 Hessian 矩阵，如果能提供 Hessian 矩阵稀疏结构的信息（用 options 参数的 HessPattern 属性），则 fmincon 函数可以算得 Hessian 矩阵的稀疏有限差分近似。

（1.3）若 x0 不是严格可行的，则 fmincon 函数选择一个新的严格可行初始点。

（1.4）若 x 的某些元素没有上界或下界，则 fmincon 函数更希望对应的元素设置为 Inf（对于上界）或-Inf（对于下界），而不希望强制性地给上界赋一个很大的值，给下界一个很小的负值。

（1.5）线性约束最小化课题中也有几个问题需要注意：

• Aeq 矩阵中若存在密集列或近密集列（A dense or fairly dense column），会导致满秩并使计算费时；

• fmincon 函数剔除 Aeq 中线性相关的行。此过程需要进行反复的因子分解，因此，如果相关行很多的话，计算将是一件很费时的事情；

• 每一次迭代都要用下式进行稀疏最小二乘求解

$$B = Aeq^T R^{-T}$$

其中 R^T 为前提条件的 Cholesky（乔累斯基）因子。

（2）中型优化问题：

（2.1）如果用 Aeq 和 beq 清楚地提供等式约束，将比用 lb 和 ub 获得更好的数值解。

（2.2）在二次子问题中，若有等式约束并且因等式（dependent equalities）被发现和剔除的话，将在过程标题中显示' dependent'（当 output 参数要求使用 options.Display = 'iter'）。只有在等式连续的情况下，因等式才会被剔除。若等式系统不连续，则子问题将不可行并在过程标题中打印'infeasible'信息。

（3）求大型优化问题的代码中不允许上限和下限相等，即不能有 lb (2)= =ub (2)，否则给出下面的出错信息：

Equal upper and lower bounds not permitted in this large-scale method.
Use equality constraints and the medium-scale method instead.

若只有等式约束，仍然可以使用大型算法。当既有等式约束又有边界约束时，使用中型算法。

（4）目标函数和约束函数都必须是连续的，否则可能会只给出局部最优解。

（5）当问题不可行时，fmincon 函数将试图使最大约束值最小化。

（6）目标函数和约束函数都必须是实数。

（7）对于大型优化问题，使用大型优化算法时，用户必须在 fun 函数中提供梯度（options 参数的 GradObj 属性必须设置为'on'），并且只可指定上界和下界约束，或者只有线性约束必须存在，Aeq 的行数不能多于列数。

（8）如果在 fun 函数中提供了解析梯度，选项参数 DerivativeCheck 不能与大型方法一

起用，以比较解析梯度和有限差分梯度。可以通过将 options 参数的 MaxIter 属性设置为 0 来用中型方法核对导数，然后用大型方法求解问题。

例 3-11 求解下列优化问题：

目标函数

$$f(x) = -x_1 x_2 x_3$$

约束条件

$$0 \le x_1 + 2x_2 + 2x_3 \le 72$$

解：将约束条件改写成下面的不等式

$$-x_1 - 2x_2 - 2x_3 \le 0$$

$$x_1 + 2x_2 + 2x_3 \le 72$$

两个约束条件都是线性的，在 Matlab 中实现：

```
x0=[10;10;10];
A=[-1 -2 -2;1 2 2];
b=[0;72];
[x,fval]=fmincon('-x(1)*x(2)*x(3)',x0,A,b)
Warning: Large-scale (trust region) method does not currently solve this type of problem,
  switching to medium-scale (line search).
> In fmincon at 260
Optimization terminated: magnitude of directional derivative in search
  direction less than 2*options.TolFun and maximum constraint violation
    is less than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
   lower         upper        ineqlin      ineqnonlin
                                    2
x =
    24.0000
    12.0000
    12.0000
fval =
-3.4560e+003
>>
```

线性不等式约束条件的值<=0

```
>> A*x-b
ans =
    -72
      0
```

例 3-12 求表面积为常数 150 m^2 的体积最大的长方体体积。

解：设长方体的长、宽、高分别为 x_1、x_2 和 x_3，根据题意得到下面的数学模型：

$$\min \quad z = -x_1 x_2 x_3$$

$$2(x_2 x_3 + x_3 x_1 + x_1 x_2) = 150$$

由于约束条件是**非线性等式约束**，所以需要编写一个约束条件 M 文件 Ex1012c.m：

```
function [c,ceq]=Ex1012c(x)
c=0;
ceq=x(2)*x(3)+x(3)*x(1)+x(1)*x(2)-75;
```

在 Matlab 中实现：

```
x0=[4;5;6];
lb=zeros(3,1);
[x,fval,exitflag,output,lambda]=fmincon('-x(1)*x(2)*x(3)',x0,[],[],[],[],lb,[],@Ex1012c)
Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.
x =
    5.0000
    5.0000
```

```
     5.0000
fval =
-125.0000
exitflag =
     1
output =
            iterations: 11
             funcCount: 57
       constrviolation: 2.3688e-08
              stepsize: 1.0568e-04
             algorithm: 'interior-point'
         firstorderopt: 2.1418e-06
          cgiterations: 0
               message: 'Local minimum found that satisfies the constraints.…'
lambda =
            eqlin: [0x1 double]
          eqnonlin: 2.5000
           ineqlin: [0x1 double]
            lower: [3x1 double]
            upper: [3x1 double]
       ineqnonlin: 1.0941e+07
```

优化结果显示过程成功收敛，搜索方向小于两倍 options.TolX，最大违约值小于 options.TolCon，主动约束为 1 个。

问题的解为 x (1) = x (2) = x (3) = 5.0000m，最大体积为 125.0000m^3。exitflag = 1，表示过程成功收敛于解 x 处。output 输出变量显示了收敛过程中的迭代次数、目标函数计算次数、步长、算法等信息。lambda 则包含模型信息。

例 3-13 求解下列优化问题：

$$\begin{aligned} \min \quad & 2x_1^2 + 2x_2^2 - 2x_1x_2 - 4x_1 - 6x_2 \\ & -x_1 - 5x_2 \geq -5 \\ & -2x_1^2 + x_2 \geq 0 \\ & 0 \leq x_1, \quad 0 \leq x_2 \end{aligned}$$

初始点 x0 = [0; 0.75]。

解：由于约束条件中有非线性不等式，所以需要编写一个约束条件 M 文件 Ex1013c.m：

```
function [c,ceq]=Ex1013c(x)
c=2*x(1)^2-x(2);
ceq=0;
```

在 Matlab 中实现：

```
x0=[0;0.75];
A=[1 5];
b=5;
lb=zeros(2,1);
[x,fval,exitflag,output,lambda]=fmincon('2*x(1)^2+2*x(2)^2-2*x(1)*x(2)-4*x(1)-6*x(2)',x0,
                                        A,b,[],[],lb,[],@Ex1013c)
Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.
x =
    0.6589
    0.8682
fval =
   -6.6131
exitflag =
     1
output =
            iterations: 8
```

```
            funcCount: 27
      constrviolation: 0
             stepsize: 3.1988e-08
            algorithm: 'interior-point'
        firstorderopt: 2.0000e-06
         cgiterations: 0
              message: 'Local minimum found that satisfies the constraints.…'
lambda =
           eqlin: [0x1 double]
        eqnonlin: 0
         ineqlin: 0.9335
           lower: [2x1 double]
           upper: [2x1 double]
      ineqnonlin: 0
```

问题的解为 x (1) = 0.6589，x (2) = 0.8682，最小值为 −6.6131。

3.3 目标规划

前面介绍的最优化方法只有一个目标函数，是单目标最优化方法。但是，在许多实际工程问题中，往往希望多个指标都达到最优值，所以它有多个目标函数。这种问题称为多目标最优化问题。

多目标最优化问题的数学模型为

$$\begin{aligned}&\min_{x\in R^n} F(x)\\&G_i(x)=0 \qquad i=1,2,\cdots,m_1\\&G_i(x)\le 0 \qquad i=m_1+1,m_1+2,\cdots,m\\&x_i\le x\le x_u\end{aligned}$$

其中 $F(x)$ 为目标函数向量。

此优化问题在 Matlab 中主要由函数 fgoalattain 来实现。此问题在控制系统中有广泛的应用。

函数：fgoalattain

功能：求解多目标达到问题。

数学模型：

$$\begin{aligned}&\underset{x,\gamma}{\text{minimize}}\ \gamma\\&F(x)-\text{weight}\cdot\gamma\le\text{goal}\\&c(x)\le 0\\&ceq(x)=0\\&A\cdot x\le b\\&Aeq\cdot x=beq\\&lb\le x\le ub\end{aligned}$$

其中 x，weight，goal，b，beq，lb 和 ub 为向量，A 和 Aeq 为矩阵，c (x)，ceq (x)和 F (x)为函数，返回向量。F (x)，c (x) 和 ceq (x)可以是非线性函数。

格式：

```
x = fgoalattain(fun,x0,goal,weight)
x = fgoalattain(fun,x0,goal,weight,A,b)
x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq)
x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq,lb,ub)
x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq,lb,ub,nonlcon)
x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq, lb,ub,nonlcon,options)
x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq, lb,ub,nonlcon,options,P1,P2,...)
[x,fval] = fgoalattain(...)
[x,fval,attainfactor] = fgoalattain(...)
[x,fval,attainfactor,exitflag] = fgoalattain(...)
[x,fval,attainfactor,exitflag,output] = fgoalattain(...)
```

[x,fval,attainfactor,exitflag,output,lambda] = fgoalattain(...)

说明：

fgoalattain 求解多目标达到问题。

x = fgoalattain(fun,x0,goal,weight) 试图通过变化 x 来使目标函数 fun 达到 goal 指定的目标。初值为 x0，weight 参数指定权重。

x = fgoalattain(fun,x0,goal,weight,A,b) 求解目标达到问题，约束条件为线性不等式 A*x <= b。

x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq) 求解目标达到问题，除提供上面的线性不等式外，还提供线性等式 Aeq*x = beq。当没有不等式存在时，设置 A=[ ]和 b=[ ]。

x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq,lb,ub) 为设计变量 x 定义下界 lb 和上界 ub 集合，这样始终有 lb <= x <= ub。

x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq,lb,ub,nonlcon) 将目标达到问题归结为 nonlcon 参数定义的非线性不等式 c (x)或非线性等式 ceq (x)。fgoalattain 优化的约束条件为 c(x) <= 0 和 ceq(x) = 0。若不存在边界，设置 lb=[ ] 和（或）ub=[ ]。

x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq,lb,ub,nonlcon,options) 用 options 中设置的优化参数进行最小化。

x = fgoalattain(fun,x0,goal,weight,A,b,Aeq,beq,lb,ub,nonlcon,options,P1,P2,...) 将问题参数 P1，P2 等直接传递给函数 fun 和 nonlcon。若不需要参数 A，b，Aeq，beq，lb，ub，nonlcon 和 options，将它们设置为空矩阵。

[x,fval] = fgoalattain(...) 返回解 x 处的目标函数值。

[x,fval,attainfactor] = fgoalattain(...) 返回解 x 处的目标达到因子。

[x,fval,attainfactor,exitflag] = fgoalattain(...) 返回 exitflag 参数，描述计算的退出条件。

[x,fval,attainfactor,exitflag,output] = fgoalattain(...) 返回包含优化信息的结构输出 output。

[x,fval,attainfactor,exitflag,output,lambda] = fgoalattain(...) 返回解 x 处包含 Lagrange 乘子的 lambda 参数。

·goal 变量

目标希望达到的向量值。向量的长度与 fun 函数返回的目标数 F 相等。fgoalattain 函数试图通过最小化向量 F 中的值来达到 goal 参数给定的目标。

·nonlcon 函数

该函数意义同前。

·options 变量

优化参数选项。可以用 optimset 函数设置或改变这些参数的值。

DerivativeCheck 比较用户提供的导数（目标函数或约束函数的梯度）和有限差分导数。

Diagnostics 打印将要最小化或求解的函数的诊断信息。

DiffMaxChange 变量中有限差分梯度的最大变化。

DiffMinChange 变量中有限差分梯度的最小变化。

Display 显示水平。设置为’off’时不显示输出；设置为’iter’时显示每一次迭代的输出；设置为’final’时显示最终结果。

GoalsExactAchieve 使得目标个数刚好达到，不多也不少。

GradConstr 用户定义的约束函数的梯度。

GradObj 用户定义的目标函数的梯度。使用大型算法时必须使用梯度，对于中型方法则是可选项。

MaxFunEvals

MaxIter 函数迭代的允许最大次数。

MeritFunction 如果设为’multiobj’，则使用目标达到或最大最小化目标函数的方法；若设置为’singleobj’，则使用 fmincon 函数计算目标函数。

TolCon 约束矛盾的终止容限。

TolFun 函数值处的终止容限。

TolX x 处的终止容限。

·weight 变量

为权重向量，可以控制低于或超过 fgoalattain 函数指定目标的相对程度。当 goal 的值

都是非零值时，为了保证活动对象超过或低于的比例相当，将权重函数设置为 abs (goal)（活动对象为阻止解处目标改善的对象集合）。weight = goal 或 weight = abs (goal)

注意：

（1）当目标值中的任意一个为零时，设置 weight = abs (goal)将导致目标约束看起来更像硬约束，而不像目标约束；

（2）当加权函数 weight 为正时，fgoalattain 函数试图使对象小于目标值。为了使目标函数大于目标值，将权重 weight 设置为负。为了使目标函数尽可能地接近目标值，使用 GoalExactAchieve 参数，将 fun 函数返回的第一个元素作为目标。

·attainfactor 变量

attainfactor 变量是超过或低于目标的个数。若 attainfactor 为负，则目标已经溢出；若 attainfactor 为正，则目标个数还未达到。

其它参数意义同前。

注意：

（1）当特征值为复数时，本问题不连续，这也说明了为什么收敛速度很慢。尽管原始方法假设函数是连续的，该法仍然可以向解的方向前进，因为在解的位置上，没有发生不连续的现象。当对象和目标为复数时，fgoalattain 函数将试图得到最小二乘意义上的目标；

（2）目标函数必须是连续的；

（3）fgoalattain 函数将只给出局部最优解。

例 3-15 某化工厂拟生产两种新产品 A 和 B，其生产设备费用分别为：A，2 万元/吨；B，5 万元/吨。这两种产品均将造成环境污染，设由公害所造成的损失可折算为：A，4 万元/吨；B，1 万元/吨。由于条件限制，工厂生产产品 A 和 B 的最大生产能力各为每月 5 吨和 6 吨，而市场需要这两种产品的总量每月不少于 7 吨。试问工厂如何安排生产计划，在满足市场需要的前提下，使设备投资和公害损失均达到最小。该工厂决策认为，这两个目标中环境污染应优先考虑，设备投资的目标值为 20 万元，公害损失的目标为 12 万元。

解：设工厂每月生产产品 A 为 x_1 吨，B 为 x_2 吨，设备投资费为 $f_1(x)$，公害损失费为 $f_2(x)$，则这个问题可表达为多目标优化问题：

$$\min f_1(x) = 2x_1+5x_2$$
$$\min f_2(x) = 4x_1+x_2$$
$$x_1 \leqslant 5$$
$$x_2 \leqslant 6$$
$$x_1+x_2 \geqslant 7$$
$$x_1,\ x_2 \geqslant 0$$

需要编写目标函数的 M 文件 Ex1015.m，返回**目标**计算值：

```
function f=Ex1015(x)
f(1)=2*x(1)+5*x(2);
f(2)=4*x(1)+x(2);
```

给定目标，权重按目标比例确定，给出初值，在 Matlab 中实现为：

```
goal=[20 12];
weight=[20 12];
x0=[2 5];
A=[1 0;0 1; -1 -1];
b=[5 6 -7];
lb=zeros(2,1);
[x,fval,attainfactor,exitflag]=fgoalattain(@Ex1015,x0,goal,weight,A,b,[],[],lb,[])
fgoalattain stopped because the size of the current search direction is less than
twice the default value of the step size tolerance and constraints are
satisfied to within the default value of the constraint tolerance.
x =
    2.9167    4.0833
fval =
   26.2500   15.7500
attainfactor =
    0.3125
```

exitflag =

4

故工厂每月生产产品 A 为 2.9167 吨，B 为 4.0833 吨。设备投资费和公害损失费的目标值分别为 26.250 万元和 15.750 万元。达到因子为 0.3125，计算收敛。