**Gaussian过程与信号估计**

**基于LSE算法的改进——致力于显著降低查询点数目的LSE\_fast算法**

目录

一、引言 1

二、算法介绍 2

三、收敛性分析 7

四、数值实验 11

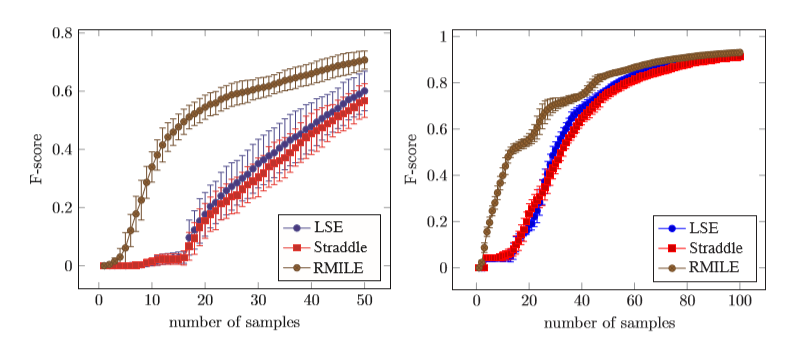
附：参考文献 18

**一、引言**

本文高质量地复现了[1]中所介绍的用于水平集估计的LSE算法，并对其做出了创新性的改进。改进后的算法命名为LSE\_fast，其特性是能够显著降低LSE算法在迭代过程中所需查询点的个数，在小样本空间和高精度要求下，其性能改进尤为明显。相信对于查询函数十分costly的情况，LSE\_fast算法将有重要的应用。

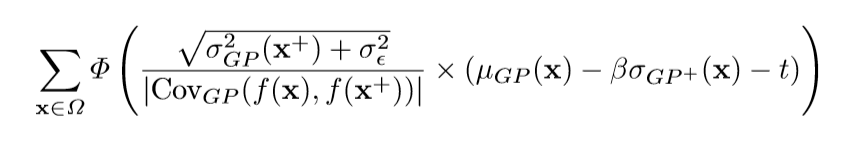
参考文献[1][2][3][4][5]均对水平集估计(Level Set Estimation)问题做出了不同程度的贡献。[1]提出了LSE算法，[3]提出了TRUVAR算法，[5]提出了RMILE算法，上述文献均对自己提出的算法进行了详尽且可靠的收敛性推导。[5]在数值实验部分将自己提出的RMILE算法与[1]提出的LSE算法做了对比，反复阅读[1]和[5]后，我最终决定基于[1]中的LSE算法进行改进，实现新的水平集估计算法，原因有以下两点：

a).[5]在数值实验部分以分类问题常用的F1-score作为评判标准，得出了RMILE算法性能优于LSE算法的结论，但没有提供对于LSE算法性能至关重要的参数（与置信区间长度有关）和（与分类准确度有关），且从[5]的实验结果（如下图）来看，两者性能相差并不大：



因此作为读者而言，这两种算法孰优孰劣，还未成定数。

b).[5]和[1]相比，算法的时间复杂度和空间复杂度更高，这是因为[5]中在计算累积分布函数之前，需要完成如下所示的复杂计算：



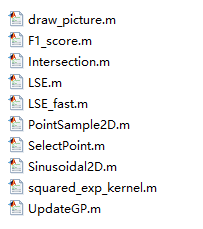
而[1]中对应的部分是对区间求交集。即与[1]相比，[5]需要多进行一些协方差矩阵有关的计算，造成了更高的时间开销，同时也需要在算法迭代过程中存储协方差矩阵，造成了更大的内存开销。

本文在引言之后的正文共有四个部分。其中（二）列出并简要介绍了复现的LSE算法以及改进的LSE\_fast算法的代码文件清单，同时对LSE\_fast算法的原理进行了详细阐述；（三）对LSE\_fast算法的迭代次数T进行了收敛性分析，分别用两种不同的思路得出了T的取值（范围）；（四）做了充分的数值仿真实验，比较了不同条件下两种算法的表现，证明了LSE\_fast算法的改进成效，同时将LSE\_fast算法用于寻找函数全局最优点，说明了其用于优化问题的潜力。

**二、算法介绍**

本部分完成了两件事：a).列出提交代码的文件清单，并简要介绍各个模块的作用；b).详细介绍LSE\_fast算法的原理。（我们考虑的是单一阈值的水平集估计问题，和二分类问题本质相同，因此，下文中“分类”和“估计”意义相同）

（一）文件清单与模块简介



每个模块的作用简介如下：

draw\_picture.m:

用于画图，包括待估计函数的真实等高线图，估计结果的等高线图和F1-score随迭代次数变化图。

F1\_score.m:

用于计算F1-score。

Intersection.m:

用于求两个区间的交集。

LSE.m:

LSE算法的主函数，调用各个子函数实现LSE算法。

LSE\_fast.m:

LSE\_fast算法的主函数，调用各个子函数实现LSE\_fast算法。

PointSample2D.m:

用于在二维区域均匀采样获得待分类的点集合。

SelectPoint.m:

根据最大化ambiguity准则选取下一个查询点。

Sinusoidal2D.m:

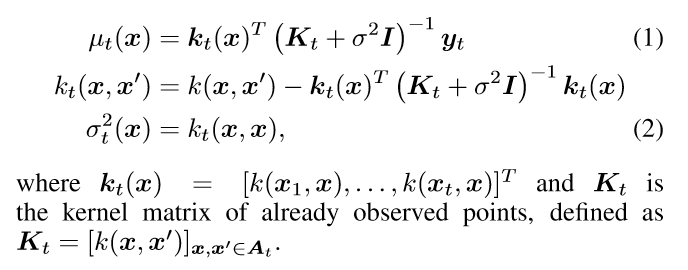
待估计的函数。

squared\_exp\_kernel.m:

高斯核函数，其中和*l*是需要指定的参数。

UpdateGP.m:

根据已观测的点及其观测结果使用以下公式更新拟合的高斯分布：（其中是观测噪声方差）



（二）LSE\_fast算法

LSE\_fast算法的伪代码如下：

The LSE\_fast algorithm

Input: sample set D, GP prior(), threshold h, accurary parameter

Output: predicted sets

1: , for all

2:

3: while do

4:

5: for all do

6:

7: if then

8:

9: else if then

10:

11:

12:

13: if then

14:

15: else then

16:

17: Compute and for all

18:

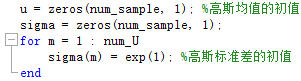
19:

由上述伪代码可知，LSE\_fast算法对于LSE算法的改进体现在第13~16行，即加入了查询结束后立即将查询点归类的机制。由后面的数值实验可以验证，引入此机制后，算法的分类准确度维持在同一水平，而所需查询点的个数显著减少，且在样本点个数越少，分类精度要求越高的情况下，LSE\_fast算法的改进效果越好。

LSE\_fast算法在LSE算法的基础上改进，遵循LSE算法的估计思路，概括如下：

算法的目标是估计一个映射关系f，进而在给定的采样区域D中估计映射后的上水平集H。我们用高斯分布去逼近待估计的映射关系f。算法执行过程中，需要维护上水平集H，下水平集L，以及未分类点集合U。

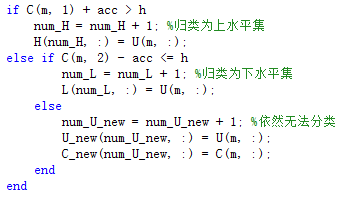
一开始，先假设一个初始的零均值，方差适中的高斯分布（我们设定的初始标准差是e）



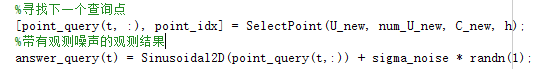
对于每个待分类的点，使用当前高斯分布的均值和方差得到此点的置信区间C(x)



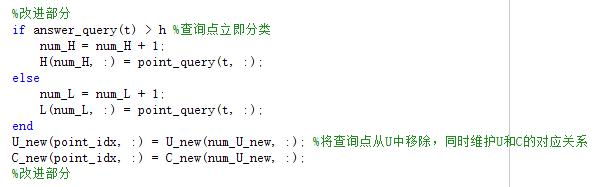
通过比较置信区间C与阈值h的位置关系将当前点U(m)进行分类



如果不是所有点都完成分类，则说明我们当前的高斯分布对于原映射f的逼近程度不够，此时需要通过查询新的点的映射结果来获取更多信息



对于LSE\_fast算法，得到新的查询结果之后立即将此查询点归类



新的待查询点的选择依据是ambiguity最大，具体操作见SelectPoint函数，ambiguity定义为

表征了一种分类的困惑度，一般而言，按照此准则选择出的点在阈值附近。（在数值实验部分可以看到）

高斯分布更新方法见UpdataGP.m



更新后重新计算未分类点的置信区间C，再尝试对这些点进行分类，若还有点未完成分类，则继续寻找新的查询点，更新高斯分布，如此循环往复，直到所有点都完成分类。

**三、收敛性分析**

本部分旨在通过理论推导，以解析的形式得出LSE\_fast算法要达成给定的估计精度，所需要的迭代次数T。下文将根据信息熵和算法本身的迭代特性两个方面给出推导。（由于LSE\_fast算法每次迭代只查询一个点，因此下文所说的“迭代次数”与“查询点个数”意义相同）

（一）通过熵值估计迭代次数T

单一阈值的上水平集估计(Super-Level Set Estimation)，等价于一个二分类问题。我们先考虑熵值最大的二分类问题，即每个样本点属于两个子类的概率相等，均为，且样本点的分类结果之间相互独立，记样本空间为D，样本点总数为，此时分类结果有种，每种结果的概率都是，这个分类问题的熵值可以精确计算，如下所示：

分类映射f就是我们的LSE算法需要估计的函数，我们将f假设为服从高斯分布，LSE算法实际上做的事情，就是通过查询（本文中与“观测”同义）某个点对应的f取值（尽管含有噪声），来获得f的全局信息，每查询一个点，获得的信息就多一些，当我们获得的信息足够多时，便可以较好地掌握f的行为，从而在新的点来临时，预测f的取值，实现水平集估计。

容易发现，如果我们获得了H\_classify大小的信息量，相当于对原分类映射有了完全的掌握，这时分类应是完全准确的。这里我们研究想要获得H\_classify大小的信息量时，需要查询多少的点个数。根据信息论中的知识，假设有随机变量X和Y，其互信息定义为

可以理解为观测Y有助于了解X的信息的多少的度量。那么，设观测点集合为A，观测A中的点所能带来的f的信息量为

根据高斯分布熵的计算公式

可得

其中是观测噪声的标准差，，的计算公式见“算法介绍”部分。

此时我们便可以得到一些有用的结论，在只观测一个点时，的值是确定的，即

而因为已观测点和未观测点的f取值并不独立，所以这也是观测其他点时得到信息量的下界，那么我们便得到此均匀二分类问题所需的观测点数目的上界，即

显然，这个上界估计是十分保守的，因为在观测噪声不是很大的情况下，我们决没有必要（或者说没有足够的预算）为了分类d个点而观测超过d个点，这便是LSE\_fast算法的出发点，因此，对于LSE\_fast算法

这个上界的下移意味着由于观测噪声的存在，我们总是无法确保能完全分类正确。

现在我们考虑观测点集合能得到带来的信息量上界，令

直观地，随着观测点集合A中的点数逐渐增多，会越来越大，从而一定会有，此时A中点的个数就是完成此均匀二分类问题所需观测点数目的下界。即

至此，我们通过信息熵推导得到了完成均匀二分类问题所需的观测点数目范围

对于非均匀二分类问题（更一般的水平集估计问题），其熵值必然小于H\_classify，此时所需观测点数目减小，上界下界都会相应地下移。

（二）通过算法的特性估计迭代次数T

为衡量分类的准确度，定义missclassification loss，简称m- loss，用表示，计算形式为

此处推导同样需要用到（一）中的，含义相同，为便于区别，改写为。

在LSE及LSE\_fast算法中，参数控制置信区间C的长度，C的长度又与(ambiguity)有关。在分类过程中，我们不断对置信区间C取交集，C越来越小，因而也越来越小，随迭代次数t减小的速度为，容易看出，当每个点的长度均小于分类精度时，区间绝不会是任何点的置信区间C的真子集，这意味着每个点都可以成功分类，迭代结束。那么根据长度随着t的变化速度，可以推算得出迭代次数T是满足下面不等式的最小整数。

其中，，是人为指定的用于控制估计效果的参数，此时估计的结果满足

**四、数值实验**

做数值实验有两个目的a).对比LSE算法与LSE\_fast算法的性能，验证LSE\_fast算法的改进效果；b).利用LSE\_fast算法寻找函数全局最优点，指出此算法在优化问题上的应用价值。

如无特殊说明，本部分的实验参数设置如下：

样本点个数；噪声标准差；

高斯分布初值；阈值；

分类精度；置信区间长度控制；

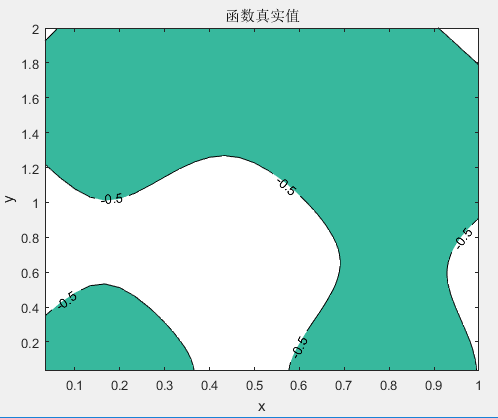
高斯核的参数，；

待估计的函数为

（一）对比LSE算法与LSE\_fast算法

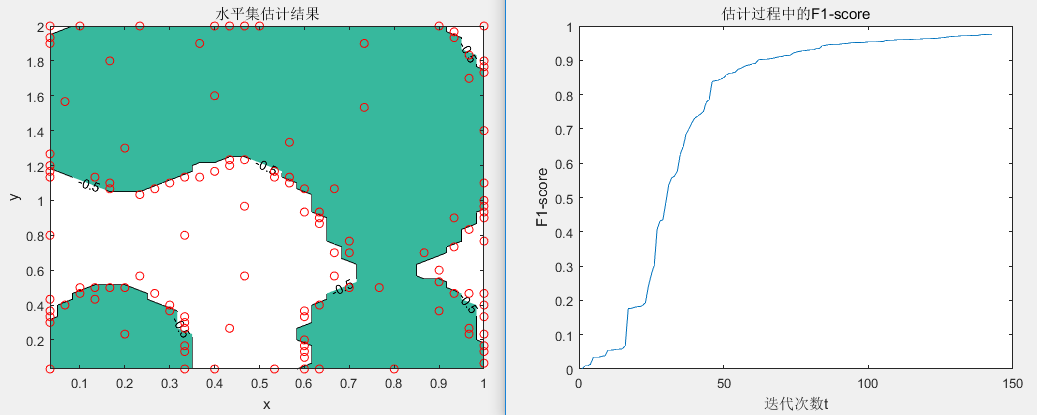
LSE\_fast改进的目的是在不损失分类精度的前提下，显著地降低LSE算法所需查询点的数目。下面控制分类精度与待分类的点数目来比较两种算法的性能。

下面三个实验使用的是同一个待估计的函数，函数真实值的阈值等高线为如下图，后续实验不再单独列出：



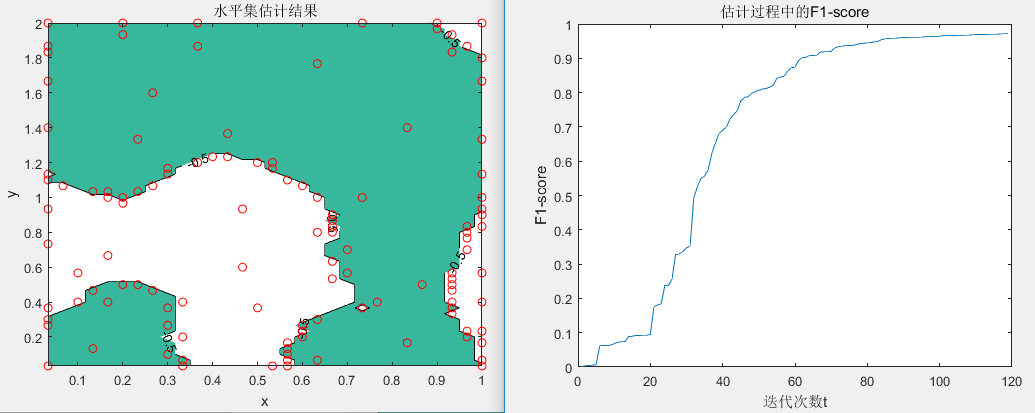
实验一：

LSE:（红圈代表查询点位置，下同）



查询点个数：

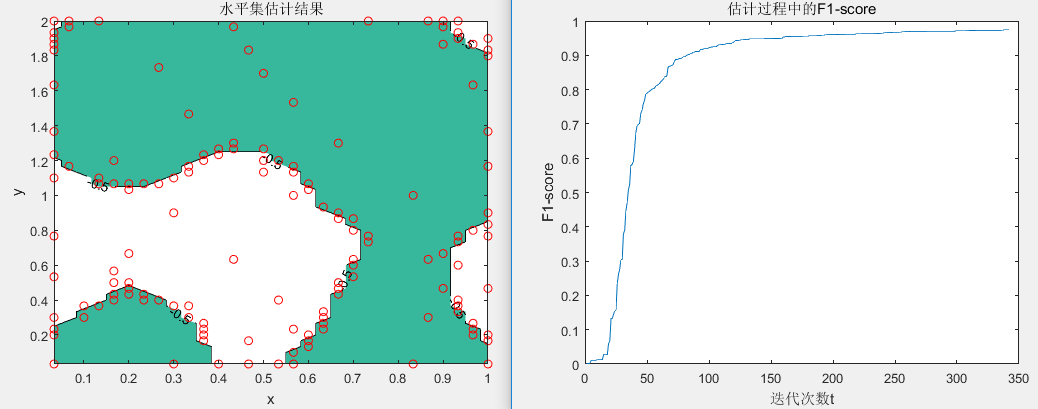
LSE\_fast:



查询点个数：

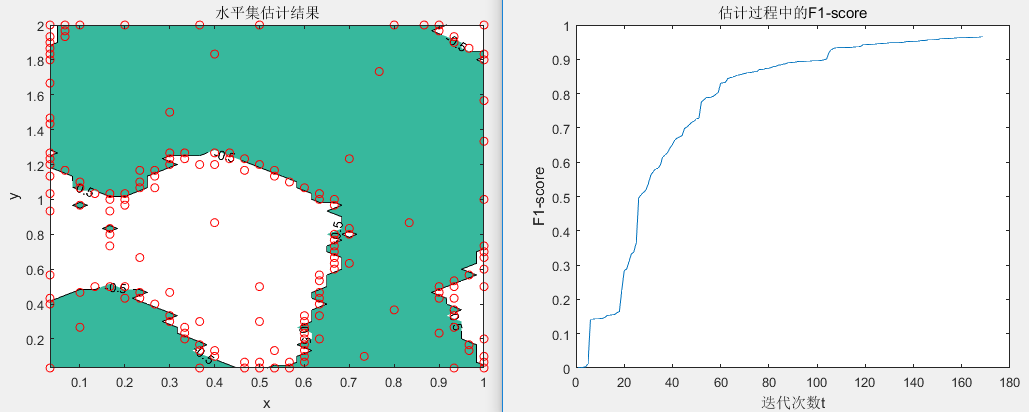
实验二：

LSE:



查询点个数：

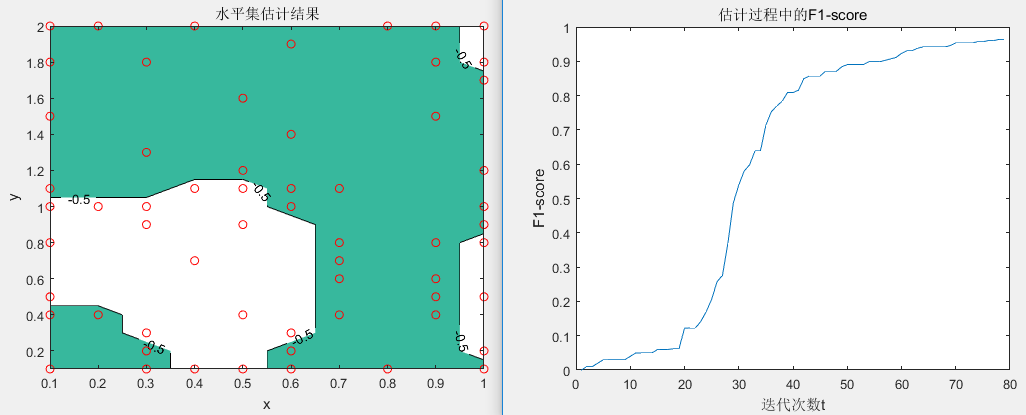
LSE\_fast:



查询点个数：

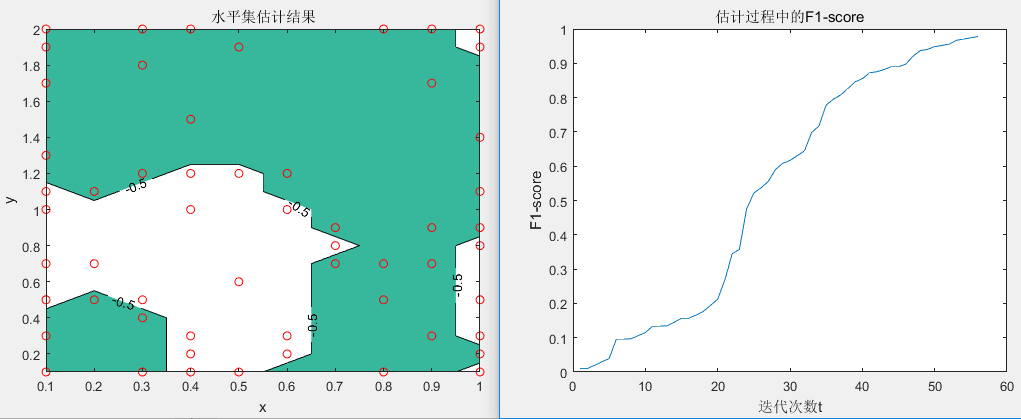
实验三：

LSE:



查询点个数：

LSE\_fast:



查询点个数：

可以看到，上述三个实验中，两种算法分类准确度都很高，具体表现为估计的等高线图和真实值十分相似，且最终F1-score均接近1。分类准确度并不是LSE\_fast改进的目标，因此不再赘述。

现将上述实验一、二、三的迭代次数汇总如下表：

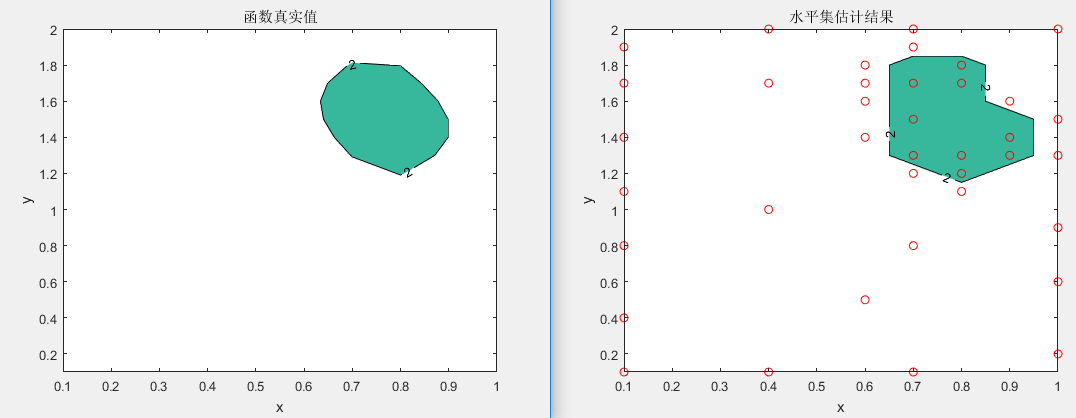
|  |  |  |  |
| --- | --- | --- | --- |
|  | 实验一 | 实验二 | 实验三 |
| LSE查询点个数 | 144 | 343 | 80 |
| LSE\_fast查询点个数 | 120 | 170 | 57 |
| 查询点个数比值(LSE\_fast/LSE) | 0.833 | 0.496 | 0.713 |

通过实验一与实验二对照可知，在高精度要求下，LSE\_fast算法能更显著地减少所需查询点的数量；通过实验一与实验三对照可知，在小样本空间下，LSE\_fast算法能更显著地减少所需查询点的数量。

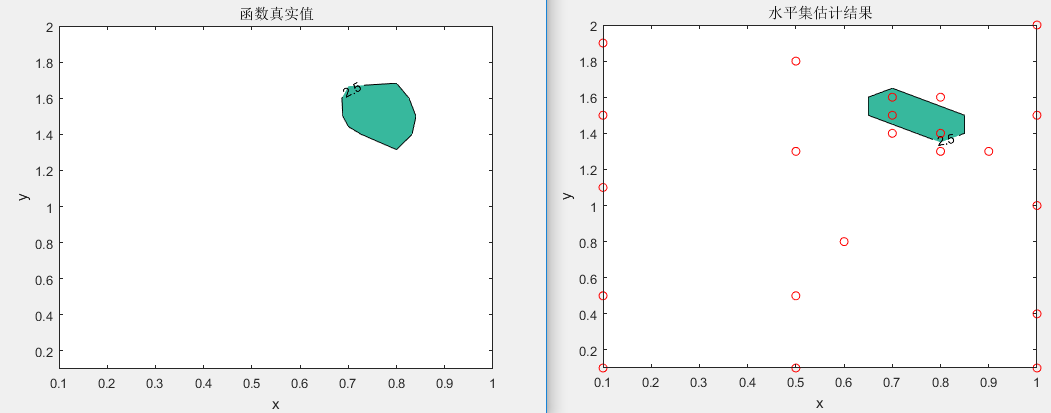
（二）函数最优点估计

当我们的水平集估计很准的时候，便可以用于其他的用途，比如估计函数的最优点，在此我们便小试牛刀，使用LSE\_fast算法寻找函数sin(10\*x)+cos(4\*y)-cos(3\*x\*y)的最大值点。寻找时，只需不断提高阈值参数h，此时估计的上水平集会越来越小，直到满足所需的估计精度。（本实验中）

调整阈值参数h=2时：



调整阈值参数h=2.5



不断增大h，即可逐步逼近原函数的最大值。用这种方法求出的最优点是全局最优点，而我们的方法却并不要求函数的凸性，相信这种方法将在优化领域有重要的贡献。

**参考文献**

[1] Active Learning for Level Set Estimation

[2] Active Learning for Identifying Function Threshold boundaries

[3] Truncated variance reduction

[4] Active Area Search via Bayesian Quadrature

[5] Robust Super-Level Set Estimation using Gaussian Processes