数学建模算法与应用

第12章 现代优化算法

现代优化算法是上世纪 80 年代初兴起的启发式算法。这些算法包括禁忌搜索（tabu search），模拟退火（simulated annealing），遗传算法（genetic algorithms），人工神经网络（neural networks）。它们主要用于解决大量的实际应用问题。目前，这些算法在理论和实际应用方面得到了较大的发展。

无论这些算法是怎样产生的，它们有一个共同的目标——求 NP-hard 组合优化问题的全局最优解。虽然有这些目标，但 NP-hard 理论限制它们只能以启发式的算法去求解实际问题。

启发式算法包含的算法很多，例如解决复杂优化问题的蚁群算法（Ant Colony Algorithms）。有些启发式算法是根据实际问题而产生的，如解空间分解、解空间的限制等；另一类算法是集成算法，这些算法是诸多启发式算法的合成。

现代优化算法解决组合优化问题，如 TSP（Traveling Salesman Problem）问题，QAP（Quadratic Assignment Problem）问题，JSP（Job-shop Scheduling Problem）问题等效果很好。

12.1　模拟退火算法

12.1.1　算法简介

模拟退火算法得益于材料统计力学的研究成果。统计力学表明材料中粒子的不同结构对应于粒子的不同能量水平。在高温条件下，粒子的能量较高，可以自由运动和重新排列。在低温条件下，粒子能量较低。如果从高温开始，非常缓慢地降温（这个过程被称为退火），粒子就可以在每个温度下达到热平衡。当系统完全被冷却时，最终形成处于低能状态的晶体。

如果用粒子的能量定义材料的状态，Metropolis 算法用一个简单的数学模型描述了退火过程。假设材料在状态i之下的能量为$E(i)$，那么材料在温度T时从状态i进入状态j就遵循如下规律

（1）如果$E(j) \leq E(i)$，接受该状态被转换。

（2）如果$E(j) > E(i)$，则状态转换以如下概率被接受

$$e^{\frac{E(i)-E(j)}{KT}},$$

其中K是物理学中的波尔兹曼常数，T是材料温度。

在某一个特定温度下，进行了充分的转换之后，材料将达到热平衡。这时材料处于状态i的概率满足波尔兹曼分布

$$P_T(X=i)=\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)}{KT}}},$$

其中X表示材料当前状态的随机变量，S表示状态空间集合。

显然

$$\lim_{T\to\infty}\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)}{KT}}}=\frac{1}{|S|},$$

其中$|S|$表示集合S中状态的数量。这表明所有状态在高温下具有相同的概率。

而当温度下降时，

$$\lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)-E_{\min}}{KT}}}$$

$$=\lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}+\sum_{j\notin S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}}$$

$$=\lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}}=\begin{cases}\dfrac{1}{|S_{\min}|}, & \text{若} i\in S_{\min},\\ 0, & \text{其它.}\end{cases}$$

其中$E_{\min}=\min_{j\in S}E(j)$且$S_{\min}=\{i\mid E(i)=E_{\min}\}$。

上式表明当温度降至很低时，材料会以很大概率进入最小能量状态。

假定要解决的问题是一个寻找最小值的优化问题。将物理学中模拟退火的思想应用于优化问题就可以得到模拟退火寻优方法。

考虑这样一个组合优化问题：优化函数为 $f: x \to R^+$，其中 $x \in S$，它表示优化问题的一个可行解，$R^+ = \{y \mid y \in R, y \geq 0\}$，$S$ 表示函数的定义域。$N(x) \subseteq S$ 表示 x 的一个邻域集合。

首先给定一个初始温度T_0和该优化问题的一个初始解$x(0)$，并由$x(0)$生成下一个解$x' \in N(x(0))$，是否接受x'作为一个新解$x(1)$依赖于下面概率

$$P(x(0) \to x') = \begin{cases} 1, & \text{若} f(x') < f(x(0)), \\ e^{-\frac{f(x')-f(x(0))}{T_0}}, & \text{其它.} \end{cases}$$

换句话说，如果生成的解x'的函数值比前一个解的函数值更小，则接受$x(1) = x'$作为一个新解。否则以概率$e^{-\frac{f(x')-f(x(0))}{T_0}}$接受$x'$作为一个新解。

泛泛地说，对于某一个温度T_i和该优化问题的一个解$x(k)$，可以生成x'。接受x'作为下一个新解$x(k+1)$的概率为

$$P(x(k) \to x') = \begin{cases} 1, & \text{若} f(x') < f(x(k)), \\ e^{-\frac{f(x')-f(x(k))}{T_i}}, & \text{其它.} \end{cases} \tag{12.1}$$

在温度T_i下，经过很多次的转移之后，降低温度T_i，得到$T_{i+1} < T_i$。在T_{i+1}下重复上述过程。因此整个优化过程就是不断寻找新解和缓慢降温的交替过程。最终的解是对该问题寻优的结果。

注意到在每个T_i下，所得到的一个新状态$x(k+1)$完全依赖于前一个状态$x(k)$，和前面的状态$x(0),\cdots,x(k-1)$无关，因此这是一个马尔可夫过程。使用马尔可夫过程对上述模拟退火的步骤进行分析，结果表明从任何一个状态$x(k)$生成x'的概率，在$N(x(k))$中是均匀分布的，且新状态x'被接受的概率满足式（12.1），那么经过有限次的转换，在温度T_i下的平衡态x_i的分布由下式给出

$$P_i(T_i)=\frac{e^{-\frac{f(x_i)}{T}}}{\sum_{j\in S} e^{-\frac{f(x_i)}{T_i}}}. \qquad (12.2)$$

当温度T降为 0 时，x_i的分布为

$$P_i^* = \begin{cases} \dfrac{1}{|S_{\min}|}, & \text{若} x_i \in S_{\min}, \\ 0, & \text{其它}, \end{cases}$$

并且

$$\sum_{x_i \in S_{\min}} P_i^* = 1.$$

这说明如果温度下降十分缓慢，而在每个温度都有足够多次的状态转移，使之在每一个温度下达到热平衡，则全局最优解将以概率 1 被找到。因此可以说模拟退火算法可以找到全局最优解。

在模拟退火算法中应注意以下问题

（1）理论上，降温过程要足够缓慢，要使得在每一温度下达到热平衡。但在计算机实现中，如果降温速度过缓，所得到的解的性能会较为令人满意，但是算法会太慢，相对于简单的搜索算法不具有明显优势。如果降温速度过快，很可能最终得不到全局最优解。因此使用时要综合考虑解的性能和算法速度，在两者之间采取一种折衷。

（2）要确定在每一温度下状态转换的结束准则。实际操作可以考虑当连续m次的转换过程没有使状态发生变化时结束该温度下的状态转换。最终温度的确定可以提前定为一个较小的值T_e，或连续几个温度下转换过程没有使状态发生变化算法就结束。

（3）选择初始温度和确定某个可行解的邻域的方法也要恰当。

12.1.2　应用举例

已知 100 个目标的经度、纬度如表 12.1(表略)所示。我方有一个基地，经度和纬度为（70,40)。假设我方飞机的速度为 1000 公里/小时。我方派一架飞机从基地出发，侦察完所有目标，再返回原来的基地。在每一目标点的侦察时间不计，求该架飞机所花费的时间（假设我方飞机巡航时间可以充分长)。

这是一个旅行商问题。给我方基地编号为 1，目标依次编号为 2，3，…，101，最后我方基地再重复编号为 102（这样便于程序中计算）。距离矩阵 $\boldsymbol{D}=(d_{ij})_{102\times102}$，其中 d_{ij} 表示表示 i,j 两点的距离，$i,j=1,2,\cdots,102$，这里 $\boldsymbol{D}$ 为实对称矩阵。则问题是求一个从点 1 出发，走遍所有中间点，到达点 102 的一个最短路径。

上面问题中给定的是地理坐标（经度和纬度），必须求两点间的实际距离。设A,B两点的地理坐标分别为(x_1,y_1)，(x_2,y_2)，过A,B两点的大圆的劣弧长即为两点的实际距离。以地心为坐标原点O，以赤道平面为XOY平面，以0度经线圈所在的平面为XOZ平面建立三维直角坐标系。

则A,B两点的直角坐标分别为

$$A(R\cos x_1\cos y_1, R\sin x_1\cos y_1, R\sin y_1),$$

$$B(R\cos x_2\cos y_2, R\sin x_2\cos y_2, R\sin y_2),$$

其中$R=6370$为地球半径。A,B两点的实际距离

$$d = R\arccos\left(\frac{\overrightarrow{OA}\cdot\overrightarrow{OB}}{\left|\overrightarrow{OA}\right|\cdot\left|\overrightarrow{OB}\right|}\right),$$

化简得

$$d = R\arccos[\cos(x_1-x_2)\cos y_1\cos y_2+\sin y_1\sin y_2].$$

求解的模拟退火算法描述如下

（1）解空间

解空间 S 可表为 $\{1,2,\cdots,101,102\}$ 的所有固定起点和终点的循环排列集合，即

$$S=\{(\pi_1,\cdots,\pi_{102})\,|\,\pi_1=1,(\pi_2,\cdots,\pi_{101})\text{为}\{2,3,\cdots,101\}\text{的循环排列},\pi_{102}=102\}$$

其中每一个循环排列表示侦察 100 个目标的一个回路，$\pi_i=j$ 表示在第 $i-1$ 次侦察目标 j，初始解可选为 $(1,2,\cdots,102)$，本文中我们先使用 Monte Carlo（蒙特卡洛）方法求得一个较好的初始解。

（2）目标函数

目标函数（或称代价函数）为侦察所有目标的路径长度。要求

$$\min f(\pi_1,\pi_2,\cdots,\pi_{102})=\sum_{i=1}^{101} d_{\pi_i\pi_{i+1}},$$

而一次迭代由下列三步构成

（3）新解的产生

设上一步迭代的解为

$$\pi_1 \cdots \pi_{u-1}\pi_u\pi_{u+1} \cdots \pi_{v-1}\pi_v\pi_{v+1} \cdots \pi_{w-1}\pi_w\pi_{w+1} \cdots \pi_{102}。$$

i） 2 变换法

任选序号u,v，交换u与v之间的顺序，变成逆序，此时的新路径为

$$\pi_1 \cdots \pi_{u-1}\pi_v\pi_{v-1} \cdots \pi_{u+1}\pi_u\pi_{v+1} \cdots \pi_{102}.$$

ii） 3 变换法

任选序号u,v和w，将u和v之间的路径插到w之后，对应的新路径为

$$\pi_1\cdots\pi_{u-1}\pi_{v+1}\cdots\pi_w\pi_u\cdots\pi_v\pi_{w+1}\cdots\pi_{102}.$$

（4）代价函数差

对于 2 变换法，路径差可表示为

$$\Delta f = (d_{\pi_{u-1}\pi_v} + d_{\pi_u\pi_{v+1}}) - (d_{\pi_{u-1}\pi_u} + d_{\pi_v\pi_{v+1}}).$$

（5）接受准则

$$P=\begin{cases}1, & \Delta f<0,\\ \exp(-\Delta f/T), & \Delta f\geq 0.\end{cases}$$

如果$\Delta f<0$，则接受新的路径。否则，以概率$\exp(-\Delta f/T)$接受新的路径，即用计算机产生一个[0,1区间上均匀分布的随机数 rand，若 rand$\leq\exp(-\Delta f/T)$则接受。

（6）降温

利用选定的降温系数α进行降温，取新的温度T为αT（这里T为上一步迭代的温度），这里选定$\alpha = 0.999$。

（7）结束条件

用选定的终止温度$e = 10^{-30}$，判断退火过程是否结束。若$T < e$，算法结束，输出当前状态。

计算结果为 44 小时左右。其中的一个巡航路径如图 12.1 所示。

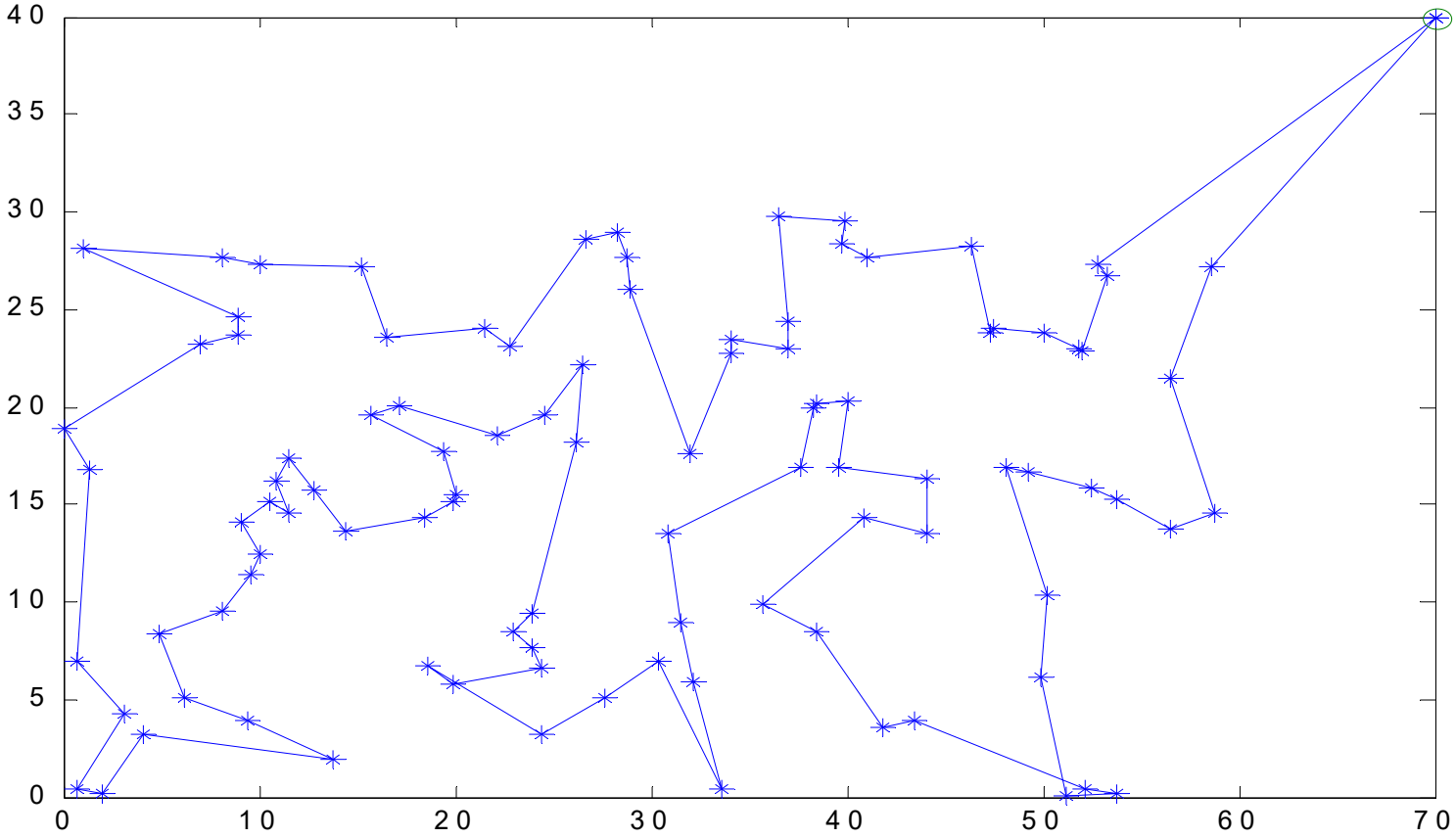

图 12.1　模拟退火算法求得的巡航路径示意图

12.2　遗传算法

12.2.1　遗传算法简介

遗传算法（Genetic Algorithms，简称 GA）是一种基于自然选择原理和自然遗传机制的搜索（寻优）算法，它是模拟自然界中的生命进化机制，在人工系统中实现特定目标的优化。遗传算法的实质是通过群体搜索技术，根据适者生存的原则逐代进化，最终得到最优解或准最优解。

它必须做以下操作：初始群体的产生、求每一个体的适应度、根据适者生存的原则选择优良个体、被选出的优良个体两两配对，通过随机交叉其染色体的基因并随机变异某些染色体的基因生成下一代群体，按此方法使群体逐代进化，直到满足进化终止条件。其实现方法如下

（1）根据具体问题确定可行解域，确定一种编码方法，能用数值串或字符串表示可行解域的每一解。

（2）对每一解应有一个度量好坏的依据，它用一函数表示，叫做适应度函数，一般由目标函数构成。

（3）确定进化参数群体规模M、交叉概率p_c、变异概率p_m、进化终止条件。

为便于计算，一般来说，每一代群体的个体数目都取相等。群体规模越大、越容易找到最优解，但由于受到计算机的运算能力的限制，群体规模越大，计算所需要的时间也相应地增加。进化终止条件指的是当进化到什么时候结束，它可以设定到某一代进化结束，也可以根据找出近似最优解是否满足精度要求来确定。表 12.2 列出了生物遗传概念在遗传算法中的对应关系。

表 12.2　生物遗传概念在遗传算法中的对应关系

生物遗传概念	遗传算法中的作用
适者生存	算法停止时，最优目标值的可行解有最大的可能被留住
个体	可行解
染色体	可行解的编码
基因	可行解中每一分量的特征
适应性	适应度函数值
种群	根据适应度函数值选取的一组可行解
交配	通过交配原则产生一组新可行解的过程
变异	编码的某一分量发生变化的过程

12.2.2　模型及算法

用遗传算法研究 12.1.2 中的问题。

求解的遗传算法的参数设定如下

种群大小$M=50$；最大代数$G=1000$；

交叉率$p_c=1$，交叉概率为 1 能保证种群的充分进化；

变异率$p_m=0.1$，一般而言，变异发生的可能性较小。

（1）编码策略

采用十进制编码，用随机数列$\omega_1\omega_2\ldots\omega_{102}$作为染色体，其中$0 \le \omega_i \le 1$（$i = 2,3,\cdots,101$），$\omega_1 = 0$，$\omega_{102} = 1$；每一个随机序列都和种群中的一个个体相对应，例如9 目标问题的一个染色体为

[0.23，0.82，0.45，0.74，0.87，0.11，0.56，0.69，0.78]，

其中编码位置i代表目标i，位置i的随机数表示目标i在巡回中的顺序，将这些随机数按升序排列得到如下巡回

6－1－3－7－8－4－9－2－5.

（2）初始种群

先利用经典的近似算法—改良圈算法求得一个较好的初始种群。

对于随机产生的初始圈

$$C = \pi_1 \cdots \pi_{u-1} \pi_u \pi_{u+1} \cdots \pi_{v-1} \pi_v \pi_{v+1} \cdots \pi_{102},$$

$$2 \le u < v \le 101, \quad 2 \le \pi_u < \pi_v \le 101,$$

交换u与v之间的顺序，此时的新路径为

$$\pi_1 \cdots \pi_{u-1} \pi_v \pi_{v-1} \cdots \pi_{u+1} \pi_u \pi_{v+1} \cdots \pi_{102}$$

记$\Delta f=(d_{\pi_{u-1}\pi_v}+d_{\pi_u\pi_{v+1}})-(d_{\pi_{u-1}\pi_u}+d_{\pi_v\pi_{v+1}})$，若$\Delta f<0$，则以新路经修改旧路经，直到不能修改为止，就得到一个比较好的可行解。

直到产生M个可行解，并把这M个可行解转换成染色体编码。

（3）目标函数

目标函数为侦察所有目标的路径长度，适应度函数就取为目标函数。我们要求

$$\min f(\pi_1,\pi_2,\cdots,\pi_{102}) = \sum_{i=1}^{101} d_{\pi_i\pi_{i+1}}$$

（4）交叉操作

交叉操作采用单点交叉。设计如下，对于选定的两个父代个体 $f_1=\omega_1\omega_2\ldots\omega_{102}$，$f_2=\omega'_1\omega'_2\ldots\omega'_{102}$，我们随机地选取第$t$个基因处为交叉点，则经过交叉运算后得到的子代个体为s_1和s_2，s_1的基因由 f_1的前t个基因和 f_2的后$102-t$个基因构成，s_2的基因由 f_2的前t个基因和 f_1的后$102-t$个基因构成，

例如

$f_1 = [0,\ \ 0.14,\ \ 0.25,\ \ 0.27,|\ \ 0.29,\ \ 0.54,\cdots,0.19,\ \ 1]$

$f_2 = [0,\ \ 0.23,\ \ 0.44,\ \ 0.56,|\ \ 0.74,\ \ 0.21,\cdots,0.24,\ \ 1]$

设交叉点为第四个基因处，则

$s_1 = [0,\ \ 0.14,\ \ 0.25,\ \ 0.27,|\ \ 0.74,\ \ 0.21,\cdots,0.24,\ \ 1]$

$s_2 = [0,\ \ 0.23,\ \ 0.44,\ \ 0.56,|\ \ 0.29,\ \ 0.54,\cdots,0.19,\ \ 1]$

交叉操作的方式有很多种选择，应该尽可能选取好的交叉方式，保证子代能继承父代的优良特性。同时这里的交叉操作也蕴含了变异操作。

（5）变异操作

变异也是实现群体多样性的一种手段，同时也是全局寻优的保证。具体设计如下，按照给定的变异率，对选定变异的个体，随机地取三个整数，满足$1<u<v<w<102$，把u,v之间（包括u和v）的基因段插到w后面。

（6）选择

采用确定性的选择策略，也就是说在父代种群和子代种群中选择目标函数值最小的M个个体进化到下一代，这样可以保证父代的优良特性被保存下来。

计算结果为 40 小时左右。其中的一个巡航路径如图 12.2 所示。

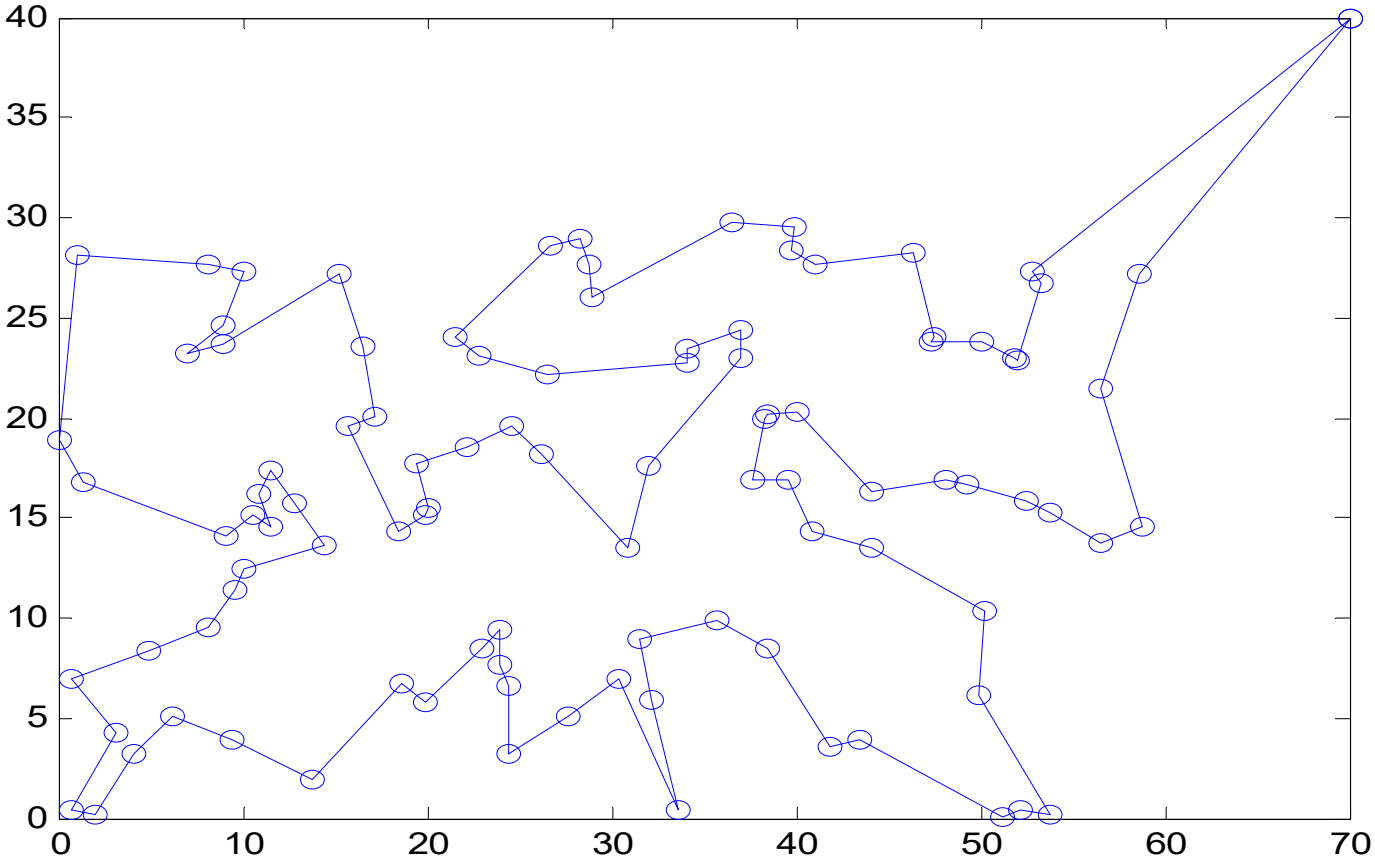

图 12.2　遗传算法求得的巡航路径示意图

12.3 改进的遗传算法

12.3.1 引言

无人机航路规划问题实际上是一个组合优化问题，是优化理论中的 NP—hard 问题。因为其解空间不连续，解邻域表达困难，所以难以用通常的算法求解。遗传算法作为现代优化算法之一，其主要特点是对非线性极值问题能以概率 1 跳出局部最优解，找到全局最优解。

而遗传算法这种跳出局部最优寻找全局最优特性都基于算法中的交叉和变异。在传统遗传算法的结构中，变异操作在交叉操作基础上进行，强调的是交叉作用，认为变异只是一个生物学背景机制。在具体交叉操作中，人们通常采用单点交叉（段交叉)、多点交叉与均匀交叉，其中单点交叉是指随机地在基因序列中选择一个断点，然后交换双亲上断点右端的所有染色体。在变异操作中，变异算子一般是用 Guassian 分布的随机变异来实现[46,47]。

近年来，也有学者尝试用 Cauchy 分布的随机序列来实现变异[48]，希望通过 Cauchy 分布宽大的两翼特性实现更大范围的变异，以利于找到全局最优解。Rudolph G 从理论上分析了采用 Cauchy 分布随机变异进化算法的局部收敛性[49]。Chellapilla K 进一步把二者结合起来[50]，采用两种分布的线性叠加，但仿真结果显示，算法改进效果并不十分明显。文献[51]将生物进化看成是随机性加上反馈，并指出其中的随机性主要是由系统的内在因素所引起，而不是由外部环境的随机扰动所造成。

而混沌系统在其混沌域中表现为随机性，它是确定系统内部随机性的反映，不同于外在的随机特性。本节根据以上特点对基于求解航路规划的遗传算法进行改进，首先将变异操作从交叉操作中分离出来，使其成为独立的并列于交叉的寻优操作，在具体遗传操作中，混沌与遗传操作联系在一起，在交叉操作中，以“门当户对”原则进行个体的配对，利用混沌序列确定交叉点，实行强度最弱的单点交叉，以确保算法收敛精度，削弱和避免寻优抖振问题；在变异操作中，利用混沌序列对染色体中多个基因进行变异，以避免算法早熟。

下面研究 12.1.2 中同样的问题。

12.3.2 模型及算法

与标准的遗传算法相比，本节做了如下的两点改进。

（1）交叉操作

本节的交叉操作采用改进型交叉。具体设计如下，首先以“门当户对”原则，对父代个体进行配对，即对父代以适应度函数（目标函数）值进行排序，目标函数值小的与小的配对，目标函数值大的与大的配对。然后利用混沌序列确定交叉点的位置，最后对确定的交叉项进行交叉。

例如(Ω_1,Ω_2)配对，他们的染色体分别是

$$\Omega_1=\omega_1^1\omega_2^1\ldots\omega_{102}^1，\ \Omega_2=\omega_1^2\omega_2^2\ldots\omega_{102}^2，$$

采用 Logistic 混沌序列$x(n+1)=4x(n)(1-x(n))$产生一个 2 到 101 之间的正整数，具体步骤如下：

取一个$(0,1)$区间上的随机数作为初始值，然后利用$x(n+1)=4x(n)(1-x(n))$迭代一次产生 1 个$(0,1)$区间上的混沌值，保存以上混沌值作为产生下一代交叉项的混沌迭代初值，再把这个值分别乘以 100 并加上 2，最后取整即可。假如这个数为 33，那么以此做为交叉点对(Ω_1,Ω_2)染色体中相应的基因进行单点交叉，

得到新的染色体$(\Omega_1^{'},\Omega_2^{'})$，

$$\Omega_1^{'}=\omega_1^1\omega_2^1\omega_3^1\omega_4^1\omega_5^1\cdots\omega_{33}^2\omega_{34}^1\cdots\omega_{60}^1\omega_{61}^1\cdots,$$

$$\Omega_2^{'}=\omega_1^2\omega_2^2\omega_3^2\omega_4^2\omega_5^2\cdots\omega_{33}^1\omega_{34}^2\cdots\omega_{60}^2\omega_{61}^2\cdots.$$

很明显这种单点交叉对原来的解改动很小，这可以削弱避免遗传算法在组合优化应用中产生的寻优抖振问题，可以提高算法收敛精度。

（2）变异操作

变异也是实现群体多样性的一种手段，是跳出局部最优，全局寻优的重要保证。这里变异算子设计如下，首先根据给定的变异率（本节选为 0.02），随机地取两个在 2 到 101 之间的整数，对这两个数对应位置的基因进行变异，变异时利用混沌序列把这两个位置的基因换成新的基因值，从而得到新的染色体。

12.3.3 仿真结果对比及算法性能分析

在仿真试验中，对本节航路规划问题分别利用单点交叉和换位变异结合的遗传算法，多点交叉和移位变异结合的遗传算法和本节中提出的改进算法进行求解比较。表 12.3 是各种算法种群规模（$M=50$）和迭代次数（$G=100$）都相同时连续 20 次求解的平均值（千米），算法平均运算时间（秒）。

表 12.3　算法性能比较表

指标	单点交叉算法	多点交叉算法	文中改进算法
平均航路距离(千米)	41572	40416	39849
算法执行时间(秒)	5.937	6.125	2.985

本节从算法结构到具体的遗传操作都进行了改进，其中变异操作从交叉操作中分离出来，使得遗传算法也可以通过并行计算实现，提高算法实现效率。其次改进后的算法，分别采用变化强度不同的交叉操作和变异操作，其中交叉操作采用强度最弱的单点交叉，保证了算法收敛精度，削弱和避免算法因交叉强度大而产生的寻优抖振问题。

当然单一的单点交叉很容易使算法早熟，文中采用较大强度的多个基因变异正好解决早熟问题。从仿真结果可以看到改进后的算法效果较为明显。

12.4　Matlab 遗传算法工具

12.4.1　遗传算法与直接搜索概述

Matlab 中遗传算法与直接搜索(Genetic Algorithm and Direct Search，简称 GADS）工具箱是一系列函数的集合，它们扩展了优化工具箱和 Matlab 数值计算环境。遗传算法与直接搜索工具箱包含了要使用遗传算法和直接搜索算法来求解优化问题的一些例程。这些算法使我们能够求解那些标准优化工具箱范围之外的各种优化问题。

所有工具箱函数都是Matlab的M文件，这些文件由实现特定优化算法的Matlab语句所写成。

使用语句

Type function_name

就可以看到这些函数的 Matlab 代码。也可以通过编写自己的 M 文件来实现和扩展遗传算法和直接搜索工具箱的性能，也可以将该工具箱与 Matlab 的其它工具箱或 Simulink 结合使用来求解优化问题。

工具箱函数可以通过图形界面或Matlab命令行来访问，它们是用Matlab语言编写的，对用户开放，因此可以查看算法，修改源代码或生成用户函数。

遗传算法与直接搜索工具箱有助于求解那些不易用传统方法解决的问题，譬如旅行商问题等。

遗传算法与直接搜索工具箱有一个精心设计的用户图形界面，可以直观、方便、快速地求解最优化问题。

1．功能特点

遗传算法与直接搜索工具箱的功能特点如下

（1）用户图形界面和命令行函数可用来快速地描述问题、设置算法选项以及监控进程。

（2）具有多个选项的遗传算法工具可用于问题创建、适应度计算、选择、交叉和变异。

（3）直接搜索工具实现了一种模式搜索方法，其选项可用于定义网格尺寸、表决方法和搜索方法。

（4）遗传算法与直接搜索工具箱函数可与 Matlab 的优化工具箱或其它的 Matlab 程序结合使用。

（5）支持自动的 M 代码生成。

2．用户图形界面和命令行函数

遗传算法工具函数可以通过命令行和用户图形界面来使用遗传算法。直接搜索工具函数也可以通过命令行和用户图形界面来进行访问。用户图形界面可用来快速地定义问题，设置算法选项，对优化问题进行详细定义。

遗传算法和直接搜索工具箱还同时提供了用于优化管理、性能监控及终止准则定义的工具，同时还提供了大量的标准算法选项。

在优化运行的过程中，可以通过修改选项来细化最优解，更新性能结果。用户也可以提供自己的算法选项来定制工具箱。

3．使用其它函数和求解器

遗传算法与直接搜索工具箱和Matlab优化工具箱是紧密结合在一起的。用户可以用遗传算法或直接搜索算法来寻找最佳起始点，然后利用优化工具箱或用Matlab 程序来进一步寻找最优解。通过结合不同的算法，可以充分地发挥Matlab和工具箱的功能以提高求解的质量。对于某些特定问题，使用这种方法还可以得到全局（最优）解。

4．显示、监控和输出结果

遗传算法与直接搜索工具箱还包括一系列绘图函数，用来可视化优化结果。这些可视化功能直观地显示了优化的过程，并且允许在执行过程中进行修改。使用输出函数可以将结果写入文件，产生用户自己的终止准则，也可以写入用户自己的图形界面来运行工具箱求解器。除此之外，还可以将问题的算法选项导出，以便日后再将它们导入到图形界面中去。

12.4.2　使用遗传算法工具初步

12.4.2.1　遗传算法使用规则

遗传算法是一种基于自然选择、生物进化过程来求解问题的方法。在每一步中，遗传算法随机地从当前种群中选择若干个体作为父辈，并且使用它们产生下一代的子种群。在连续若干代之后，种群朝着优化解的方向进化。可以用遗传算法来求解各种不适宜于用标准算法求解的优化问题，包括目标函数不连续、不可微、随机或高度非线性的问题。

遗传算法在每一步使用下列三类规则从当前种群来创建下一代

（1）选择规则（Selection Rules）：选择对下一代种群有贡献的个体（称为父辈）。

（2）交叉规则（Crossover Rules）：将两个父辈结合起来构成下一代的子辈种群。

（3）变异规则（Mutation Rules）：施加随机变化给父辈个体来构成子辈。

遗传算法与标准优化算法主要在两个方面有所不同，它们的比较情况归纳于表 12.4 中。

表 12.4　遗传算法与标准优化算法比较

标准算法	遗传算法
每次迭代产生一个单点，点的序列逼近一个优化解	每次迭代产生一个种群，种群逼近一个优化解
通过确定性的计算在该序列中选择下一个点	通过随机进化选择计算来选择下一代种群

12.4.2.2　遗传算法使用方式

遗传算法工具有两种使用方式

（1）以命令行方式调用遗传算法函数 ga。

（2）通过用户图形界面使用遗传算法工具。

1．在命令行使用遗传算法，可以用下列语法调用遗传算法函数 ga

[x, fval]=

ga(@fitnessfun,nvars,A,b,Aeq,beq,LB,UB,@nonlcon,options)

其中@fitnessfun 是目标函数句柄，nvars 是目标函数中独立变量的个数，options 是一个包含遗传算法选项参数的数据结构，其它参数的含义与非线性规划 fmincon 中的参数相同。函数返回值 x 为最终值到达的点，这里 x 为行向量，fval 为目标函数的最终值。

例 12.1　求下列问题的解

$$\max\ f(x)=2x_1+3x_1^2+3x_2+x_2^2+x_3,$$

$$\text{s.t.}\begin{cases} x_1+2x_1^2+x_2+2x_2^2+x_3\le 10,\\ x_1+x_1^2+x_2+x_2^2-x_3\le 50,\\ 2x_1+x_1^2+2x_2+x_3\le 40,\\ x_1^2+x_3=2,\\ x_1+2x_2\ge 1,\\ x_1\ge 0,\quad x_2,x_3\text{不约束}. \end{cases}$$

解　（1）编写适应度函数（文件名为 ycfun1.m）

```
function y=ycfun1(x);      %x 为行向量
c1=[2 3 1]; c2=[3 1 0];
y= c1* x' + c2* x'.^2; y=-y;
```

（2）编写非线性约束函数（文件名为 ycfun2.m）

```
function [f,g]=ycfun2(x);
f=[x(1)+2*x(1)^2+x(2)+2*x(2)^2+x(3)-10
    x(1)+x(1)^2+x(2)+x(2)^2-x(3)-50
    2*x(1)+x(1)^2+2*x(2)+x(3)-40];
g=x(1)^2+x(3)-2;
```

（3）主函数

```
clc, clear
a=[-1 -2 0;-1 0 0];b=[-1;0];
[x,y]=ga(@ycfun1,3,a,b,[],[],[],[],@ycfun2);
x, y=-y
```

遗传算法程序的运行结果每一次都是不一样的，要运行多次，找一个最好的结果。

2．通过 GUI 使用遗传算法

遗传算法工具有一个用户图形界面 GUI，它使我们可以使用遗传算法而不用在命令行方式工作。遗传算法的用户图形界面集成在优化工具箱里，打开遗传算法用户图形界面，可键入以下命令

```
optimtool
```

使用遗传算法用户图形界面解法首先必须输入下列信息

（1）Fitness function（适应度函数）——欲求最小值的目标函数。输入适应度函数的形式为@fitnessfun，其中 fitnessfun.m 是计算适应度函数的 M 函数。符号@产生一个对于函数 fitnessfun 的函数句柄。

（2）Number of variables（变量个数）——适应度函数输入向量的长度。

其它参数的含义，就不一一介绍了，与优化工具的用户图形界面解法中的参数一样。如果某个参数的值为空，可以输入空矩阵[]，或者什么都不输入。

例 12.2　用用户图形界面求解例 12.1。

在 Matlab 命令窗口运行 optimtool，打开用户图形界面，如图 12.3 所示，填入有关的参数，未填入的参数取值为空或者为默认值，然后用鼠标点一下“start”按钮，就得到求解结果，再使用“file”菜单下的“Export to Workspace…”选项，把计算结果输出到 Matlab 工作空间中去。

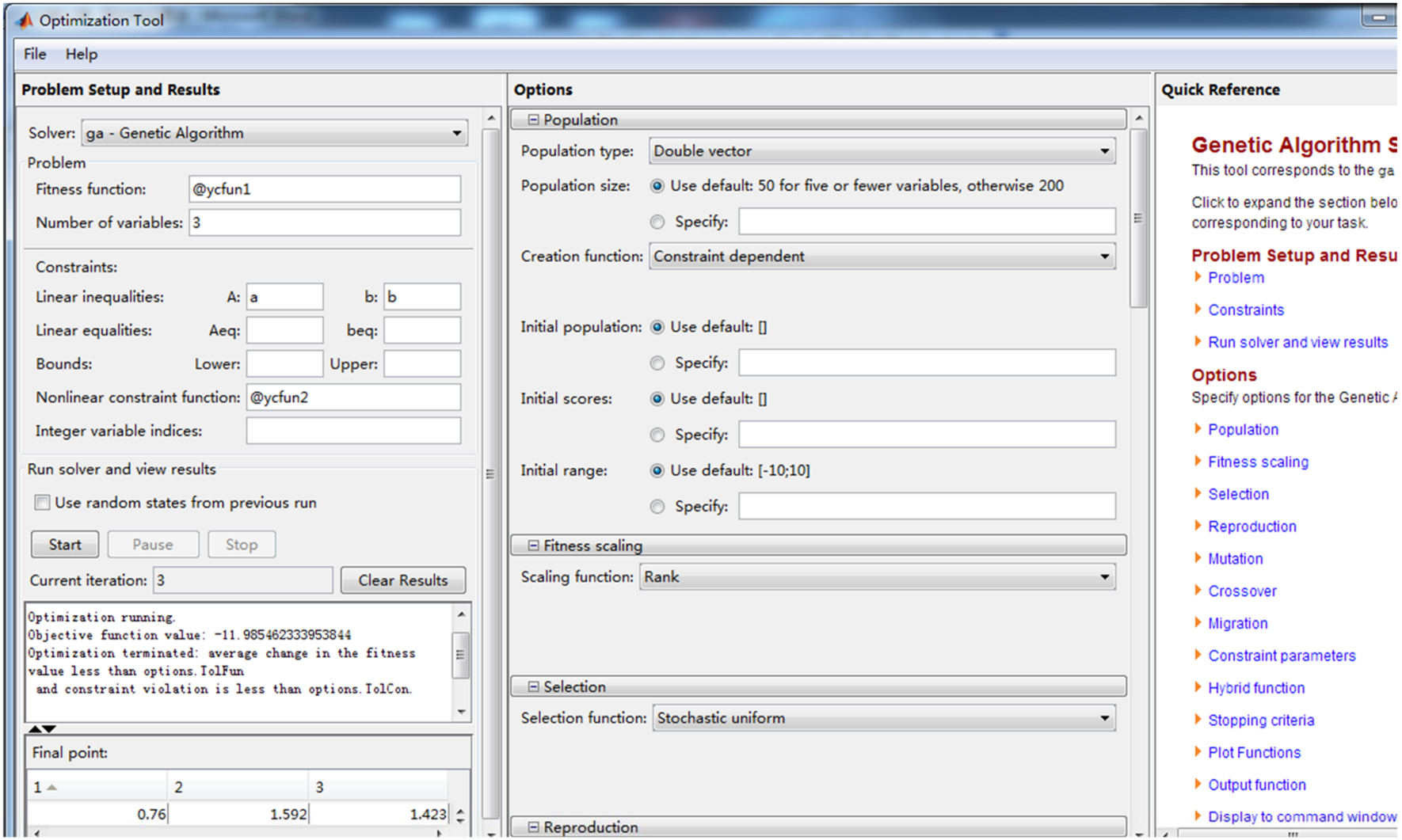

图 12.3　遗传算法用户图形界面解法

注意：Matlab 工作空间中必须存在线性不等式约束对应的变量 a，b。或者编一个小程序，对 a,b 进行赋值，然后运行该小程序。

12.4.3　直接搜索工具

直接搜索的命令为

[x,fval] = patternsearch(@fun,x0,A,b,Aeq,beq,LB,UB,@nonlcon,options)

直接搜索的用户图形界面也集成到 optimtool 中。

例 12.3　用直接搜索算法求解例 12.1。

解　编写程序如下（所有程序放在一个文件中）

```
function ex12_3
a=[-1 -2 0;-1 0 0];b=[-1;0];
[x,y]=patternsearch(@fun1,rand(1,3),a,b,[],[],[],[],@fun2);　%初始值必须为行向量
x,y=-y
```

```
%定义目标函数
function y=fun1(x);      %x 为行向量
c1=[2 3 1]; c2=[3 1 0];
y= c1* x' + c2* x'.^2; y=-y;
%定义非线性约束函数
function [f,g]=fun2(x);
f=[x(1)+2*x(1)^2+x(2)+2*x(2)^2+x(3)-10
      x(1)+x(1)^2+x(2)+x(2)^2-x(3)-50
      2*x(1)+x(1)^2+2*x(2)+x(3)-40];
g=x(1)^2+x(3)-2;
```

直接搜索算法每次的计算结果也是不一样的。