基于邻域回归的昂贵问题优化

摘要

昂贵优化问题是科学与工程领域中常见的难题，其核心挑战在于目标函数的高昂评估成本。为有效解决这一问题，本文引入了一种融合进化计算与代理模型的先进策略。通过基于邻域回归的局部搜索策略复现了 NRO 算法，然而 NRO 算法忽略了全局搜索策略，因此本文通过将进化计算的全局搜索能力与代理模型的高效建模相结合，引入 RBF 全局代理模型，成功实现了改进的 NRO 算法，即 RNRO 算法。实验结果表明，这一创新性方法在五个测试函数中取得了显著的效果，因此 RNRO 算法可以有效地实现对 NRO 算法的精确优化，显著降低了计算成本，极大地提高了 NRO 算法的收敛速度和鲁棒性，为科学与工程领域中昂贵优化问题的解决提供了可行、高效的解决途径。

关键词：进化计算，昂贵优化，回归模型，代理模型

1 引言

在实际工程和科学应用中，存在一类被称为“昂贵优化问题”的挑战。昂贵优化问题是指在求解优化问题时，评估目标函数值的成本非常高昂的一类问题。这种昂贵成本可以体现在计算时间、实验费用、资源消耗等方面。在许多实际应用中，例如工程设计、空气动力学优化、材料科学等领域，评估一个候选解的目标函数值可能需要进行昂贵的实验、数值模拟或现场测试。

解决昂贵优化问题实际上是求解昂贵优化问题的最优解，也就是在决策空间不断迭代搜索解的过程，为此引入进化算法。进化算法利用了自然界“物竞天择，适者生存”的思想，是一类模拟自然界进化过程的优化算法，其中包括遗传算法、粒子种群优化、鲸鱼算法等。这类算法通过维护一个种群，并通过模拟进化的过程来逐步改进解的质量。进化计算的优点在于其适应性强，能够在复杂、高维的搜索空间中寻找全局最优解。

为了更加高效地解决昂贵优化问题，引入了代理模型。代理模型是对昂贵目标函数的一个近似模型，它能够在计算成本较低的情况下估计目标函数值。代理模型可以是基于统计学习的回归模型、机器学习模型，常用的代理模型包括多项式回归模型、径向基函数模型、克里金模型。

因此本文的研究问题是借助代理模型和进化算法在有限的计算资源下来寻找昂贵优化问题的最优解并基于邻域回归的昂贵问题优化 [1] 进行了复现与改进，从而减少实际目标函数的评估次数，提高优化的效率。

2 相关工作

为了更好地了解和分析现实世界中各式各样的复杂系统，仿真模型得到了广泛的应用，例如交通工具、土木结构和医疗设备 [2]。各种各样的工程任务也使用了仿真模以实现决策空间的探索、敏感度分析和性能评估等。然而，这些仿真模型往往计算成本非常昂贵 [3]，例如计算流体力学和有限元分析。随着计算机性能的逐步提高，数据量的不断增长，目前的仿真模型越来越复杂，计算成本越来越昂贵，为此引入进化算法和代理模型来优化这类昂贵问题。

进化算法借鉴了达尔文的物种起源论，通过模拟自然界自然选择的进化过程而衍生的一类算法，各种各样的进化算法已经用于寻找现实世界中复杂问题的最优解，例如遗传算法 [4]、差分进化算法 [5]、粒子种群优化算法 [6]、灰狼算法 [7] 等。

进化算法不断迭代的过程中会消耗大量的函数评估次数，而昂贵优化问题的函数评估次数往往是极其有限的，因此传统的进化算法难以应用到昂贵优化问题中。为此引入代理模型，代理模型是根据有限的样本点拟合的关于目标函数的近似模型，常见的代理模型包括多项式回归模型 [8]、克里金模型 [9]、径向基函数模型 [10]、神经网络模型 [11]、支持向量机模型 [12] 等。

通过将代理模型和进化算法相结合，一方面可以极大地减少进化算法的搜索时间和计算成本，另一方面可以提高算法的全局搜索能力和自适应性。由于基于代理模型的进化算法可以在有限的计算资源下寻找到全局的最优解，因此已经成为了解决昂贵优化问题的主要方法。通过有限的样本点，可以训练出单个代理模型 [13]，也可以训练出多个代理模型，然后通过线性组合的方式集成一个新的代理模型 [14]。实际上，不存在一个模型在所有问题上都表现得很好，因此将多个代理模型集成一个代理模型的方式逐渐成为了一种趋势。例如， [14] 将多项式回归模型、径向基函数模型和克里金模型集成一个代理模型，对于每个模型的权重采用了交叉验证的方式进行计算，同时采用粒子种群优化算法作为进化算法。实验表明，该算法在低维问题优化上具有很大的优势。除了代理模型的选择之外，进化算法迭代结束时候选解的筛选也十分重要。例如，如果进化算法使用克里金模型，那么可以采用改进概率策略 (PoI) [15]、期望改进策略 (ExI) [16] 和最小置信下限策略 (LCB) [17] 对候选解进行筛选，从而提高解的质量。

基于代理模型的进化算法一般只能解决低维的优化问题，为了解决高维的优化问题，可以对问题的维度做降维处理，例如， [18] 提出了一种利用随机投影矩阵将原问题空间投影到低维的多个子空间，然后将多个子空间的代理模型集成为高维空间的代理模型。 [19] 通过将映射技术引入克里金模型，从而减少了解空间的维度同时有利于代理模型的构建。

3 本文方法

3.1 本文方法概述

对于给定的输入包括种群规模 N、邻居规模 T、步长 F，NRO 算法最终会输出昂贵优化问题的最优解，即最小值点，算法的总体框架如算法 1 所示。第一步，初始化种群和适应度评估；第二步，确定个体的邻居；第三步，预测下降方向；第四步，生成新的子代；第五步，输出昂贵优化问题的最优解。

Algorithm 1 Framework of NRO

Input: population size **N**, neighborhood size **T**, step size **F**

Output: the best individual $\mathbf{x^{best}}$

Initialize the population $\mathbf{P} = \{\mathbf{x^1}, \mathbf{x^2}, \cdots, \mathbf{x^N}\}$
Evaluate the objective functions of all the individuals in the population **P**
while termination criterion is not fulfilled **do**
  for i = 1 to **N do**
    Determine the neighborhood of individual $\mathbf{x^i}$ (denoted as $\mathbf{B}\left(\mathbf{x^i}\right)$) by recording its **T** nearest neighbors according to Euclidean distance
    d=predictDescDir($\mathbf{x^i}$,$\mathbf{B}\left(\mathbf{x^i}\right)$)
    Determine the best individual in the population **P** (denoted as $\mathbf{x^{worst}}$)
    v=generateOffspring($\mathbf{x^{best}}$,**d**,**F**)
    Evaluate the objective function of **v**
    Determine the worst individual in the population **P** (denoted as $\mathbf{x^{worst}}$)
    if $\mathbf{f}(\mathbf{x^{worst}}) \geq \mathbf{f}(\mathbf{v})$ **then**
      $\mathbf{x^{worst}} = \mathbf{v}$
    end if
  end for
end while
Determine the best individual in the population P (denoted as $\mathbf{x^{best}}$)
return $\mathbf{x^{best}}$

3.2 初始化种群和适应度评估

首先初始化种群 P，也就是随机生成昂贵优化问题的 N 个初始解，然后利用真实目标函数评估种群 P 中所有解的函数值并作为适应度值，由于该算法的输出是找到昂贵优化问题的最小值，因此适应度值越小越好。

3.3 确定个体的邻居

初始化完成后，随即进入迭代过程，当且仅当停止条件满足时，迭代结束并输出最优解，昂贵优化问题的停止条件一般为预先设定的函数评估次数耗尽或者优化算法收敛。当进入主循环后，按顺序依次遍历种群中的所有个体，然后根据欧氏距离计算当前个体 x^i 与种群中所有个体的距离，确定欧氏距离最小的 T 个个体，记为当前个体 x^i 的邻居 $B(x^i)$。

3.4 预测下降方向

对于个体 x^i 及其邻居 $B(x^i)$，首先可以确定邻居 $B(x^i)$ 的决策空间，即

$$S(x^i) = \{z | z = (z_1, \cdots, z_D)^T \in R^D, z_j \in [l_j^i, u_j^i]\} \tag{1}$$

其中 $l_j^i = \min\{x_j^{i_1}, \cdots, x_j^{i_T}\}, u_j^i = \max\{x_j^{i_1}, \cdots, x_j^{i_T}\}$，也就是决策空间的上下界，然后利用 $B(x^i)$ 中的所有点拟合出一个线性回归模型。假定所求的线性回归模型的形式为

$$y = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + \cdots + \beta_D x_D = \boldsymbol{\beta}^T \boldsymbol{x} \tag{2}$$

其中 $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2, \cdots, \beta_D)^T, (1, x_1, x_2, \cdots, x_D)^T$，则对 $B(x^i)$ 中的所有点均有

$$A\boldsymbol{\beta} = \boldsymbol{f} \tag{3}$$

其中 $A = \begin{bmatrix} x_1^{i_1} & x_2^{i_1} & \cdots & x_D^{i_1} \\ x_1^{i_2} & x_2^{i_2} & \cdots & x_D^{i_2} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{i_T} & x_2^{i_T} & \cdots & x_D^{i_T} \end{bmatrix}, \boldsymbol{f} = \begin{bmatrix} f(x^{i_1}) \\ f(x^{i_2}) \\ \vdots \\ f(x^{i_T}) \end{bmatrix}$，则利用最小二乘法最小化代价函数

$$\|A\boldsymbol{\beta} - \boldsymbol{f}\|_2^2 = (A\boldsymbol{\beta} - \boldsymbol{f})^T (A\boldsymbol{\beta} - \boldsymbol{f}) \tag{4}$$

可以得到

$$\boldsymbol{\beta} = (A^T A)^{-1} A^T \boldsymbol{f} \tag{5}$$

因为线性回归模型是单调递增或者单调递减的，因此线性回归模型的最优解 x^* 总是在决策空间的顶点处取到，即

$$x_j^* = \begin{cases} l_j^i, & \beta_j > 0 \\ u_j^i, & \beta_j \le 0 \end{cases}, j = 1, 2, \ldots, D \tag{6}$$

而决策空间 $S(x^i)$ 的中心点为

$$\boldsymbol{c} = \left[\frac{1}{2}(l_1^i + u_1^i), \frac{1}{2}(l_2^i + u_2^i), \cdots, \frac{1}{2}(l_D^i + u_D^i)\right]^T \tag{7}$$

所以可构造下降方向

$$d = x^* - c = (d_1, d_2, \cdots, d_D)^T \tag{8}$$

其中

$$d_j = \begin{cases} -\frac{1}{2}(u_j^i - l_j^i), & \beta_j > 0 \\ \frac{1}{2}(u_j^i - l_j^i), & \beta_j \le 0 \end{cases}, j = 1, 2, \cdots, D \tag{9}$$

因此对于给定的个体 x^i 及其邻居 $B(x^i)$，总能利用线性回归模型构造出其对应的下降方向，而下降方向也决定了新的个体的生成，算法伪代码如算法 2 所示。

Algorithm 2 predictDescDir

Input: $\mathbf{x^i}$ and its neighborhood $\mathbf{B}\left(\mathbf{x^i}\right)$ size $\mathbf{T}$, step size $\mathbf{F}$

Output: descent direction $\mathbf{d}$

1: Compute the hyperrectangle space of $\mathbf{B}\left(\mathbf{x^i}\right)$ (denoted as $\mathbf{S}\left(\mathbf{x^i}\right)$)

2: Find the linear regression relation $\mathbf{y} = \beta_\mathbf{0} + \boldsymbol{\beta}^\mathbf{T}\mathbf{x}$ for the observed values of $\mathbf{B}\left(\mathbf{x^i}\right)$

3: Determine the descent direction according to Equation (6)

4: **return** $\mathbf{d}$

3.5 生成新的子代

算法伪代码如算法 3 所示。一旦下降方向确定，那么子代也随之确定。假设当前种群 P 中的全局最优解为 x^{best}，步长为 F，下降方向为 d，那么利用

$$v = x^{best} + F \cdot d \tag{10}$$

即可构造新的个体 v，同时新的个体可能超出原问题决策空间的上下界，因此需要做越界处理，越界包括小于下界和大于上界，若新的个体 v 在第 j 个维度上越界，那么

$$v_j = \begin{cases} (lb_j + x_j^{best})/2, & v_j < lb_j \\ (ub_j + x_j^{best})/2, & v_j > ub_j \end{cases} \tag{11}$$

其中 lb, ub 分别为原问题决策空间的上下界，然后利用真实函数评估这个新的个体，如果这个新的个体的适应度值优于种群 P 中最差的个体的适应度值，即拥有最大适应度值的个体，那么这个新的个体将替换掉这个最差的个体，从而保持种群规模恒定，避免种群规模不断扩大而造成的计算成本增加。

Algorithm 3 generateOffspring

Input: parent $\mathbf{x}^{\mathbf{best}}$, descent direction $\mathbf{d}$, step size $\mathbf{F}$

Output: offspring $\mathbf{v}$

1: $\mathbf{v} = \mathbf{x}^{\mathbf{best}} + \mathbf{F} \cdot \mathbf{d}$

2: Check the boundaries of v by using Equation (11)

3: **return** $\mathbf{v}$

3.6 输出昂贵优化问题的最优解

通过不断迭代直到函数评估次数用尽或者算法收敛，算法收敛是指在若干次迭代中找到的解与种群中最好的解的欧氏距离小于一个指定的阈值。迭代结束后，算法将输出种群 P 中适应度值最小的解，即昂贵优化问题的最优解。

4 复现细节

4.1 与已有开源代码对比

本文基于邻域回归的昂贵问题优化 [1] 进行了复现与改进，文中的 NRO 算法是开源的，其源代码如图 1 所示。由于本文的 NRO 算法存在一些问题，因此在源代码的基础上进行了改进，改进的 NRO 算法称为 RNRO。NRO 算法的执行时间很快，对于某些测试函数也可以达到很好的效果，但还是存在其局限性，NRO 仅仅在每一个个体的邻居区域进行搜索，忽略了全局搜索，容易陷入局部最小值，在单模态问题上表现优异，但在多模态问题上有待改进，因此引入了全局搜索模型作为辅助，从而提高算法的收敛性和鲁棒性，RNRO 算法的总体框架如算法 4 所示。

```
function [bestFitness,record,dsp] = NRO(fobj,dim,lb,ub,maxFEs,N,F,NS)
X = (ub-lb)*rand(N,dim)+lb;
Fitness = arrayfun(@(j)fobj(X(j,:)),1:N)';
FEs = N;
% find out the best solution
bestFitness = min(Fitness);
record = [FEs,bestFitness];
dsp = {
    'NRO with global best strategy';
    'fixed step';
};
while maxFEs > FEs
    for j = 1 : N
        [~,sortedIdx] = sort(pdist2(X(j,:),X));
        NeighborIdx = sortedIdx(1:NS);
        NeighborX = X(NeighborIdx,:);
        NeighborFitness = Fitness(NeighborIdx);
        [~,baseIdx] = min(Fitness);
        baseX = X(baseIdx,:);
        increase = max(NeighborX) - min(NeighborX);
        NeighborX = [NeighborX ones(NS,1)];
        coef = NeighborX'*NeighborX\NeighborX'*NeighborFitness;
        coef = coef(1:dim)';
        increase(coef>0) = -increase(coef>0);
        U = baseX + increase * F;
        U(U>ub) = (ub + baseX(U>ub))/2;
        U(U<lb) = (lb + baseX(U<lb))/2;
        UFitness = fobj(U);
        FEs = FEs + 1;
        % replacement
        [gWorstFitness,gWorstIdx] = max(Fitness);
        if UFitness <= gWorstFitness % global replacement
            X(gWorstIdx,:) = U;
            Fitness(gWorstIdx) = UFitness;
        end
        % record
        bestFitness = min(bestFitness,UFitness);
    end
    record = [record;[FEs,bestFitness]];
end
end
```

图 1. NRO 算法开源代码

RBF 作为一种常用的代理模型，有着许多优点，能利用有限的样本点尽可能地近似真实目标函数，且计算成本相对较低，因此引入 RBF 代理模型作为全局搜索模型，其函数表达式为

$$f(x) = \sum_{i=1}^{N} \omega_i \varphi(\|x - x^{(i)}\|) \tag{12}$$

其中 $\varphi(r)$ 为核函数，包括高斯函数 $\varphi(r) = e^{-\frac{r^2}{2c^2}}$、复合二次型函数 $\varphi(r) = \sqrt{r^2 + c^2}$、立方函数 $\varphi(r) = r^3$ 等，本节采用高斯核函数作为 RBF 核。

对于给定的 N 个样本点 $x^{(1)}, x^{(2)}, \cdots, x^{(N)}$ 及其真实函数值 $y^{(1)}, y^{(2)}, \cdots, y^{(N)}$，有

$$A\boldsymbol{\omega} = \boldsymbol{y} \tag{13}$$

其中 $A = \begin{bmatrix} \varphi(\|x^{(1)} - x^{(1)}\|) & \varphi(\|x^{(1)} - x^{(2)}\|) & \cdots & \varphi(\|x^{(1)} - x^{(N)}\|) \\ \varphi(\|x^{(2)} - x^{(1)}\|) & \varphi(\|x^{(2)} - x^{(2)}\|) & \cdots & \varphi(\|x^{(2)} - x^{(N)}\|) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi(\|x^{(N)} - x^{(1)}\|) & \varphi(\|x^{(N)} - x^{(2)}\|) & \cdots & \varphi(\|x^{(N)} - x^{(N)}\|) \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$，则通过

Algorithm 4 Framework of RNRO

Input: population size $\mathbf{N}$, neighborhood size $\mathbf{T}$, step size $\mathbf{F}$

Output: the best individual $\mathbf{x^{best}}$

$\mathbf{v} = \mathbf{x^{best}} + \mathbf{F} \cdot \mathbf{d}$
Initialize the population $\mathbf{P} = \{\mathbf{x^1}, \mathbf{x^2}, \cdots, \mathbf{x^N}\}$
Evaluate the objective functions of all the individuals in the population $\mathbf{P}$
while termination criterion is not fulfilled **do**
    Build a global surrogate RBF model $\mathbf{f_s}(\mathbf{x})$ with $\mathbf{P}$
    Obtain $\mathbf{x^{best}}$ by using JADE to solve $\mathbf{x^{best}} = \arg\min \mathbf{f_s}(\mathbf{x})$ and replace the worst individual in the population $\mathbf{P}$ with $\mathbf{x^{best}}$
    NRO($\mathbf{N}, \mathbf{T}, \mathbf{F}$)
end while
Determine the best individual in the population P (denoted as $\mathbf{x^{best}}$)
return $\mathbf{x^{best}}$

表 1. 五个测试函数的特点

Test function	Dim(d)	Search space $\mathbf{\Omega}$	Optimal value	Characteristic
Ellipsoid	10,20,30	$[-20, 20]^d$	0	Unimodal
Rosenbrock	10,20,30	$[-20, 20]^d$	0	Multimodal
Ackley	10,20,30	$[-20, 20]^d$	0	Multimodal
Griewank	10,20,30	$[-20, 20]^d$	0	Multimodal
Rastrigin	10,20,30	$[-20, 20]^d$	0	Multimodal

最小二乘法最小化代价函数

$$\|A\boldsymbol{\omega} - \boldsymbol{y}\|_2^2 = (A\boldsymbol{\omega} - \boldsymbol{y})^T(A\boldsymbol{\omega} - \boldsymbol{y}) \tag{14}$$

可以得到

$$\boldsymbol{\omega} = (A^T A)^{-1} A^T \boldsymbol{y} \tag{15}$$

在每一次进入 NRO 算法进行局部搜索前，可以利用全局 RBF 代理模型和差分进化算法 JADE 求解出全局的最优解，然后替换掉种群 P 中适应度最大的个体，以此保持种群规模不变。

4.2 实验数据集

为了评估本文复现的 NRO 算法和改进的 RNRO 算法的性能并做一个定量和定性的分析，本文将 NRO 算法和 RNRO 算法在 CEC2015 的五个计算昂贵问题的测试函数上进行了测试，五个计算昂贵问题的测试函数对应的函数名称 (Test function)、维度 (Dim)、搜索空间 (Search space)、真实最优值 (optimal) 和性质 (characteristic) 如表 1 所示。

4.3 实验参数设置

对于本文复现的 NRO 算法，昂贵优化问题的维度或决策变量的个数为 D，取值包括 $10, 20, 30$，种群规模 N 为 $3 \cdot D$，邻居大小 T 为 $D + 3$，步长 F 为 0.4。所有算法都是在 Windows 系统上基于 Matlab 运行的，且实验结果都是通过独立运行了 20 次取平均值计算的。

5 实验结果分析

5.1 四种替换策略的实验结果对比分析

在每一次搜索当前个体的邻居区域后都会生成一个新的个体，然后将这个个体与种群中最差的个体进行比较，如果新个体优于最差的个体，那么将用新个体替换掉最差的个体，为了验证这个替换策略的有效性，分别采取了四种替换策略替换最好 (NRO-RB)、替换最差 (NRO-RW)、随机替换 (NRO-RR)、不替换 (NRO-WR) 进行消融实验，在 CEC2015 的测试函数 Ackley 且维度为 10 的情况下记录了四种替换策略的收敛情况并绘制成收敛曲线，如图 2 所示。

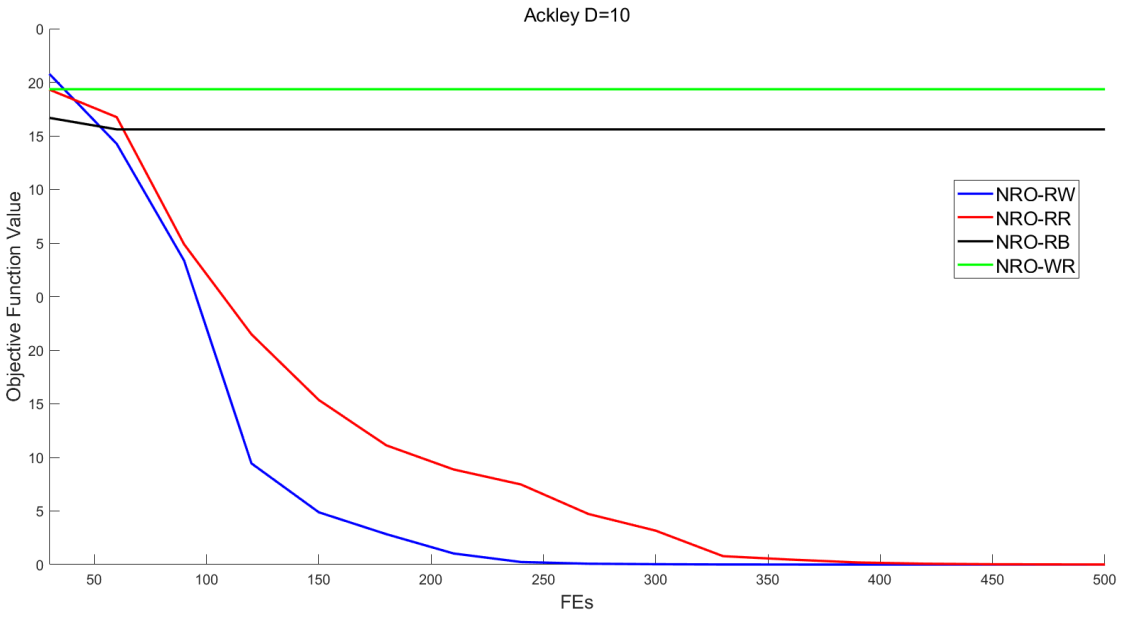

图 2. 四种替换策略的收敛曲线

根据图 2 可以知道，替换最好和不替换的策略性能最差，目标函数无法找到最优值，替换最好的策略会将种群中找到的最优解淘汰掉，这显然不符合进化算法优胜劣汰的思想，而不替换的策略会使得种群的个体保持不变，从而无法通过进化获得更好的个体。替换最差和随机替换的策略性能最好，替换最差的策略可以将种群中适应度最高的个体淘汰掉，从而使得种群不断进化并逐渐分布在全局最优解的附近，这也很好的验证了 NRO 算法中采取这个策略的有效性，而对于随机替换而言，替换的可能是最好，可能是最差，可以是种群中的任意一个个体，可以很好的保持物种多样性，即解空间的全局性，避免陷入局部最优解，因此有些时候要优于替换最差的策略。

5.2 NRO 算法和 RNRO 算法的实验结果对比分析

由于原本的 NRO 算法仅仅在种群中每个个体的邻域内搜索，忽略了全局搜索，因此在多模态问题中难以找到全局的最优解，因此改进的 NRO 算法 (RNRO) 引入了全局搜索策略，也就是通过构建全局 RBF 代理模型和差分进化算法找到一个新的个体并替换掉种群 P 中最差的个体，为了对比 NRO 算法和 RNRO 算法的性能差异，在每一次运行时保持初始种群 P 以及其它参数 D、N、T、F 相同的情况下，独立运行了 20 次，测试了 NRO 算法和 RNRO 算法找到的目标函数的最小值，如表 2 所示。

由表 2 可知，在五个测试函数的 $10, 20, 30$ 三个维度，总共 15 个问题上，RNRO 算法在 15 个问题上均优于 NRO 算法，因为全局搜索策略的引入，RNRO 算法在每一次执行局部邻域搜索前都会先利用种群的所有样本点构建全局 RBF 代理模型，这也就保证了 RNRO 算法可以尽可能地避免陷入局部的最小值，通过不断地全局搜索并添加新的个体到种群中，也可以很好地保证解空间的多样性，因而可以找到目标函数更好的最小值，也可以提高算法的收敛速度和鲁棒性，NRO 算法和 RNRO 算法的收敛曲线如图 3 所示。

横坐标表示函数的评估次数，纵坐标表示当前找到的目标函数的最小值，可以看到，随着函数评估次数的逐渐增加，RNRO 算法找到的目标函数的最小值超过了 NRO 算法，这也验证了全局搜索策略的有效性。

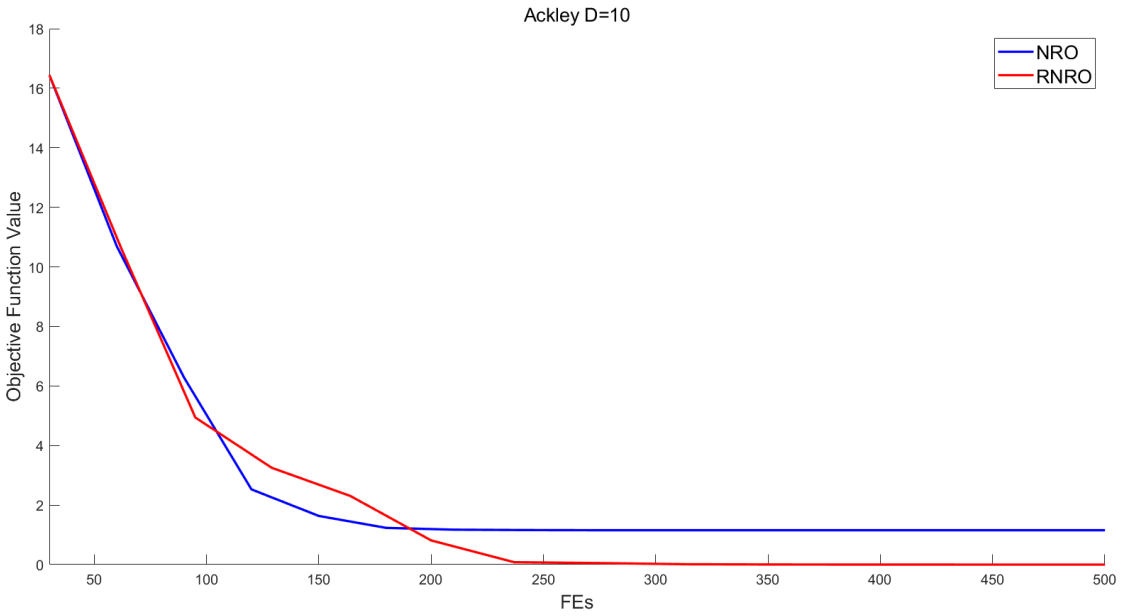

图 3. NRO 算法和 RNRO 算法的收敛曲线

6 总结与展望

基于邻域回归的 NRO 算法以及改进过的 RNRO 算法在 10,20,30 维的五个测试函数上均具有良好的性能，并且 RNRO 算法优于 NRO 算法，这也极大地体现了 RBF 全局代理模型的有效性。然而，无论是 NRO 算法还是 RNRO 算法都存在一些问题，这两种算法在单模态问题上优于多模态问题，在低维问题上优于高维问题。因此，对于多模态问题和高维问题，目

表 2. NRO 算法和 RNRO 算法的对比

Problem	D	NRO	RNRO
	10	1.340E-06	**5.000E-08**
Ellipsoid	10	7.630E-02	**6.000E-04**
	10	2.370E+02	**5.000E-03**
	10	6,378E+01	**2.153E+01**
Rosenbrock	10	4.374E+03	**3.300E+01**
	10	1.702E+02	**7.461E+01**
	10	2.452E+00	**6.267E-01**
Ackley	10	1.967E+00	**5.077E-01**
	10	4.725E+00	**3.300E-03**
	10	3,499E-01	**1.134E-01**
Griewank	10	2.240E-02	**3.700E-03**
	10	1.277E-01	**0.000E+00**
	10	8.626E+01	**4.353E+01**
Rastrigin	10	2.510E+02	**1.482E+02**
	10	4.560E+02	**2.234E+02**

前的方法仍需更深入的研究。对于更大规模、更高维度的问题，算法的适应性和性能还有进一步改进的空间。未来的研究可探索引入更为灵活、自适应的算法结构，以更好地适应多样的问题场景。

此外，代理模型的选择和集成方法是当前研究的关键。虽然已有一些有效策略，但在实际应用中可能会受到问题特性的制约。未来的工作可以侧重于发展更通用、适应性更强的代理模型选择和集成策略，不仅仅局限于单个代理模型，可以将多个代理模型集成为一个代理模型，充分发挥各个代理模型的优势，提高方法的适用性。

在未来的研究中，一方面，可以尝试深入整合深度学习技术，引入深度学习等先进技术，以提升代理模型对复杂问题的建模能力，使其更好地适应真实世界的挑战。另一方面，面对实际应用中更为复杂的多目标问题，未来的研究可以将焦点扩展到多目标昂贵优化，以更全面地解决实际场景中的挑战。通过对这些方向的深入研究，我们有望进一步提高昂贵优化问题的求解效率和质量，为实际应用领域提供更为可靠的解决方案。

参考文献

[1] Yuren Zhou, Xiaoyu He, Zefeng Chen, and Siyu Jiang. A neighborhood regression optimization algorithm for computationally expensive optimization problems. *IEEE Transactions on Cybernetics*, 52(5):3018–3031, 2020.

[2] Qi Zhou, Xinyu Shao, Ping Jiang, Zhongmei Gao, Chaochao Wang, and Leshi Shu. An active learning metamodeling approach by sequentially exploiting difference information from variable-fidelity models. *Advanced Engineering Informatics*, 30(3):283–297, 2016.

[3] Jun Zheng, Xinyu Shao, Liang Gao, Ping Jiang, and Zilong Li. A hybrid variable-fidelity global approximation modelling method combining tuned radial basis function base and kriging correction. *Journal of Engineering Design*, 24(8):604–622, 2013.

[4] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[5] Kenneth Price, Rainer M Storn, and Jouni A Lampinen. *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media, 2006.

[6] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995.

[7] Huachao Dong and Zuomin Dong. Surrogate-assisted grey wolf optimization for high-dimensional, computationally expensive black-box problems. *Swarm and Evolutionary Computation*, 57:100713, 2020.

[8] Zongzhao Zhou, Yew Soon Ong, My Hanh Nguyen, and Dudy Lim. A study on polynomial regression and gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm. In *2005 IEEE congress on evolutionary computation*, volume 3, pages 2832–2839. IEEE, 2005.

[9] Jay D Martin and Timothy W Simpson. Use of kriging models to approximate deterministic computer models. *AIAA journal*, 43(4):853–863, 2005.

[10] Rommel G Regis. Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions. *IEEE Transactions on Evolutionary Computation*, 18(3):326–347, 2013.

[11] John Eason and Selen Cremaschi. Adaptive sequential sampling for surrogate model generation with artificial neural networks. *Computers & Chemical Engineering*, 68:220–232, 2014.

[12] Stella M Clarke, Jan H Griebsch, and Timothy W Simpson. Analysis of support vector regression for approximation of complex engineering analyses. 2005.

[13] Dong Zhao and Deyi Xue. A comparative study of metamodeling methods considering sample quality merits. *Structural and Multidisciplinary Optimization*, 42:923–938, 2010.

[14] Handing Wang, Yaochu Jin, and John Doherty. Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems. *IEEE transactions on cybernetics*, 47(9):2664–2677, 2017.

[15] Michael TM Emmerich, Kyriakos C Giannakoglou, and Boris Naujoks. Single-and multi-objective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, 2006.

[16] Ruwang Jiao, Sanyou Zeng, Changhe Li, Yuhong Jiang, and Yaochu Jin. A complete expected improvement criterion for gaussian process assisted highly constrained expensive optimization. *Information Sciences*, 471:80–96, 2019.

[17] Bo Liu, Qingfu Zhang, and Georges GE Gielen. A gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. *IEEE Transactions on Evolutionary Computation*, 18(2):180–192, 2013.

[18] Daofu Guo, Zhigang Ren, Yongsheng Liang, and An Chen. Scaling up radial basis function for high-dimensional expensive optimization using random projection. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[19] Xiwen Cai, Liang Gao, and Xinyu Li. Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems. *IEEE Transactions on Evolutionary Computation*, 24(2):365–379, 2019.