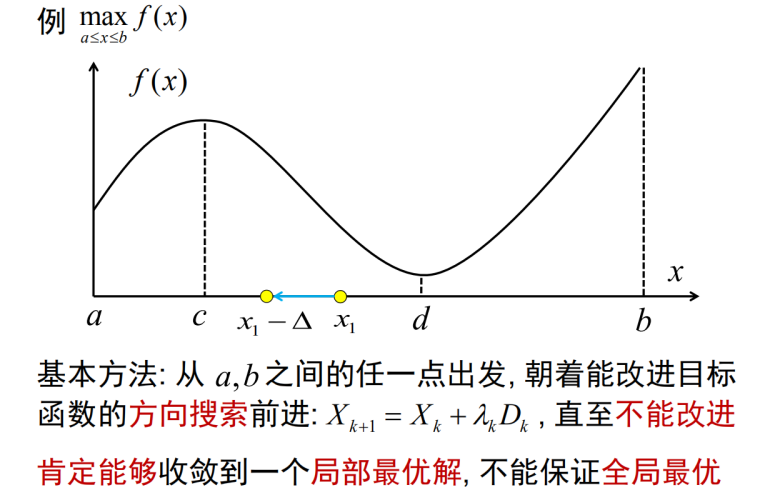
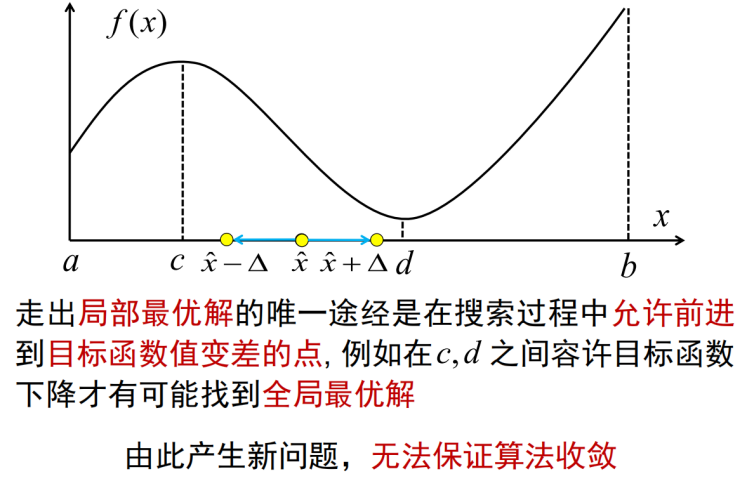
运筹学主要用于解决现实生活中各种复杂问题，特别是改善或优化。其主要分支包括一般的数学规划模型如线性规划、整数规划、非线性规划、动态规划、网络优化等，也包括特定实际问题的数学模型如网络计划、排队论、存储论、决策论、对策论等。

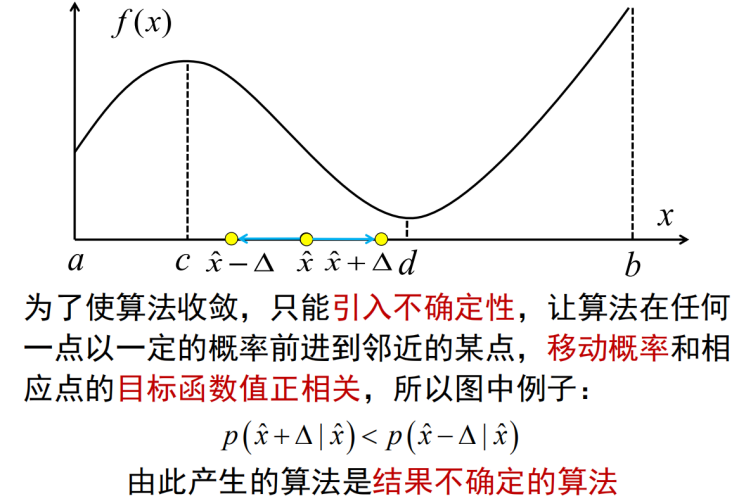
很显然，一切运筹学所研究的问题都是优化的问题，而从优化本身出发，可分为基于确定性的搜索与基于不确定性的搜索：前者是经典的优化课程介绍的主要内容，后者包括模拟退火、禁忌搜索、遗传算法、免疫算法、蚂蚁算法等方法，一般统称为智能算法。

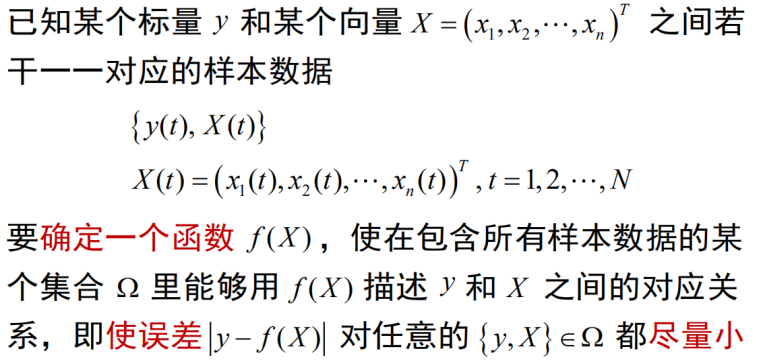
基于确定性的搜索如梯度下降法：



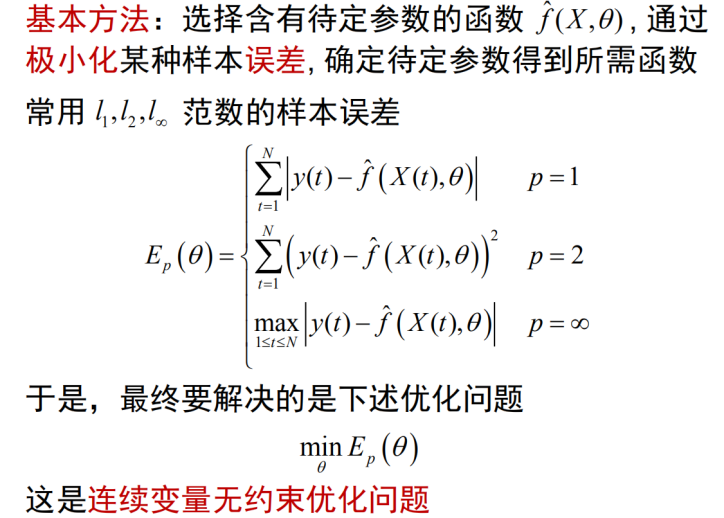
基于不确定的搜索：

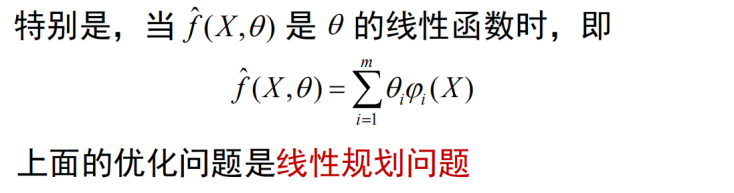


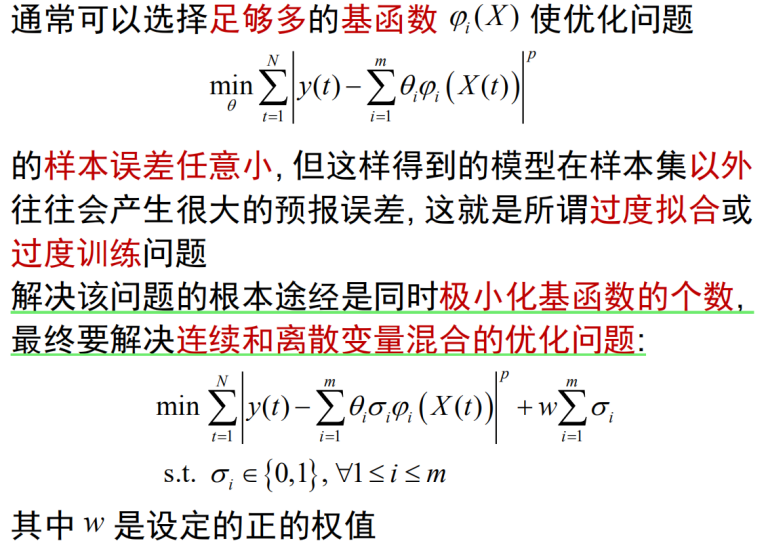
基于不确定性的搜索改进版： 而要在实际问题中应用运筹学的相关知识，则需要建立合适的模型，以一个典型的回归例子举例：

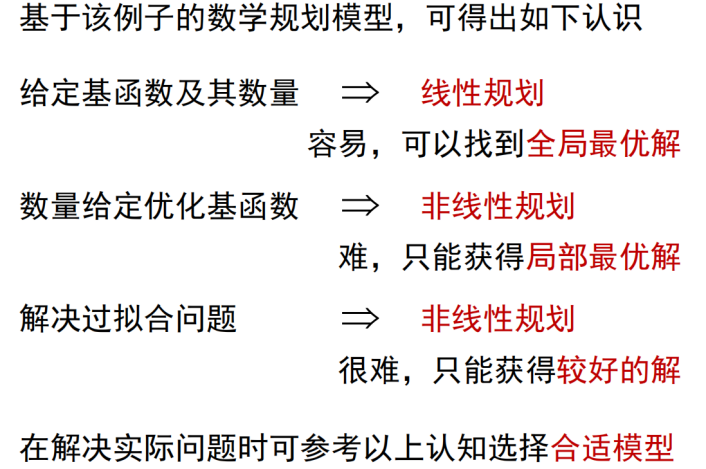


我们通常采用极小化误差函数的方法：





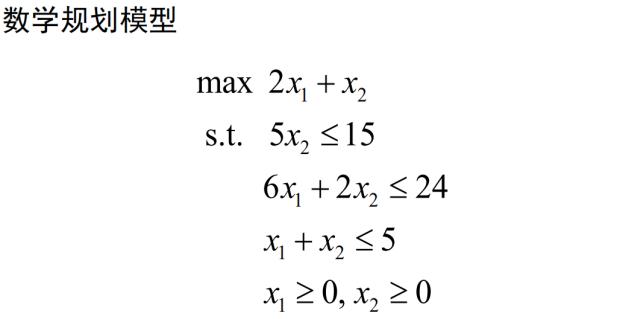




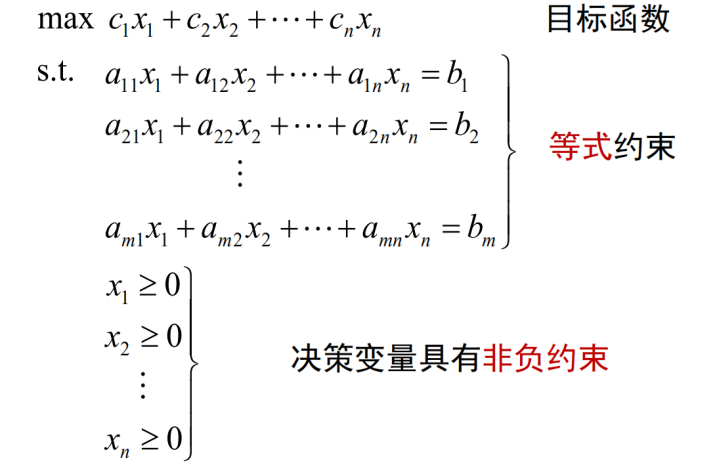
运筹学中最简单的是线性规划模型，我们中学时都遇到过这样的题目：



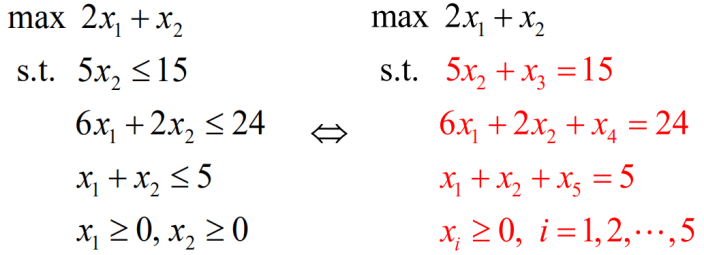
显然可建立数学模型为:



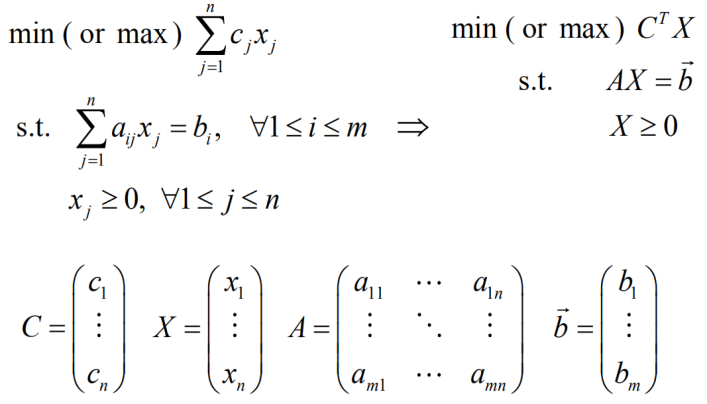
为了统一地研究这一类问题，引入线性规划标准模型：



其中可引入变量的方式将不等式条件转化为等式条件：



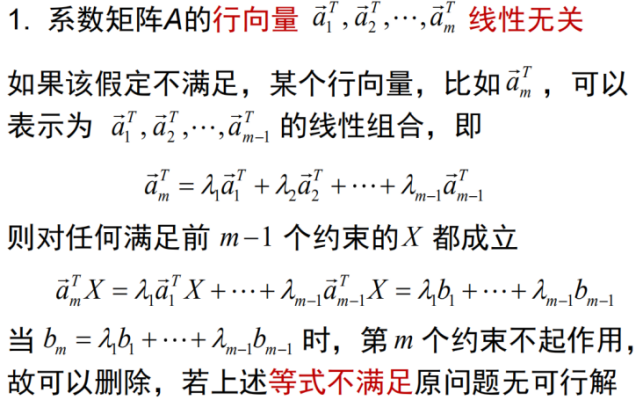
标准模型具有如下的矩阵形式：

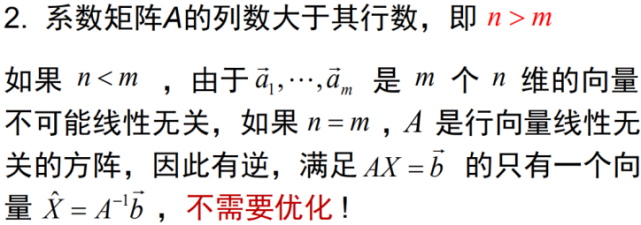


为方便进一步分析可引入两条基本假定：

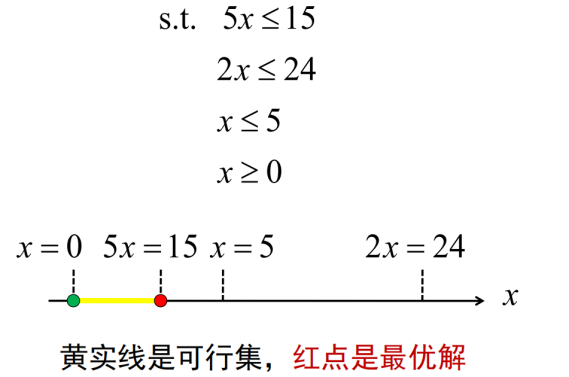
1.系数矩阵A的行向量线性无关

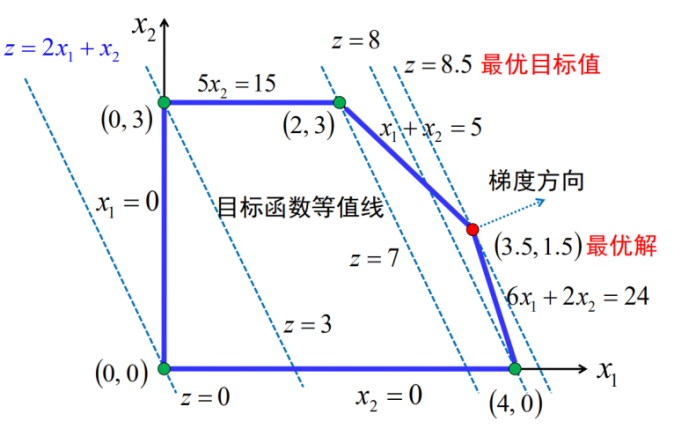
2.系数矩阵A的列数大于其行数，即 n＞m

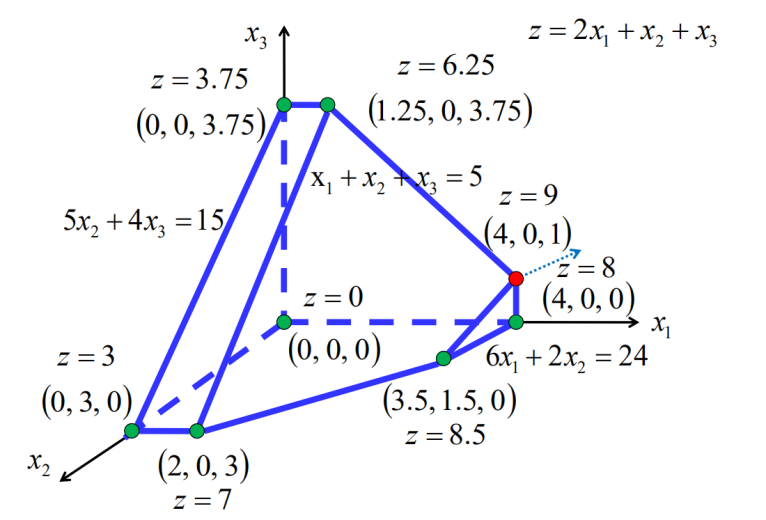


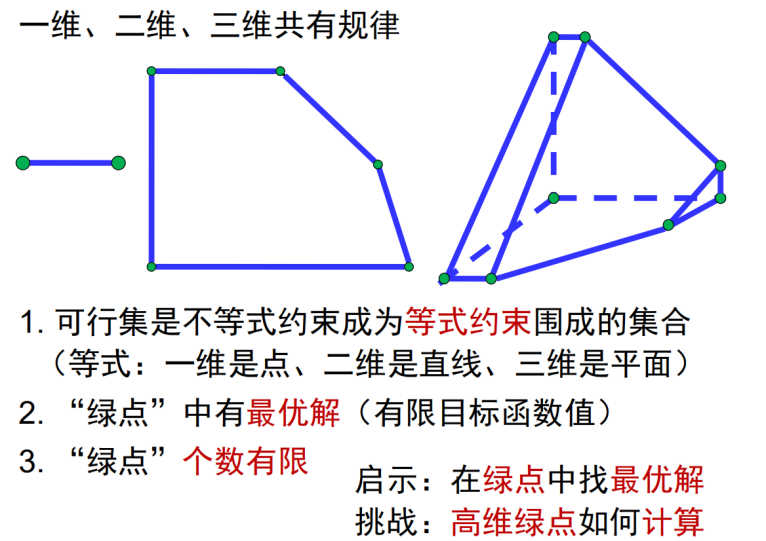


在解决低维的线性规划问题时，常用图解法，一维到三维的情况如所示：





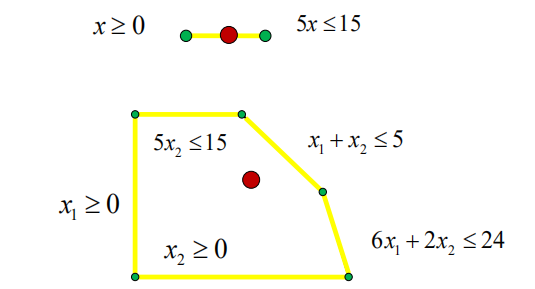


一维图解法中的线段（可行集）在高维时变成空间中的多面体形式，像这样的的图形被成为单纯形，或者说是空间中的凸集。由简单的的观察可得：线性规划的最优解总在单纯形的顶点处取得，所以要求解线性规划只需要寻找一种能遍历原问题可行集顶点的方法。

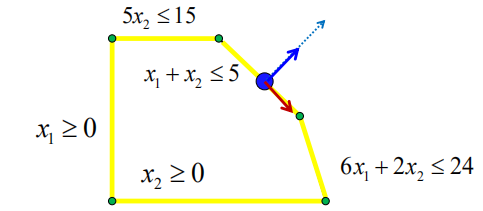
最优解总在单纯形的顶点处取得的详细证明需要运用矩阵的方法，然而可以借助目标函数的等值面给出简单的证明：

目标函数：min(or max)CTX，它的等值面CTX=f(X)是空间中的一组相互平行的平面，平面的法线方向便是目标函数值发生变化的梯度方向。

如果可行解为可行集内点，即没有起作用约束（可行解使得某约束条件等号成立），沿梯度方向必可改进目标函数，不可能是最优解，如红点所示：

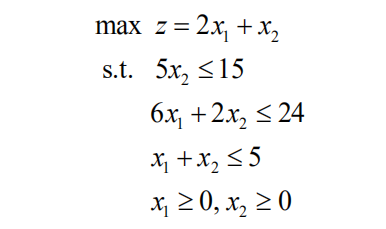


如果可行解不是起作用约束的唯一解，但梯度如下图所示，和所有起作用约束的法线垂直，此时蓝点是最优解，但沿着红色方向前进可以得到至少增加一个起作用约束的最优解，因此也可得到是最优解的绿点（有无穷多最优解）：

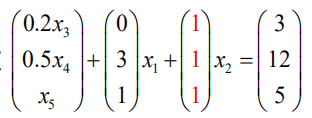


在高于三维的空间中运用图解法很不方便，但是借助于低维情况的分析，我们可以通过所谓“入基”、“出基”的手段遍历高维时的顶点：

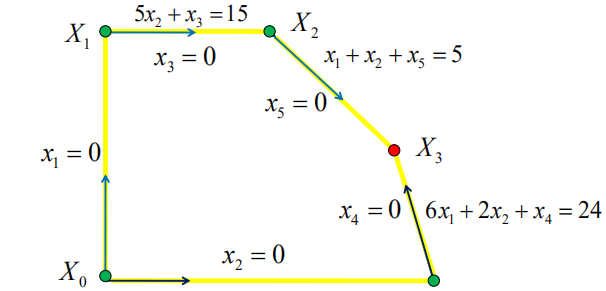
以如下问题为例：



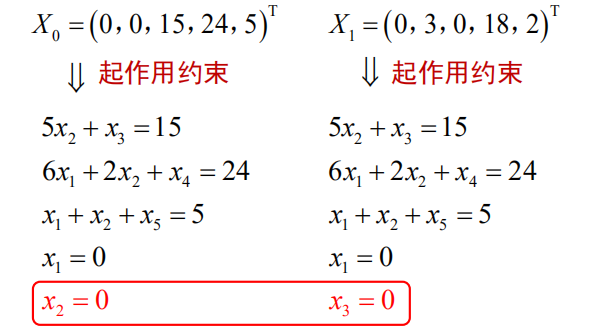
首先通过引入附加变量x3, x4, x5的方法转化为标准型问题，系数矩阵如下：



各约束条件及各边的起作用约束如下：



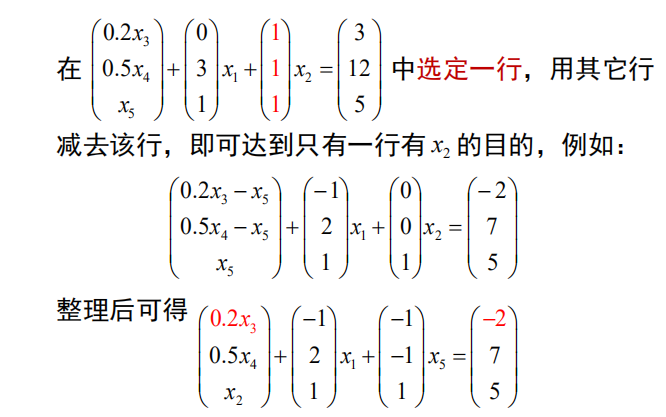
观察发现：

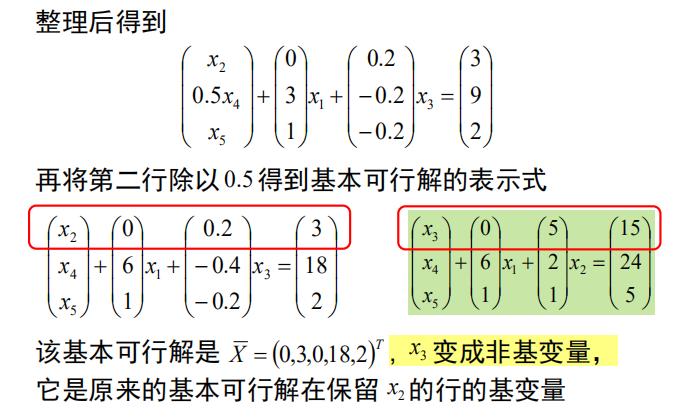


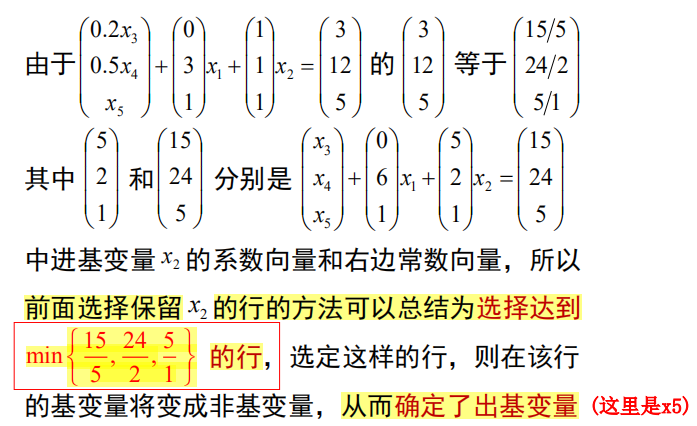
1. 相邻顶点间只有一个起作用约束不同（非基变量）

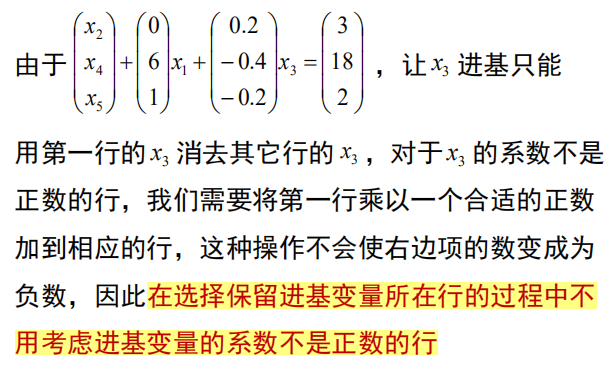
2. 相邻顶点搜索是变量的“进基”“出基”互换

于是我们可以引入下面所示的单纯形算法：

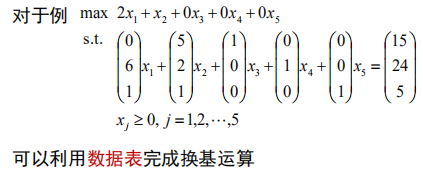


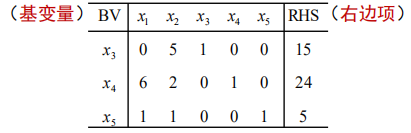


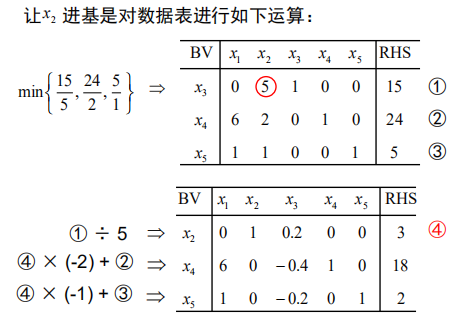


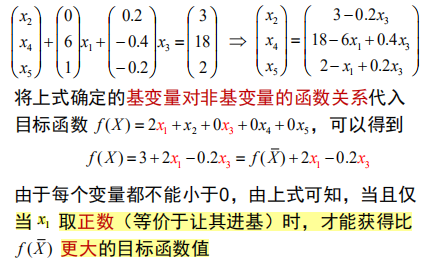


简单起见，有如下的单纯形表法：

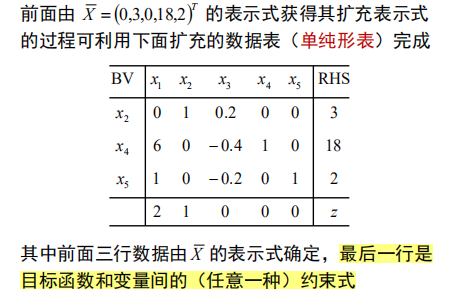


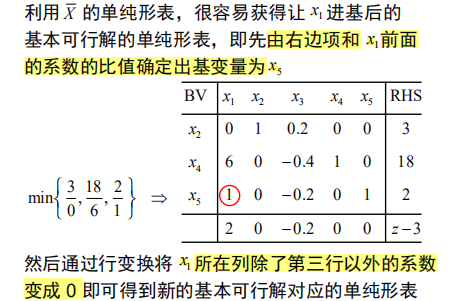


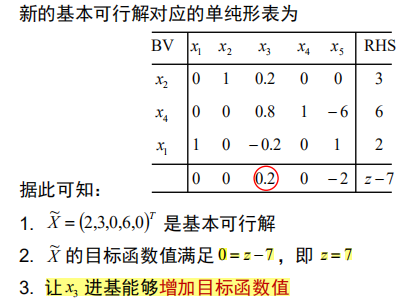




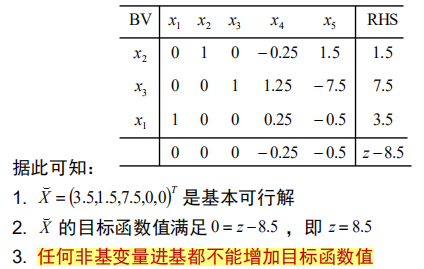
加入函数值的关系之后得到最终的单纯形表：

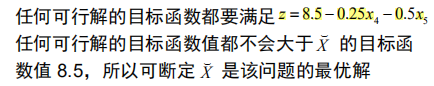




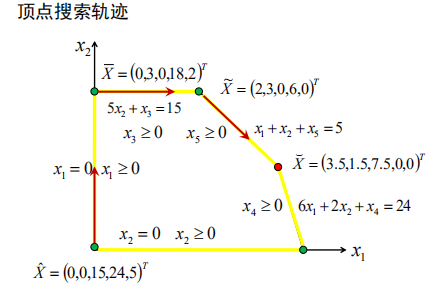


x3进基后新的基本可行解对应的单纯形表为

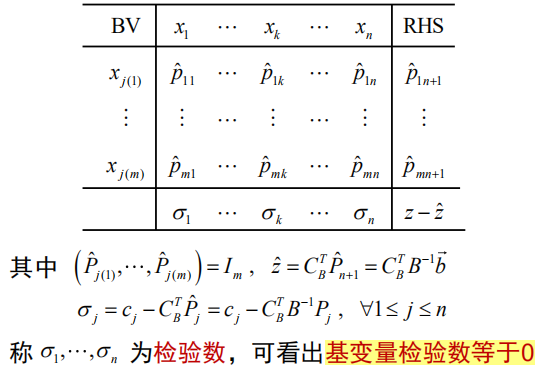


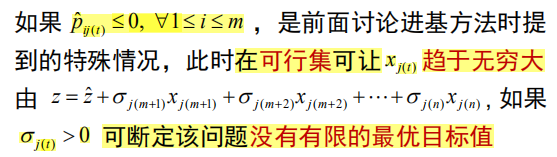


单纯形的“进基”、“出基”方法有着其明确的几何含义，具体而言，上述的搜索轨迹如下图所示：

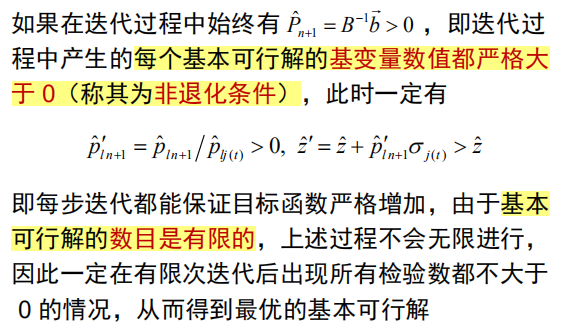


一般而言，单纯表方法的每一步都有如下的表格：

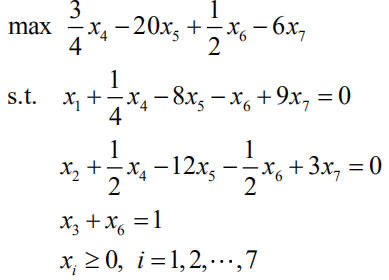


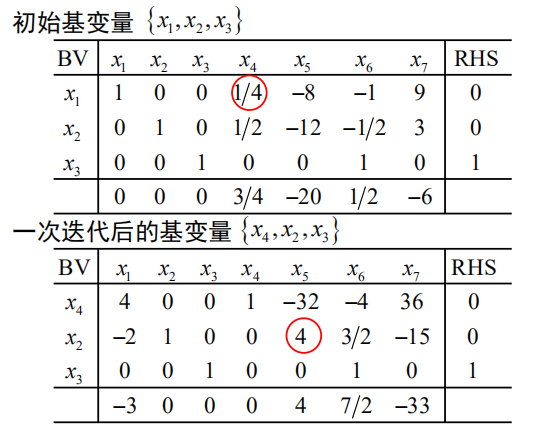


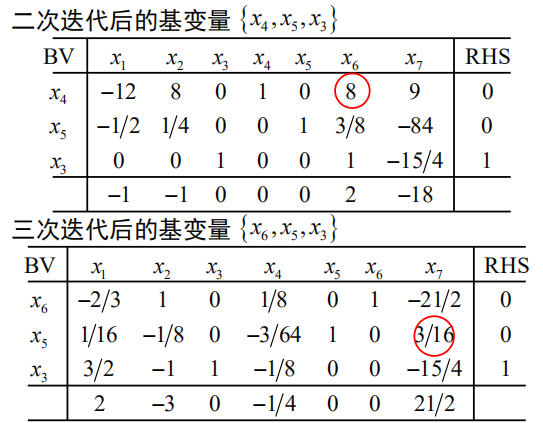
另一方面，单纯形算法的收敛性有如下的充分条件

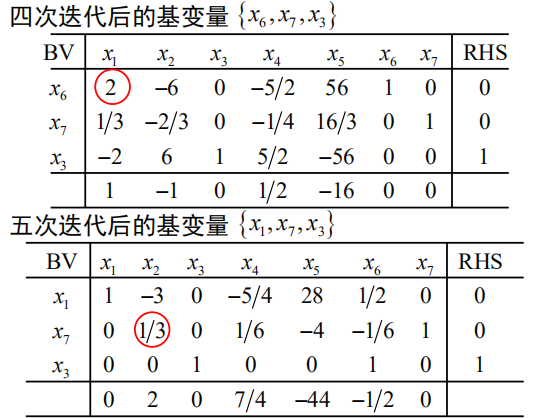


然而，某些退化情况依然会导致导致采用最大检验数规则的单纯形算法不收敛：



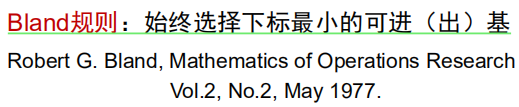






整个迭代过程中，虽然可行基矩阵不断改变，但对应的基本可行解始终是X=(0,0,1,0,0,0,0)T ，没有变化，目标函数值也一直没有变化，出现循环。

退化情况的本质是多个可行基阵对应于一个基本可行解。此时经过一次进出基迭代后得到的是同一个基本可行解，因此有可能出现迭代算法在一个基本可行解的几个基阵之间循环不止的情况只要设法避免回到已经搜索过的基阵，就可以保证算法有限步内停止。比方说如下的Bland规则：

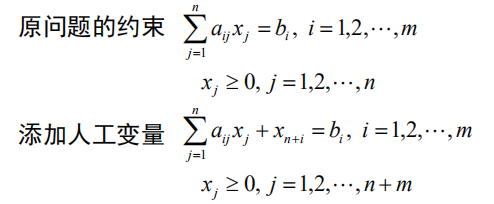


从任意基本可行解出发，采用Bland规则进行单纯形法迭代，在有限次迭代后停止于以下两种情况之一：

1、得到一个最优的基本可行解；

2、确定目标函数没有有限的最优值

单纯形法的迭代过程总需要从一个基本可行解（单纯形的顶点）开始，确定初始解的基本方法是添加人工变量，通过在迭代过程中把这些变量换出可行基获得原问题可行基。



具体算法有大M法、两阶段法等。

