Advances in Applied Mathematics 应用数学进展, 2023, 12(5), 2128-2137
Published Online May 2023 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2023.125216

Hans汉斯

求解约束优化问题的改进樽海鞘算法

李端洋*，史兰艳

北京建筑大学理学院，北京

收稿日期：2023年4月9日；录用日期：2023年5月3日；发布日期：2023年5月12日

摘　要

针对约束优化问题，提出了一种运用改进樽海鞘算法求解约束优化问题的方法。通过外点法将约束优化问题转化为一系列无约束优化问题，然后运用双领导者结合败者淘汰策略的樽海鞘算法(DLSSA)，并对所得界约束优化问题进行求解以获得约束问题的解。利用6个约束优化基准测试问题对所得算法进行数值实验，实验结果表明，对于所有问题算法，PF-DLSSA所求解均优于算法PF-SSA所求解，即实验所得的最优值及其他数据都优于对比算法PF-SSA所求，所得算法能够有效地求解约束优化问题，且比对比算法效果要好。

关键词

约束优化问题，外点罚函数法，樽海鞘算法，双领导者策略，败者淘汰策略，数值实验

An Improved Salp Swarm Algorithm for Constrained Optimization Problems

Duanyang Li*, Lanyan Shi

School of Science, Beijing University of Civil Engineering and Architecture, Beijing

Received: Apr. 9th, 2023; accepted: May 3rd, 2023; published: May 12th, 2023

Abstract

Aiming at the constrained optimization problem, an improved salp swarm algorithm was proposed to solve the constrained optimization problem. The constrained optimization problem is transformed into a series of unconstrained optimization problems by the outer point method, and then the salp swarm algorithm (DLSSA), which combines the two-leader and loser elimination strategy, is used to solve the unconstrained optimization problem to obtain the solution of the

*通讯作者。

文章引用: 李端洋, 史兰艳. 求解约束优化问题的改进樽海鞘算法[J]. 应用数学进展, 2023, 12(5): 2128-2137.
DOI: 10.12677/aam.2023.125216

constrained problem. The numerical experiments of the proposed algorithm are carried out by using six constraint optimization benchmark problems. Experimental results show that, for all problem algorithms, the solution of algorithm PF-DLSSA is superior to that of algorithm PF-SSA, that is, the optimal value and other data obtained by the experiment are superior to that obtained by the comparison algorithm PF-SSA. The proposed algorithm can effectively solve constrained optimization problems, and the effect is better than that of the comparison algorithm.

Keywords

Constrained Optimization Problem, Outer Point Penalty Function Method, Salp Swarm Algorithm, Two-Leader Strategy, Loser's Elimination Strategy, Numerical Experiment

1. 引言

生活及工业生产等其他领域的问题通常伴随着一定的约束条件[1]。因此，如何使用改进所得算法去求解约束优化问题十分值得研究。用群智能优化算法去求解有约束优化问题的方法有很多，例如外点罚函数法[2]，内点罚函数法[3]，增广拉格朗日法[4]，多目标法[5]等。本章采用外点罚函数法结合算法DLSSA去求解约束优化问题。

对于工业生产及其他领域的优化问题，往往伴随着约束条件，而不是纯粹的某一方面达到最优[5]，对应到优化问题中，即所求解问题的数学模型中须加入一定的约束条件，这些约束条件可描述为一系列的等式约束条件及不等式约束条件，其具体的数学模型如下：

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & h_i(x)=0, \quad i=1,2,\cdots,s \\ & g_j(x)\le 0, \quad j=s+1,\cdots,p \end{aligned} \tag{1}$$

文献[6]针对樽海鞘群算法求解精度不高和收敛速度慢等缺点，提出一种正弦余弦算法的樽海鞘群算法(SCSSA)。引入 Logistics 混沌序列生成初始种群，增加初始个体的多样性；将正弦余弦算法作为局部因子嵌入到樽海鞘群算法中，对樽海鞘个体进行正弦和余弦优化；对最优樽海鞘的领域空间进行差分演化变异策略，增强局部搜索能力。将改进算法在 8 个典型复杂函数优化问题上进行仿真实验，并同正弦余弦算法和樽海鞘群算法进行对比。实验结果表明，该算法具有更好全局搜索能力和局部搜索能力，寻优精度比标准算法有所增强。文献[7]针对樽海鞘群算法求解精度不高和收敛速度慢等缺点，提出一种多子群的共生非均匀高斯变异樽海鞘群算法，根据不同适应度值将樽海鞘链群分为三个子种群，各个子种群分别进行领导者位置更新、追随者共生策略和链尾者非均匀高斯变异等操作。使用统计分析、收敛速度分析、Wilcoxon 检验、经典基准函数和 CEC2014 函数的标准差来评估改进樽海鞘群算法的效率。结果表明，改进算法具有更好的寻优精度和收敛速度。尤其在求解高维和多峰测试函数上，改进算法拥有更好性能。文献[8]提出了一种具有交叉方案和 Levy 飞行的樽海鞘算法。即采用 Levy 飞行策略分别改进樽海鞘算法中领导者和追随者的位置更新方式。数值实验已经对各种测试函数进行了测试，测试函数包括单模态、多模态、复合函数。数值实验的结果表明，所提出的算法 SSACL 在精度、稳定性和性能方面均优于其他先进算法，此外，Wilcoxon 的秩和检验说明了所提出的算法的优势明显。为了提升樽海鞘算法

的求解精度和全局搜索能力。

2. 罚函数法

罚函数法的目的将约束优化问题转化为一个新约束优化问题，用无约束优化方法来求解[9]。其主要方法是将约束条件转换为惩罚项，从而重新构造目标函数，通过调整惩罚因子，得到一系列的无约束优化问题，使其趋向最优解[10]。其主要包括外点罚函数法，内点罚函数法等。

2.1. 外点罚函数法

外点罚函数法可用于求解同时存在等式与不等式约束的约束优化问题，又称为障碍发法，即将惩罚项视为超出可行域的障碍。外点罚函数法的罚函数的可定义为：

$$G(x)=\sum_{i=1}^{s}\left(h_i(x)\right)^2+\sum_{j=s+1}^{p}\left\{\max\left\{g_j(x),0\right\}\right\}^2 \tag{2}$$

等式右端第一项为等式约束条件所对应的惩罚项，第二项为不等式约束所对应的惩罚项[11]。

2.2. 内点罚函数法

与外点罚函数法不同，内点罚函数法的惩罚因子随着迭代的进行不断减小，且内点罚函数法仅适用于只存在不等式条件的约束优化问题，这类问题的数学模型如下：

$$\begin{aligned}&\min \quad f(x)\\&\text{s.t.} \quad g_j(x)\le 0,\ j=1,\cdots,q\end{aligned} \tag{3}$$

内点罚函数法罚函数的定义为

$$R(x)=\sum_{j=1}^{q}\frac{1}{g_j(x)} \tag{4}$$

该罚函数仅由不等式约束对应的惩罚项构成[12]。

3. 改进的樽海鞘算法

在求解界约束子问题时运用群体智能优化算法，本文采用双领导者结合败者淘汰策略的樽海鞘算法(Double leader combined with loser elimination strategy of salp swarm algorithm, DLSSA)，算法 DLSSA 的具体过程如下所示。

3.1. 双领导者策略

为避免樽海鞘算法中的个体后期陷入局部最优，因此引入双领导者策略，即增加一个领导者，追随者随机选择其中一个领导者追随进行位置更新。

3.1.1. 领导者位置更新

在基本的樽海鞘算法中仅存在一个领导者，在迭代后期“链”的结构逐渐趋向稳定，导致个体出现聚集现象从而陷入局部最优。本章增加了一个新的领导者，即种群中的前两个个体均被视为领导者，在搜索前期，即当前迭代数 t 未达到最大迭代数 T 的一半时，两个领导者均按照原樽海鞘算法的领导者位置更新公式进行位置更新，其更新公式如下：

$$x_1^j(t+1)=\begin{cases}F^j+c_1\left(\left(ub^j-lb^j\right)c_2+lb^j\right) & c_3\ge 0.5\\ F^j-c_1\left(\left(ub^j-lb^j\right)c_2+lb^j\right) & c_3<0.5\end{cases} \tag{5}$$

其中 $i=1$，2。在搜索后期，即当前迭代数 t 超过最大迭代数 T 的一半 $T/2$ 后，第一个领导仍按(4-1)式进行位置更新，而令第二个领导者远离食物的位置，其更新机制类似于象群算法中女首领的位置更新[13]，即：

$$x_2(t+1)=0.01F \tag{6}$$

其中 $t>T/2$。

3.1.2. 追随者位置更新

种群中除前两个个体外，其余个体均为追随者，对种群中的追随者，可选择不同的领导者进行跟随。为增强种群多样性，追随者随机加入不同的樽海鞘链，为使跟随两个不同领导者的概率基本相同，在每一个追随者 $x_l(t)(l=3,4,\cdots,N)$ 位置更新前，生成一个(0,1)中的随机数 p_l，，当 p_l 大于 0.5 时，该个体加入第一组，追随第一个领导者，反之该个体加入第二组，追随第二个领导者，从而所有追随者随机分成两组，记为 $x_{1,2}(t),x_{1,3}(t),\cdots,x_{1,N_1}(t)$ 与 $x_{2,2}(t),x_{2,3}(t),\cdots,x_{2,N_2}(t)$，其中 $N_1+N_2=N$。若令 $x_{1,1}(t)=x_1(t)$，$x_{2,1}(t)=x_2(t)$ 则追随者的位置更新公式可描述如下：

$$x_{g,k}(t+1)=\left(x_{g,k}(t)+x_{g,k-1}(t)\right)/2 \tag{7}$$

其中 $g=1$，2，表示该个体在第 g 组，而 k 则表示该组中的第 k 个追随者，且有 $k\geq 2$。

3.2. 败者淘汰策略

在自然界的种群中，不同的生物个体进化程度不同。优胜劣汰，自然选择的现象十分普遍。在种群中许多实力较弱的个体，这些个体的狩猎等其他竞争能力较低，因此会被种群淘汰。使用新生个体对现有较弱个体进行替换，有利于保持种群多样性。因此本章引入败者淘汰策略，以改进原樽海鞘算法：在所有追随者完成位置更新后，计算每个个体的适应度值，种群中适应度值最差的 10%的个体，即所对应目标函数值大的 10%的个体，将被新生成的个体所替换，替换公式如下：

$$x_f(t+1)=lb+(ub-lb)rand \tag{8}$$

其中 x_f 表示种群中适应度值大的 10%的个体。

在进行上述过程后，对所有个体进行适应度值的计算，其中适应度值最小的个体与食物的适应度值进行比较，两者中适应度值更小的个体为新的食物，并令迭代次数加 1，进行再一次的领导者与追随者的位置更新，重复上述过程，直至达到终止条件即达到最大迭代次数 T，然后停止迭代。

3.3. 双领导者结合败者淘汰策略改进的樽海鞘算法的步骤

双领导者结合败者淘汰策略改进的樽海鞘算法具体步骤如下：

步骤 1：初始化种群，设置最大迭代次数 T，并令当前迭代次数 $t=0$；

步骤 2：计算个体适应度值，选出食物的位置；

步骤 3：领导者 1 按照公式(5)进行更新，迭代前期，即 t 小于 $T/2$ 时领导者 2 按照(5)式更新，迭代后期，即 t 不小于 $T/2$ 时领导者 2 按照式(6)进行位置更新；

步骤 4：对每个追随者，先生成一个随机数以选择其追随的领导者，然后根据公式(7)进行位置更新；

步骤 5：计算所有个体的适应度值，并比较当前最优个体与食物的适应度值：适应度值更小的则为新的食物的位置；

步骤 6：对种群中适应度值大的 10%的个体根据公式(8)进行位置更新；

步骤 7：判断是否满足终止条件：

若满足，则停止迭代输出全局最优解；

否则，令 $t = t + 1$，并回到步骤 3。

3.4. 双领导者结合败者淘汰策略改进的樽海鞘算法的伪代码

双领导者结合败者淘汰策略改进的樽海鞘算法伪代码如下：

1. 初始化种群，并设定最大迭代次数 T，Let $t = 0$；

2. 计算个体的适应度值，并确定食物的位置；

3. while $t < T$

3.1 领导者 1 根据公式(5)进行位置更新；

If $t < T/2$

领导者 2 按照公式(5)进行位置更新；

Else

领导者 2 按照公式(6)进行位置更新；

End

3.2 对每个追随者，先生成一个随机数以选择其追随的领导者，然后根据公式(7)进行位置更新；

3.3 计算所有个体的适应度值，并比较当前最优个体与食物的适应度值：适应度值更小的则为新的食物的位置；

3.4 对种群中适应度值大的 10%的个体根据公式(8)进行位置更新；

3.5 Let $t = t + 1$.

4. End

5.输出食物及其适应度值。

4. 基于外点罚函数改进的双领导者结合败者淘汰策略樽海鞘算法

对于改进得到的双领导者结合败者淘汰策略的樽海鞘算法DLSSA，在无约束问题的求解方面已经通过数值实验表明了其求解能力优异，但其无法求解约束优化问题，本节结合外点罚函数法和算法DLSSA得到的算法PF-DLSSA，以用于解决约束问题。

对于约束优化问题(1)可转换如下：

$$F(x) = f(x) + \lambda \sum_{i=1}^{s} \left(h_i(x)\right)^2 + \lambda \sum_{j=s+1}^{p} \left\{\max\left\{g_j(x), 0\right\}\right\}^2 \tag{9}$$

其中 $f(x)$为约束问题的目标函数，λ 为惩罚因子，其表达式为 $\lambda = 10^k$，($k = 0,1,\cdots,20$)，k 为算法 PF-DLSSA 的外层迭代次数，即随着外层迭代的进行，惩罚因子逐渐增大，从而得到一系列的无约束优化问题。

本文利用外点罚函数法，将式(5-1)所示的优化问题转化为式(5-7)为目标函数的无约束优化问题，即将约束问题转化为一系列的无约束优化问题，再使用算法 DLSSA 求解转换后的无约束优化问题，从而得到算法 PF-DLSSA。算法 PF-DLSSA 的求解过程主要为两个部分，即外层迭代及内层迭代。其中首先进行外层迭代，通过改变惩罚因子 λ，将约束优化问题转换为无约束优化问题；而后进行内层迭代，根据算法 DLSSA 的流程和位置更新机制进行领导者和追随者的位置更新，以及较差的个体淘汰及替换；内层迭代的终止条件为达到最大内层迭代次数 T，然后判断其是否满足外层迭代的终止条件 $G(x) \le \varepsilon$ 或 $k \ge 20$：若满足，则输出食物的位置及其适应度值；否则更新惩罚因子，进行下一次的迭代。

4.1. 算法 PF-DLSSA 的步骤

算法PF-DLSSA的步骤如下：

步骤 1：初始化惩罚因子 $\lambda=1$ 及外层迭代精度 ε，并设置外层迭代次数 $k=0$；

步骤2：根据约束优化问题中的目标函数、等式约束与不等式约束构造式(9)目标函数；

步骤3：种群初始化，设置种群中个体数量N，内层迭代最大次数T，并令当前内层迭代次数为t=0；

步骤4：按照式(9)计算每一个个体的适应度值，得到食物的位置及确定两个领导者的位置；

步骤5：领导者1按照公式(5)进行更新，迭代前期，即t小于T/2时领导者2按照(5)式更新，迭代后期，即t不小于T/2时按照式(6)进行位置更新；

步骤6：对每个追随者，先生成一个随机数以选择其追随的领导者，然后根据公式(7)进行位置更新；

步骤7：按照式(9)计算所有个体的适应度值，并比较当前最优个体与食物的适应度值：适应度值更小的则为新的食物的位置；

步骤8：对种群中适应度值大的10%的个体根据公式(8)进行位置更新，并令t=t+1；

步骤9：判断是否满足内层终止条件$t=T$：

若满足，则终止内层迭代；

否则，回到步骤4；

步骤10：判断是否满足外层迭代终止条件：

若满足则输出食物的位置及其适应度值；

否则，令外层迭代次数$k=k+1$，更新惩罚因子并回到步骤2。

4.2. 算法 PF-DLSSA 的伪代码

算法PF-DLSSA的伪代码如下：

1. 设置惩罚项及外层迭代精度，外层迭代次数k=0；
2. 根据约束优化问题中的目标函数、等式约束与不等式约束构造式(9)目标函数；
3. 初始化种群，设置种群规模N，最大迭代次数T，当前内层迭代次数t=0；
4. 计算个体的适应度值，并确定食物的位置及两个领导者；
5. while $G(x)\le\varepsilon$
6. while $t<T$

6.1 领导者1根据公式(5)进行位置更新；

If $t<T/2$

领导者2按照公式(5)进行位置更新；

Else

领导者2按照公式(6)进行位置更新；

End

6.2对每个追随者，先生成一个随机数以选择其追随的领导者，然后根据公式(7)进行位置更新；

6.3计算所有个体的适应度值，并比较当前最优个体与食物的适应度值：适应度值更小的则为新的食物的位置；

6.4对种群中适应度值大的10%的个体根据公式(8)进行位置更新；

6.5 Let $t=t+1$.

7. End
8. Let $k=k+1$
9. End
10. 输出食物的位置及其适应度值。

5. 数值实验

为检验算法PF-DLSSA求解约束优化问题的能力，本章选取了较为常用的6个约束优化问题进行了数值实验，6个约束优化问题的目标函数及约束条件及理论最优值在附录部分。

对比算法为算法 PF-DLSSA 与外点罚函数法结合原樽海鞘算法得到的算法 PF-SSA。算法 PF-SSA 分为外层迭代与内层迭代，首先进行外层迭代，通过改变惩罚因子 λ，将约束优化问题转换为无约束优化问题；而后进行内层迭代，根据算法 SSA 的流程和位置更新机制进行领导者和追随者分别按照式(2-3)与式(2-5)进行位置更新；内层迭代的终止条件为达到最大内层迭代次数 T，然后判断其是否满足外层迭代的终止条件或：若满足，则输出食物的位置及其适应度值；否则更新惩罚因子，进行下一次的迭代。

数值实验中，算法PF-DLSSA与算法PF-SSA种群规模均设置为100，内层最大迭代次数T为500，外层迭代次数最大值设为20，惩罚因子为10^k。为避免实验存在偶然性，对每一个问题进行30次的重复求解，30次求解所得到目标函数值的最好值，最差值，平均值及标准差见表1。

Table 1. Comparison results of objective function values for constrained optimization problems
表1. 约束优化问题目标函数值对比结果

问题	算法	最优值	最差值	平均值	标准差
c_1	PF-DLSSA	0.75	0.907	0.7901	0.0622
	PF-SSA	0.8019	1.0006	0.8331	0.668
c_2	PF-DLSSA	−0.09583	−0.01901	−0.0941	0.0184
	PF-SSA	−0.0264	0.0401	−0.0133	0.0497
c_3	PF-DLSSA	−5.508	−5.1513	−5.3997	0.0184
	PF-SSA	30 次独立求解均不能在 20 次外层迭代中得到可行解			
c_4	PF-DLSSA	7.1049	7.2019	7.1109	8.3519
	PF-SSA	30 次独立求解均不能在 20 次外层迭代中得到可行解			
c_5	PF-DLSSA	−6.787e+3	2.235e+3	−8.23	3.06e+3
	PF-SSA	−1.203e+3	2.6636e+4	2.8798e+3	6.948e+3
c_6	PF-DLSSA	−14.9	−7.5226	−10.13	1.7985
	PF-SSA	−8.19	−5.8355	−7.033	1.231

见表1，对于所有问题，算法PF-DLSSA所求得的目标函数值均优于算法PF-SSA所求。对于问题c_1，算法PF-DLSSA在30次独立求解所得目标函数值的最好值为理论最优值，且标准差明显小于算法PF-SSA所求目标函数值的标准差。对于问题c_2，两算法均未在30次独立求解中求得理论最优值−1，但算法PF-DLSSA在30次独立求解中得到的目标函数最好值及平均值均小于−0.09，明显小于算法PF-SSA所求得的目标函数值。对于问题c_3与问题c_4，算法PF-SSA在30次独立求解中均不能在20次外层迭代中得到满足约束条件的解，而算法PF-DLSSA可以求得可行解，且对于问题c_3，算法PF-DLSSA在30次独立求解中得到的目标函数值的标准差较小。对于问题c_5，两个算法所得结果均较差，但算法PF-DLSSA所求得的目标函数值优于算法PF-SSA所求得的目标函数值。对于问题c_6，算法PF-DLSSA所求目标函数值的最好值接近理论最优值−15，而算法PF-SSA相较理论最优值相差很远。综上，算法PF-DLSSA求解无约束优化问题的性能好于算法PF-SSA。

6. 结论

本章介绍了约束优化问题的基本概念及其数学模型，同时介绍了一些求解约束优化问题的方法，即罚函数法、多目标函数法等。将改进所得算法DLSSA与外点罚函数法结合得到算法PF-DLSSA以用于约束优化问题的求解。为验证算法PF-DLSSA求解约束优化问题的性能，选取了6个约束优化问题进行数值实验，数值实验结果表明，数值实验的结果表明算法PF-DLSSA求解约束优化问题的性能好于算法PF-SSA。本文创新点如下：

1) 对樽海鞘算法进行改进得到求解无约束优化问题能力更优的算法DLSSA。

2) 提出一种基于改进的樽海鞘算法结合外点罚函数法的算法PF-DLSSA，以用以解决约束优化问题。

3) 选取标准测试问题进行数值实验，实验结果表明算法 PF-DLSSA 的求解约束优化问题的能力强于算法 PF-SSA。

参考文献

[1] 贺向阳, 徐玲. 一个求解有约束的全局优化问题的变换函数法[J]. 科学技术与工程, 2008, 8(24): 6445-6447+6462.

[2] 刘爱兵. 基于外点罚函数法门式起重机箱型主梁的优化设计[J]. 机械工程师, 2014(11): 211-213.

[3] 李金龙, 江征风, 万鹏. 内点罚函数法在链传动优化设计中的应用[J]. 现代机械, 2003(2): 21-80.

[4] 叶峰, 邵之江, 梁昔明, 钱积新. Lagrange 乘子初始值和罚因子迭代方式的研究[J]. 厦门大学学报(自然科学版), 2001(S1): 34-38.

[5] 刘文斌. 有约束的多目标优化问题[J]. 经济数学, 1986(3): 66-70.

[6] 王吉权, 程志文, 张攀利, 代伟婷. 求解有约束优化问题的实数遗传算法改进研究[J]. 控制与决策, 2019, 34(5): 937-946.

[7] 陈忠云, 张达敏, 辛梓芸. 正弦余弦算法的樽海鞘群算法[J]. 计算机应用与软件, 2020, 37(9): 209-214.

[8] 陈忠云, 张达敏, 辛梓芸. 多子群的共生非均匀高斯变异樽海鞘群算法[J]. 自动化学报, 2022, 48(5): 1307-1317.

[9] He, M. and Lang, C. (2021) Salp Swarm Algorithm with Crossover Scheme and Lévy Flight for Global Optimization. *Journal of Intelligent & Fuzzy Systems*, **40**, 1-12.

[10] 周淑娟. 基于罚函数法的汽车动力传动系统参数优化匹配研究[J]. 汽车维修, 2023(1): 27-30.

[11] 王林军, 王锬, 杜义贤, 徐柳, 黄文超, 刘晋玮. 一种基于外罚函数法的结构可靠性分析方法[J]. 三峡大学学报(自然科学版), 2019, 41(1): 92-96.

[12] 李根. 基于内点罚函数法及 Visual Basic 螺旋弹簧优化设计[J]. 汽车工艺师, 2020(12): 37-39+43.

[13] Li, W., Wang, G. and Alavi, A.H. (2020) Learning-Based Elephant Herding Optimization Algorithm for Solving Numerical Optimization Problems. *Knowledge-Based Systems*, **195**, Article ID: 105675. https://doi.org/10.1016/j.knosys.2020.105675

附　录

6 个约束问题如下：

C_1：

$$\begin{aligned}&\min f(x)=x_1^2+(x_2-1)^2\\&\text{s.t. } h_1(x)=x_2-x_1^2=0,\\&\quad -1\le x_1\le 1,-1\le x_2\end{aligned}$$

全局最优值为 0.75。

C_2

$$\begin{aligned}&\min f(x)=-\frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1+x_2)}\\&\text{s.t. } g_1(x)=x_1^2-x_2+1\le 0\\&\quad g_2(x)=-x_1+x_2-3\le 0\\&\quad 0\le x_1\le 10,\ 0\le x_2\le 10,\end{aligned}$$

全局最优值为−0.095825。

C_3：

$$\begin{aligned}&\min f(x)=-x_1-x_2\\&\text{s.t. } g_1(x)=-2x_1^4+8x_1^3-8x_1^2+x_2-2\le 0,\\&\quad g_2(x)=-4x_1^4+32x_1^3-88x_1^2+96x_1+x_2-36\le 0,\\&\quad 0\le x_1\le 3,0\le x_2\le 4\end{aligned}$$

全局最优值为−5.508013。

C_4：

$$\begin{aligned}&\min f(x)=x_1+x_2+x_3\\&\text{s.t. } g_1(x)=-1+0.0025(x_4+x_6)\le 92,\\&\quad g_2(x)=1+0.0025(x_5+x_7-x_4)\le 0,\\&\quad g_3(x)=-1+0.01(x_8-x_5)\le 0,\\&\quad g_4(x)=-x_1x_6+833.33252x_4+100x_1-83333.333\le 0,\\&g_5(x)=-x_2x_7+1250x_5+x_2x_4-1250x_4\le 0,\\&g_6(x)=-x_3x_8+1250000(x_2-2)^2+x_3x_5-2500x_5\le 0,\\&100\le x_1\le 10000,\ 1000\le x_i\le 10000(i=2,3),\ 10\le x_i\le 1000(i=4,5,6,7,8)\end{aligned}$$

全局最优值为 7049.3307。

C_5：

$$\begin{aligned}&\min f(x)=(x_1-10)^3+(x_2-20)^3\\&\text{s.t. } g_1(x)=-(x_1-5)^2-(x_2-5)^2+100\le 0,\\&\quad g_2(x)=-(x_1-6)^2-(x_2-5)^2-82.81\le 0\\&\quad 13\le x_1\le 100,\ 0\le x_2\le 100\end{aligned}$$

全局最优值为−6961.81388。

C_6：

$$\min f(x)=5\sum_{i=1}^{4}x_i-5\sum_{i=1}^{4}x_i^2-\sum_{i=5}^{13}x_i$$

$$\text{s.t.}\quad g_1(x)=2x_1+2x_2+x_{10}+x_{11}-10\le 0$$

$$g_2(x)=2x_1+2x_3+x_{10}+x_{12}-10\le 0$$

$$g_3(x)=2x_2+2x_3+x_{11}+x_{12}-10\le 0$$

$$g_4(x)=-8x_1+x_{10}\le 0$$

$$g_5(x)=-8x_2+x_{11}\le 0$$

$$g_6(x)=-8x_3+x_{12}\le 0$$

$$g_7(x)=-2x_4-x_5+x_{10}\le 0$$

$$g_8(x)=-2x_6-x_7+x_{11}\le 0$$

$$g_9(x)=-2x_8-x_9+x_{12}\le 0$$

$$0\le x_i\le 1\ (i=1,\cdots,9)\quad 0\le x_i\le 10\ (i=10,11,12)\quad 0\le x_{13}\le 1$$

全局最优值为−15。