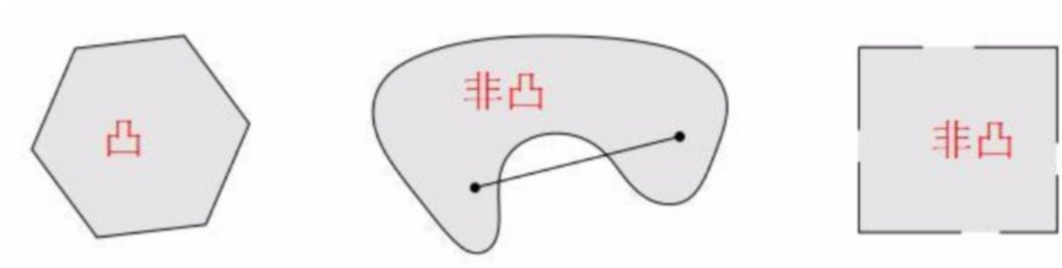
6.3 凸集与凸集分离定律-李涛

6-3、凸集与凸集分离定理

1、凸集

实数域_R_上（或复数_C_上）的向量空间中，如果集合_S_中任两点的连线上的点都在_S_内，则称集合_S_为凸集，如下图所示：



数学定义为：

设集合$D \subset R^n$，若对于任意两点$x, y \in D$，及实数$\lambda(0 \le \lambda \le 1)$都有：

$\lambda x + (1-\lambda) y \in D$

则称集合D为凸集。

2、超平面和半空间

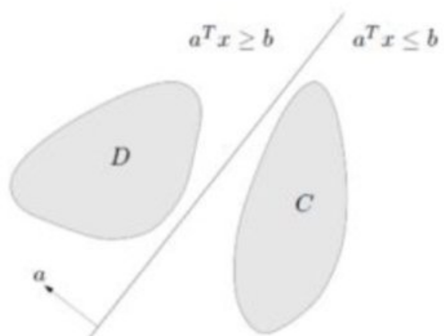
实际上，二维空间的超平面就是一条线（可以使曲线），三维空间的超平面就是一个面（可以是曲面）。其数学表达式如下：

超平面： $H = \{x \in R^n | a_1 + a_2 + \ldots + a_n = b\}$

半空间： $H^+ = \{x \in R^n | a_1 + a_2 + \ldots + a_n \ge b\}$

3、凸集分离定理

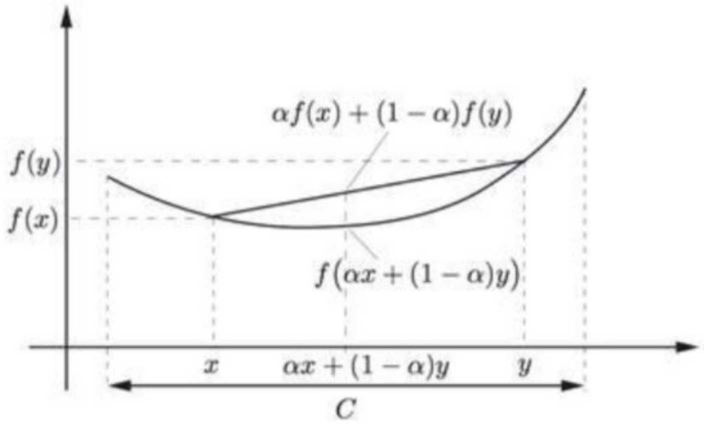
所谓两个凸集分离，直观地看是指两个凸集合没有交叉和重合的部分，因此可以用一张超平面将两者隔在两边，如下图所示：



4、凸函数

凸函数就是一个定义域在某个向量空间的凸子集C上的实值函数

凸函数就是一个定义域在某个向量空间的凸子集C上的实值函数。



数学定义为：

对于函数f(x)，如果其定义域C是凸的，且对于∀x,y∈C，$0 \le \alpha \le 1$，

有：

$$f(\theta x+(1-\theta) y) \le \theta f(x)+(1-\theta) f(y)$$

则f(x)是凸函数。

注：如果一个函数是凸函数，则其局部最优点就是它的全局最优点。这个性质在机器学习算法优化中有很重要的应用，因为机器学习模型最后就是在求某个函数的全局最优点，一旦证明该函数（机器学习里面叫“损失函数”）是凸函数，那相当于我们只用求它的局部最优点了。

```
from cvxpy import *
import cvxpy as cvx
```

```
problem :
minimize (x-y)^2
subject to
        x+y=1
        x-y>=1
```

```
from cvxpy import *
# Create two scalar optimization variables.
# 在CVXPY中变量有标量(只有数值大小)，向量，矩阵。
# 在CVXPY中有常量(见下文的Parameter)
x = Variable() #定义变量x,定义变量y。两个都是标量
y = Variable()
# Create two constraints.
#定义两个约束式
constraints = [x + y == 1,
              x - y >= 1]
#优化的目标函数
obj = Minimize(square(x - y))
```

```
#把目标函数与约束传进Problem函数中
prob = Problem(obj, constraints)
prob.solve()  # Returns the optimal value.
print("status:", prob.status)
print("optimal value", prob.value) #最优值
print("optimal var", x.value, y.value) #x与y的解
# Replace the objective. 不同的目标函数，相同的约束
prob2 = cvx.Problem(cvx.Maximize(x + y), prob.constraints)
print("optimal p1 value", prob2.solve())

# Replace the constraint (x + y == 1).#不同的约束，相同的目标函数
constraints = [x + y <= 3] + prob.constraints[1:] #注意：此处是列表相加，
prob.constraints[1:]取prob的约束集中
#从第二个开始的所有约束。
prob2 = cvx.Problem(prob.objective, constraints)
print("optimal P2 value", prob2.solve())
```