第 5 章

CHAPTER 5

模拟退火算法

模拟退火(Simulated Annealing,SA)算法是一种仿物理学现象的群体智能算法,其概念最早由 Metropolis 等于 1953 年提出,Kirkpatrick 于 1983 年首次使用模拟退火算法求解组合优化问题。模拟退火算法是一种基于 Monte Carlo 迭代求解策略的随机优化方法,其能够为具有 NP 复杂性的优化问题提供有效的近似最优解,它克服了其他优化过程容易陷入局部最优的缺点和对初始值的依赖性。

目前,无论是理论研究还是应用研究,模拟退火算法都已足够成熟。尤其是它在应用研究领域的范围非常广泛,其在实际的工程应用中已得到了广泛使用,比如车间生产调度、机器学习、图像处理、控制工程、神经网络、模式识别、离散变量的组合优化问题等领域。

5.1 模拟退火算法的思想

5.1.1 退火现象

模拟退火算法的思想来源于热力学领域的退火现象,退火现象是指液体逐渐降温后形成结晶态的物理现象。当液体的温度较高时,组成液体的大量原子运动活跃,此时原子在彼此之间进行着相对自由的移动。随着温度的降低,液体的内能也逐渐降低,大量原子逐渐失去活跃性,逐渐趋向于有序排列,当液体达到最低能量状态时,液体便形成一个纯净的晶体。

晶体状态是能量最低的状态,但并不是所有冷却系统下都可以使液体达到这个能量最低状态。事实上,如果液态金属被迅速冷却,则它并不会达到这个状态,而只能达到一种处于较高能量的多晶体状态或非晶体状态,如图 5.1(a)所示。因此,退火过程的核心在于缓慢地降低温度,保证大量原子在丧失活跃性前能够有充足的时间进行有序的重新

分布，才能使液体达到晶体状态，如图 5.1(b)所示。简而言之，物理退火过程由加温过程、等温过程和冷却过程组成。

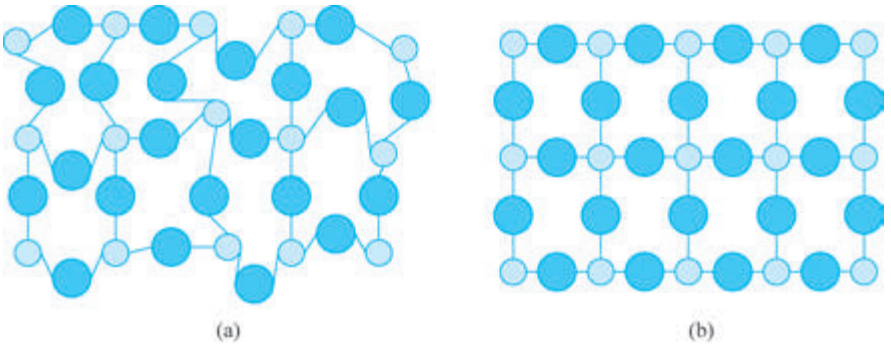

图 5.1　非晶体(a)与晶体(b)的原子排列结构

1. 加温过程

加温的目的是增强粒子的热运动，使其偏离平衡位置。当温度足够高时，固体将溶解为液体，从而消除系统原先可能存在的非均匀态，使其随后进行的冷却过程以某一平衡态为起点。溶解过程与系统的能量增大过程相联系，系统能量也随温度的升高而增大。

2. 等温过程

通过热力学知识可知，对于与周围环境交换热量而温度不变的封闭系统，系统状态的自发变化总是朝着自由能减小的方向进行，当自由能达到最小时，系统达到平衡态。

3. 冷却过程

冷却的目的是使粒子的热运动逐渐减弱并趋于有序，系统的能量逐渐下降，从而达到能量最低时的晶体状态。

5.1.2　Metropolis 准则

Metropolis 准则是一种重点抽样方法，其描述为：系统从状态 i 变化到另一个状态 j 时，如果相应的能量关系为 $E_j < E_i$，那么系统将接受状态 j；否则以概率 p 接受状态 j。概率 p 的计算方式如下：

$$p = \mathrm{e}^{-\frac{E_j - E_i}{kT}}$$

综上可知状态 i 变换到状态 j 的概率为

$$p(i \to j) = \begin{cases} 1, & E_j < E_i \\ e^{-\frac{E_j - E_i}{kT}}, & E_j \geqslant E_i \end{cases} = \begin{cases} 1, & E_j < E_i \\ e^{-\frac{|\Delta E|}{kT}}, & E_j \geqslant E_i \end{cases}$$

其中，T 为温度；E_i 和 E_j 分别为状态 i 和状态 j 时的能量，ΔE 为两者的差值；k 为 Boltzmann 常数。

5.1.3 算法原理

模拟退火算法与金属退火过程相似，它们之间的对应关系如表 5.1 所示。模拟退火算法是在一个初始温度下，微粒搜索当前状态的临近状态，判断各种状态之间的优劣程度，并使用一个接受准则(Metropolis 准则)确定是否改变当前状态。根据 Metropolis 准则，温度越高时，系统越容易接受新的状态，随着温度的下降，系统逐渐趋于稳定，改变状态的概率下降。

表 5.1 模拟退火算法与金属退火过程的对应关系

金 属 退 火	模 拟 退 火
粒子状态	问题的解
能量最低状态	最优解
溶解过程(加温过程)	设定初始温度
等温过程	Metropolis 采样过程
冷却操作	控制参数的下降
能量	目标函数

用固体冷却过程模拟求解组合优化问题的过程，将内能 E 模拟为目标函数值，温度 T 模拟为控制参数，即得到求解组合优化问题的模拟退火算法：由初始解 x 和初始控制参数值 T 开始，对当前解重复“产生新解→计算目标函数差值→接受或舍弃新解”的迭代，并逐步衰减 T 的数值，算法终止时的解即为问题的最优解或次最优解，这是一种基于蒙特卡洛(Monte Carlo)迭代求解法的启发式随机搜索过程。

温度是 Metropolis 算法的一个重要控制参数，模拟退火算法可视为递减控制参数 T 时的 Metropolis 算法迭代。迭代初期 T 较大，可以接受较差的问题解；迭代后期 T 较小，只能接受较好的问题解；最后 T 趋向于 0，此时不再接受任何比当前解更劣的解。T 的衰减越小，算法迭代的时间越长，不过可减小马尔可夫链的长度(等温条件下进行迭代优化的次数)，以使到达准平衡分布的时间变短。

假设开始状态为 A，迭代数次之后更新到最优解 B，这时候发现 B 点的能量要比 A 低，则说明接近最优解了，因此 100%概率进行状态转移。状态到达 B 点后，发现下一步能量上升，如果是梯度下降算法则是不允许继续向前的，而这里模拟退火算法会以一定

的概率跳出这个局部最优解，这个概率与当前的状态、能量和迭代的次数均有关系。如果跳出了 B 点来到了 C 点，又会继续以一定的概率跳出来，直到到达 D 点就会稳定下来，其原理如图 5.2 所示。

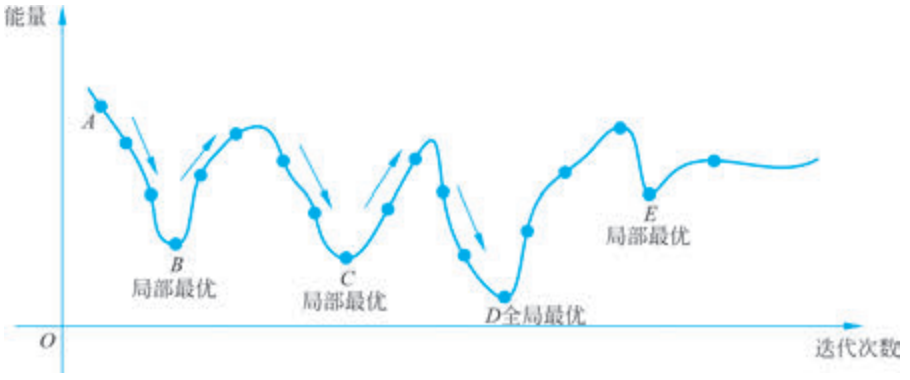

图 5.2 模拟退火算法寻优示意图

5.2 模拟退火算法的设计

使用模拟退火算法求解优化问题时，需要根据具体问题的特点来选择和调整关键参数和评估函数。不同的参数和评估函数可能对算法的性能和结果产生影响，因此需要进行实验和调优来找到最佳的设置。

1. 温度设置和终止准则

温度设置分为初始温度设置和终止温度设置。初始温度越大，获得最优解的概率越大，但是计算的时间也将增加。因此，初始温度的设置应该综合考虑算法的优化质量和优化效率。实际上终止温度与初始温度的差值、温度变换情况共同影响着算法迭代的次数和时间。

目前设置初始温度常见的方法有经验法、随机法和自适应法等。经验法就是根据使用者的经验或者启发式规则来确定初始温度，其操作相对简单，但是可能需要进行多次实验才能找到合适的初始温度。随机法通过随机生成一组初始解，然后计算它们的目标函数值，最后根据这些目标函数值来选择一个合适的初始温度。随机法可以根据实际的问题特点来确定初始温度，但是可能会占用更多的计算资源。自适应法是使用较高的初始温度进行搜索，然后根据搜索过程中的信息动态调整温度的方法，这种方法可以结合搜索情况调整温度下降的速度，平衡算法的全局搜索能力和局部开发能力。

设置终止温度的常见方法有固定终止温度、基于接受概率的终止温度和基于迭代次

数的终止温度等方法。固定终止温度就是在算法开始之前，预先设定一个确定的终止温度，当温度下降到终止温度以下时，算法便停止搜索，这种方法简单直观，但是也需要多次实验来找到合适的终止温度。基于接受概率的终止温度就是设置一个接受概率的阈值作为终止条件。当算法在某个温度下的接受概率低于设定的阈值时，可以认为搜索已经足够接近最优解，此时停止搜索，而这个温度便是终止温度。基于迭代次数的终止温度是指算法执行的迭代次数达到设定值时，停止搜索，停止时的温度便是终止温度。

2. 状态产生函数

状态产生函数的作用是在当前解的基础上产生候选解，设计这个函数时应该保证产生的候选解能够尽可能地遍布全部解空间。一般情况下，状态产生函数由两部分组成，即产生候选解的方式和产生候选解的概率分布。候选解的产生方式由问题的性质决定，一般是在当前邻域结构内以一定的概率产生，而邻域函数和概率方式可以多样化设计；候选解的概率分布可以是正态分布、均匀分布、指数分布和伽马分布等。

3. 状态接受函数

状态接受函数一般以概率的方式给出，不同接受函数的差别主要在于接受概率的形式不同，不过它们的设置均遵循以下原则：在固定的温度下，接受目标函数值更优的候选解概率要大于目标函数值更劣的候选解概率；随着温度的下降，接受目标函数值更劣的候选解概率要逐渐减小；当温度趋向于零时，只能接受目标函数值更优的候选解。

4. 温度更新函数

温度更新函数也被称为退温函数，其决定了温度下降的方式，目前最常用的温度更新函数为指数退温函数，即 $T(t)=\beta^t \cdot T_0$。其中 $T(t)$ 为迭代 t 次时的温度；T_0 为初始温度；β 为退火率，其取值范围为 $0<\beta<1$，其大小可以根据实际情况不断发生变化。

5. 马尔可夫链长度 L

马尔可夫(Markov)链长度也就是算法在等温条件下进行迭代优化的次数，L 通常取为问题规模 n 的一个多项式函数。用接受和拒绝的比率来控制 L，当温度很高时，L 应该尽量小，随着温度的逐渐下降，L 应该逐渐增大。

5.3 模拟退火算法的基本框架

模拟退火算法的实现流程如图 5.3 所示。

具体的步骤如下。

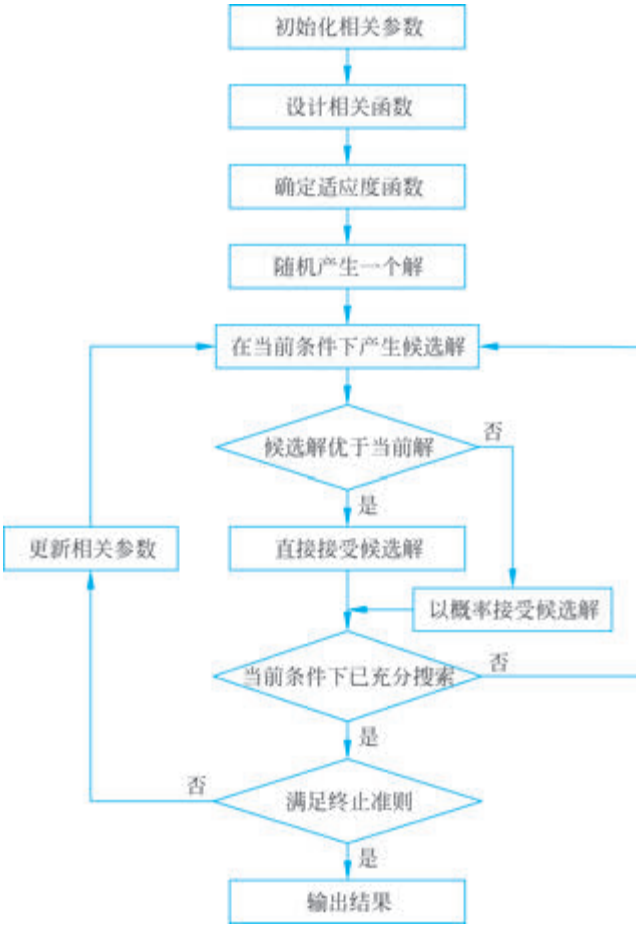

图 5.3 模拟退火算法的实现流程

步骤 1：设置初始温度、终止温度、退火率和马尔可夫链长度。

步骤 2：设计状态产生函数、状态接受函数和温度更新函数。

步骤 3：确定适应度函数，一般将目标函数作为适应度函数。

步骤 4：在解空间中随机产生一个解 s。

步骤 5：在当前温度下，根据状态产生函数，在当前解的邻域中产生候选解 s^*。

步骤 6：计算当前解和候选解 s^* 对应的适应度值并比较两者的数值大小。

步骤 7：根据状态接收函数判断是否接受候选解 s^*，若候选解优于当前解，则直接接受候选解，否则以一定概率接受候选解。

步骤 8：判断是否完成马尔可夫链长度的搜索，是则跳到步骤 9，否则回到步骤 5。

步骤 9：判断是否满足终止条件，是则跳到步骤 10，否则更新相关参数，然后回到步骤 5。

步骤 10：输出优化结果。

5.4　模拟退火算法的改进策略

5.4.1　模拟退火算法存在的问题

模拟退火算法的应用很广泛，能够有效地求解复杂的组合优化问题，但是它也存在以下一些问题。

（1）模拟退火算法不能保证一次就收敛到最优值，一般需要进行多次尝试才能获得。

（2）温度设置与算法的执行效率相矛盾，要使搜索到最优解的概率增大就应该增大初始温度，但这同时也会使算法在迭代中花费大量的计算时间；反之，如果初始温度过低，虽然可以节省计算时间，但是搜索性能也将受到影响。

（3）随着优化问题维度的增加，模拟退火算法的求解效率会下降，使得搜索到最优解的概率减小。

（4）因为模拟退火算法有一定的概率接受劣解，所以算法在跳出局部最优解的同时也会丧失已经搜索到的最优解。如图 5.2 所示，当搜索到 D 点时，仍有一定的概率跳出这个全局最优点而来到点 E。

5.4.2　增加保留算子策略

模拟退火算法本身不具备记忆功能，所以为了避免搜索过程中由于执行概率接受环节而丢失当前遇到的最优解，可以通过增加存储环节，将到目前为止搜索到的最好状态保存下来。

其操作也比较简单，如图 5.4 所示，在算法迭代前，当前最优解就是算法产生的初始解，在迭代过程中，如果需要将候选解更新为新解，那么就比较当前解和候选解的适应

图 5.4　保留算子操作

度，如果候选解的适应度优于当前解，那么更新历史最优解为候选解，否则保持历史最优解不变，最终历史最优解便是算法迭代优化的输出结果。

5.4.3 多粒子寻优策略

将原始模拟退火算法中产生候选解时的“一次扰动”替换成“n 次扰动”，即在当前解的邻域中产生 n 个候选解，然后分别计算 n 个候选解的适应度值，将其中最优的候选解与当前解通过状态接受函数进行判断。

因为当前解邻域内的每一个点都可以作为候选解，如果 s^* 是该邻域内的一个可以使得目标函数达到最优的解，按照原始模拟退火算法的流程，可能需要循环 i 次才能够找到这个解。但是如果一次生成 n 个候选解，然后择优比较的话，找到 s^* 的概率将增大，相应的循环次数也可以减小。在保证马尔可夫链长度不变的情况下，生成解的质量将会有所提高。

5.4.4 重升温策略

由模拟退火算法的特点可知，如果温度很低，算法便很难再跳出局部最优解，此时算法可能会陷入无效的迭代，即迭代多次之后对优化结果仍没有影响。此时可以在保留这个局部最优解的基础上，增加重升温的过程，使得算法能够以较大的概率跳出局部最优解，从而继续搜索潜在的最优解。

重升温过程的具体实现方法可以根据问题的特点和求解精度的要求而有所不同。一种比较常见的方法是在降温的每个阶段结束后，增加温度一段时间，然后再继续下一个降温阶段。

需要注意的是，重升温过程将增加算法的时间复杂性，此时也将增加算法搜索最优解的时间。因此，在实际的应用中为了权衡算法的探索能力和收敛速度，重升温过程增加的温度应该根据当前解的质量进行调整，如果当前解的质量已经非常高，那么应该适当减小增温幅度甚至舍弃重升温过程，进而保证算法的收敛速度，否则适当提高增温幅度，激活各状态的接受概率，避免算法在局部最优解处停滞不前，进而保证算法搜索解的质量。

5.4.5 多普勒型降温策略

指数降温方式降温缓慢，搜索最优解效率低，单次降温在温度较低时容易陷入局部最优解状态；快速降温方式则过早地降到了低温状态，使得后续迭代过程基本不会再更

新优化结果；而多普勒型曲线则能够综合二者的优点并消除它们的缺陷，不但趋于低温的速度不紧不慢，而且在退火的过程中还进行不断的重升温过程，使得算法在优化过程中具有多次跳出局部最优解的机会，从而更容易搜索到全局最优解。图 5.5 展示了三种降温方式下，温度随迭代次数增加而变化的曲线。

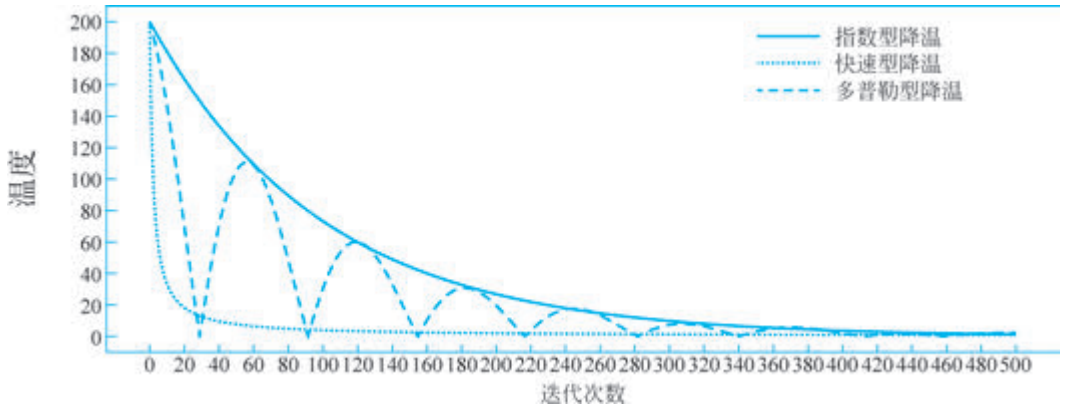

图 5.5　降温曲线对比

由于多普勒型降温是多次降温的过程，难免会丢失已经搜索到的最优解，因此可以将多普勒型降温策略和保留算子策略相结合，最终优化效果会更佳。

多普勒型降温曲线的表达式如下所示，其中 T 为当前温度；T_0 为初始温度；δ 为降温系数；t 为当前迭代次数；$T_{\max}$ 为总迭代次数。

$$T = T_0 \delta^t \left[\cos\left(0.05 \cdot T_{\max}\left(1 - \frac{t}{T_{\max}}\right)\right)\right] + \cos\left(1 - \frac{t}{T_{\max}}\right)$$

5.5　模拟退火算法的时间复杂度

如果算法从初始温度降到终止温度需要 N 次，问题解的维度为 D，马尔可夫链的长度为 L，每次生成候选解的个数为 M；那么算法的时间复杂度为 $N \times L \times O(M \times D)$。

计算方式为：生成初始解的时间复杂度为 $O(1)$；单次生成候选解的时间复杂度为 $O(M \times D)$；计算每个解对应适应度值的时间复杂度为 $O(M \times D)$；某个温度下算法进行迭代的时间复杂度为 $L \times [O(M \times D) + O(M \times D)]$，则完整迭代过程的时间复杂度为

$$O(1) + N \times L \times [O(M \times D) + O(M \times D)] \approx N \times L \times O(M \times D)$$

5.6 实例应用

5.6.1 求解函数最值

使用模拟退火算法求解一组 x 与 y，使得函数 $f(x,y)=x^2+y^2-|1.5xy|$ 在 $x\in[-10,10]$，$y\in[-10,10]$ 条件下取值最小。

算法的搜索空间(即函数 $f(x,y)$ 的三维图像)如图 5.6 所示。

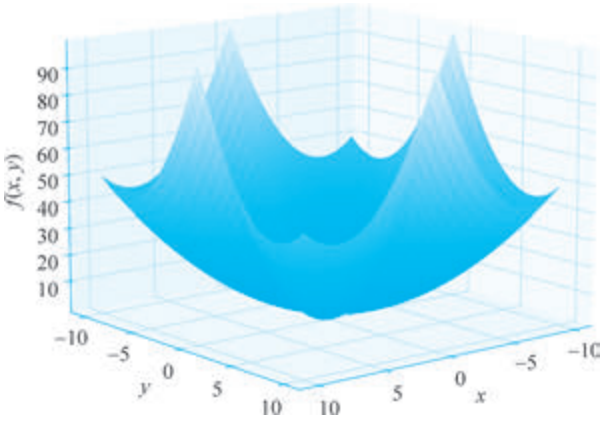

图 5.6 函数 $f(x,y)$ 的三维图像

使用二次退火的方式求解函数的最小值，首次退火过程产生新解的方式为随机生成，目的是充分探索解空间。引入保留算子存储首次退火过程产生的最优解，再次退火过程在最优解的邻域内不断产生新解，目的是充分进行局部开发，如果有更优解则更新，否则保持不变，即不再接受较劣解。

相关参数设置：初始温度为 300，终止温度为 1，退火率为 0.98，马尔可夫链长度为 20，再次退火过程中生成邻域解的最大步长为 0.5。

算法首次退火的迭代结果如图 5.7 所示，收敛曲线显示适应度值在迭代初期和中期上下波动较大，在迭代后期逐渐趋于稳定，这是因为温度逐渐下降，算法跳出局部最优解的概率也在逐渐下降。为了更清晰地展示算法在迭代后期的表现，图 5.8 提供了首次退火下局部迭代的效果图。首次退火过程获得的最优解为 $f(x,y)=0.013\,813\,4$(保留 7 位小数)，如果没有引入保留算子，首次退火的结果是 $f(x,y)=0.097\,791\,7$。

算法再次退火的迭代结果和局部迭代的效果分别如图 5.9 和图 5.10 所示，因为引入了保留算子策略，所以收敛曲线不再波动。经过首次退火过程，算法已经找到了较优

解，因此再次退火过程中适应度值只有小范围的下降，最终获得的最优解为 $f(x,y)=0.000\ 002\ 2$，此时 $x=0.00171$，$y=-0.000\ 31$，这个结果非常接近理论最优解 $f(x,y)=0$，表明模拟退火算法在求解函数最值问题上具有较强的寻优能力。

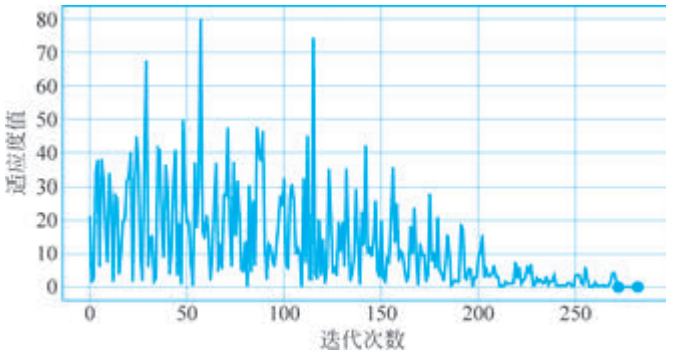

图 5.7　首次退火的完整迭代结果

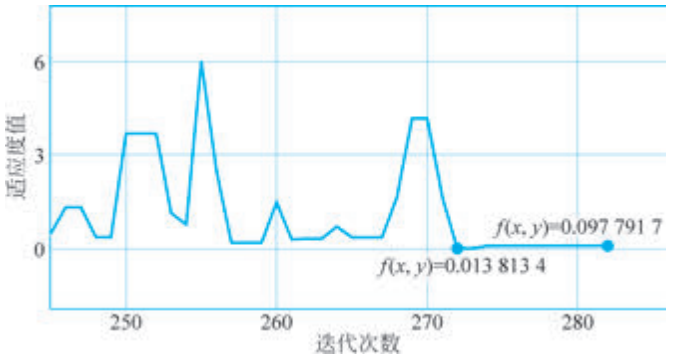

图 5.8　首次退火的局部迭代结果

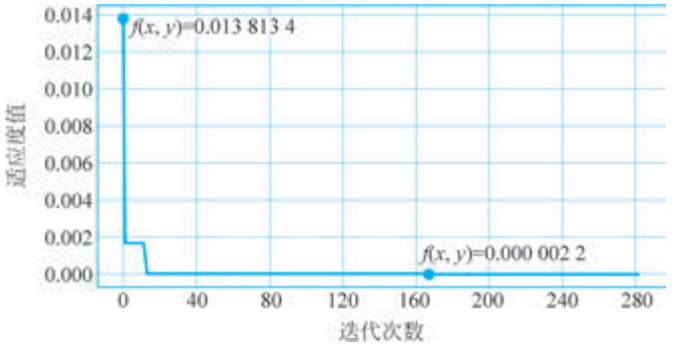

图 5.9　再次退火的完整迭代结果

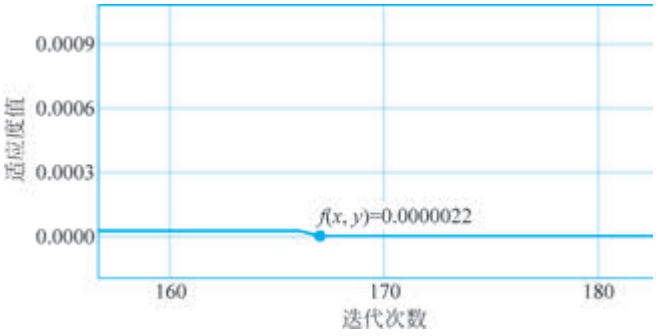

图 5.10　再次退火的局部迭代结果

5.6.2　拉压弹簧设计

拉压弹簧如图 5.11 所示，设计拉压弹簧的目的是在满足最小挠度、振动频率和剪应力这三者的约束下，使得拉压弹簧的重量最小。该问题涉及三个依次排列的关键决策变量：弹簧线圈的直径 d、弹簧簧圈的直径 D 及绕线圈数 P。

图 5.11　拉压弹簧示意图

如果我们用 x_1、x_2 和 x_3 分别表示弹簧线圈直径、弹簧簧圈直径和绕线圈数，那么目标函数可表示为

$$\min f(x)=(x_3+2)x_2x_1^2$$

约束条件为

$$g_1(x)=1-\frac{x_2^3x_3}{71\,785x_1^4}\leqslant 0$$

$$g_2(x)=\frac{4x_2^2-x_1x_2}{12\,566(x_2x_1^3-x_1^4)}+\frac{1}{5108x_1^2}-1\leqslant 0$$

$$g_3(x)=1-\frac{140.45x_1}{x_2^2x_3}\leqslant 0$$

$$g_4(x)=\frac{x_1+x_2}{1.5}-1\leqslant 0$$

$$0.05 \leqslant x_1 \leqslant 2;\ 0.25 \leqslant x_2 \leqslant 1.3;\ 2 \leqslant x_3 \leqslant 15$$

使用二次退火的方式求解该问题，相关参数设置为：初始温度为 300，终止温度为 1，退火率为 0.99，马尔可夫链长度为 20，再次退火过程中生成邻域解的最大步长为 0.5。

模拟退火算法首次退火的迭代结果如图 5.12 所示，收敛曲线显示适应度值在迭代过程中一直处于波动状态，这是因为该优化问题的解所对应的适应度值本身较小，在优化过程中产生的差值 ΔE 很小，由 Metropolis 准则可以得出，即使温度下降到终止温度时，算法仍有较大的概率跳出当前解。若要使曲线在迭代后期趋于收敛状态，可以通过调低终止温度实现，因为本书考虑了再次退火的过程，所以即使首次退火过程未达到收敛状态也无关紧要。首次退火获得的最优解为 $f(x)=0.028\ 56$（保留 5 位小数），如果没有引入保留算子，首次退火的结果是 $f(x)=0.105\ 44$。

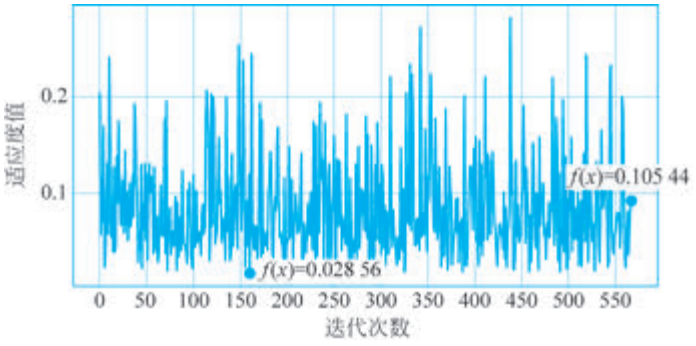

图 5.12　首次退火的迭代结果

算法再次退火的迭代结果如图 5.13 所示，因为引入了保留算子策略，所以收敛曲线不再波动。算法在首次退火已经获得较优解的情况下再次搜寻更优解，最终获得的最优解为 $f(x)=0.013\ 00$，此时弹簧线圈直径 $d=0.053\ 84$，弹簧簧圈直径 $D=0.409\ 36$，弹簧簧圈 $P=8.952\ 10$，这是一组满足约束条件的较优化参数，对拉压弹簧的设计具有指导意义。

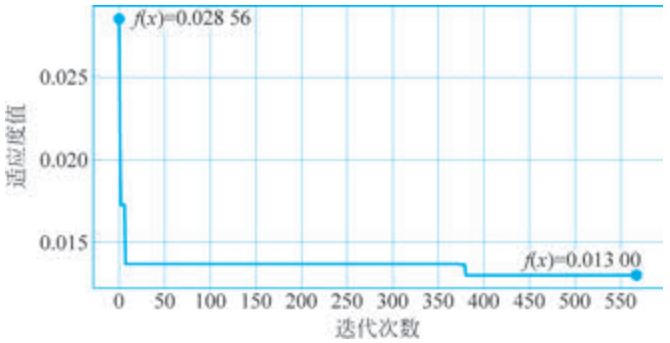

图 5.13　再次退火的迭代结果

5.6.3 压力容器设计

设计压力容器时的目标是最小化其制造成本，包括配对、成型和焊接等环节。压力容器如图 5.14 所示，它的两端都有封盖封住，头部一端的封盖为半球状。在压力容器的设计问题中，我们定义 4 个主要的优化参数：L 为圆柱形主体(不计头部)的截面长度，R 为圆柱体的内侧半径，T_s 表示圆柱体壁的厚度，T_h 为头部的厚度。

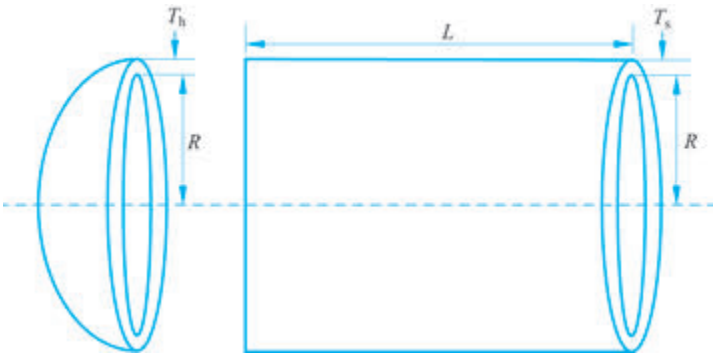

图 5.14 压力容器示意图

该问题的目标函数可表示为

$$x=[x_1,x_2,x_3,x_4]=[T_s,T_h,R,L]$$

$$\min f(x)=0.6224x_1x_3x_4+1.7781x_2x_3^2+3.1661x_1^2x_4+19.84x_1^2x_3$$

约束条件为

$$g_1(x)=-x_1+0.0193x_3\leqslant 0$$

$$g_2(x)=-x_2+0.009\,54x_3\leqslant 0$$

$$g_3(x)=-\pi x_3^2-\frac{4\pi x_3^3}{3}+1\,296\,000\leqslant 0$$

$$g_4(x)=x_4-240\leqslant 0$$

$$0\leqslant x_1\leqslant 100;\ 0\leqslant x_2\leqslant 100;\ 10\leqslant x_3\leqslant 100;\ 10\leqslant x_4\leqslant 100$$

使用二次退火的方式求解该问题，相关参数设置为：初始温度为 300，终止温度为 1，退火率为 0.99，马尔可夫链长度为 20，再次退火过程中生成邻域解的最大步长为 0.5。

模拟退火算法首次退火的迭代结果如图 5.15 所示，由收敛曲线可以看出，适应度值在迭代初期下降很快，在之后的迭代过程中只有小幅度的下降，在第 480 代左右达到收敛状态。首次退火过程获得的最优解为 $f(x)=23\,531.420\,05$(保留 5 位小数)，如果没

有引入保留算子，首次退火的结果是 $f(x)=23\,531.420\,05$。

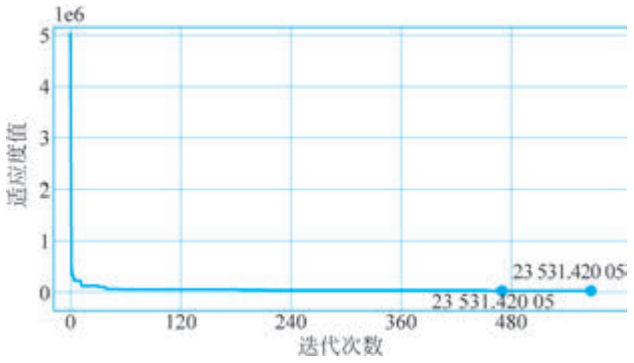

图 5.15　首次退火的迭代结果

算法再次退火的迭代结果如图 5.16 所示，适应度值随着迭代次数的增加而逐渐减小，在 300 代左右达到了收敛状态。最终获得的最优解为 $f(x)=8628.796\,95$，此时圆柱体壁厚 $T_s=1.343\,05$、头部壁厚 $T_h=0.672\,92$、内壁半径 $R=69.401\,00$、截面长度 $L=5.996\,70$，这是一组满足约束条件的较优化参数，对压力容器的设计具有指导意义。

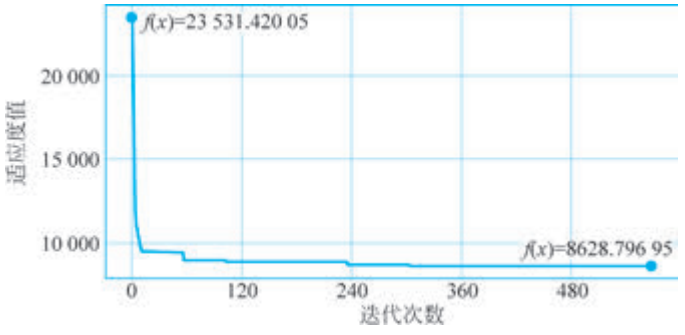

图 5.16　再次退火的迭代结果

从首次退火的适应度曲线可以看出，整个迭代过程并没有接受较劣解，这是因为在这个优化问题的求解过程中，当前解和较劣解的适应度差值 ΔE 太大，而初始高温又不够大，由 Metropolis 准则可知此时接受较劣解的概率极低。设置初始温度为 300 000，保留其他参数不变，此时迭代结果如图 5.17 所示。将初始温度调高后，算法将增加接受较劣解的概率，适应度迭代曲线出现明显的上下波动，经过再次退火过程所得的优化结果也比初始温度为 300 时的优化结果要更优，但是此时算法也将占用更多的计算资源。

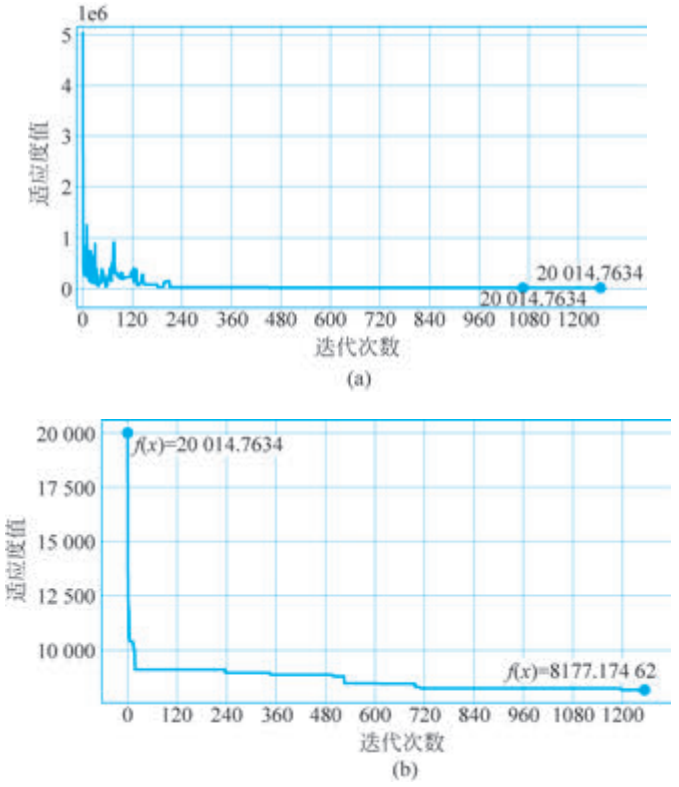

图 5.17 极高温下退火的迭代结果

5.7 习题

(1) 什么是模拟退火算法？它的基本原理是什么？

(2) 模拟退火算法的关键参数有哪些？

(3) 什么是退火率？它有什么作用？

(4) 什么是 Metropolis 准则？它在算法中有什么作用？

(5) 模拟退火算法适不适合求解组合优化问题？

(6) 模拟退火算法的马尔可夫链长度是多少？

(7) 模拟退火算法有哪些特点？

(8) 保留算子有什么作用？

(9) 改进模拟退火算法的重升温和降温策略。

(10) 使用模拟退火算法求一组 x 与 y，使函数 $f(x,y)=|x^2+y^3|-(x+y)^2-y$ 在条件 $x\in[-10,10]$，$y\in[-10,10]$ 下取得最小值。