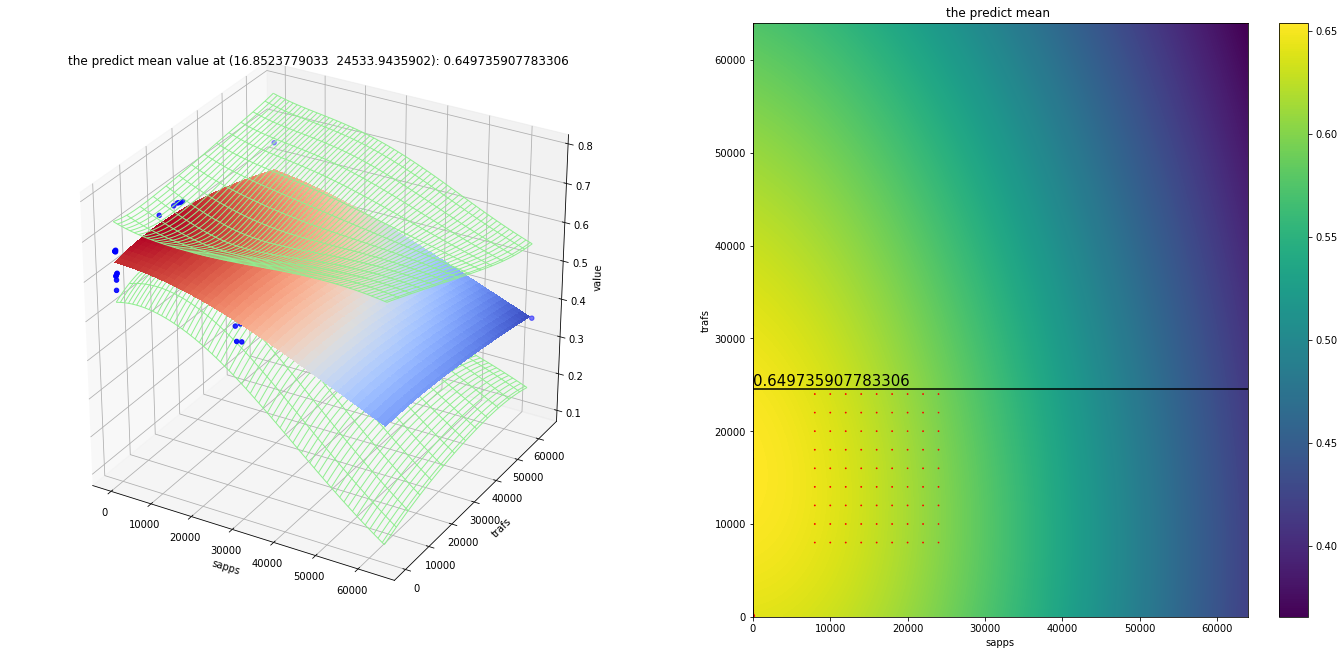
## 2018.05.25:

上一次讨论的想法是利用贝叶斯优化，找到最优的业务层设计参数组合：

V为每次仿真的业务性能评估值

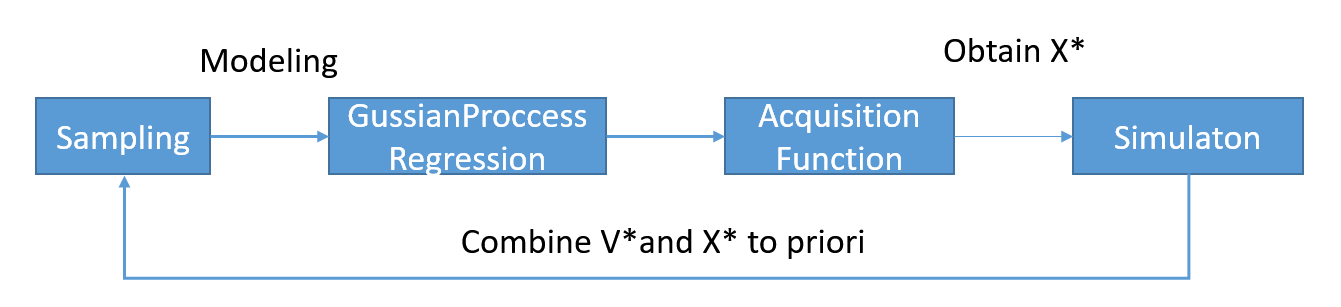
为每次仿真的业务层设计参数组合

演示效果：



上图为BOA

BOA（贝叶斯优化）流程：



上次讨论之后，提出用GMM模型代替贝叶斯优化中的GP模型

经过接近一周的看论文，看材料，初步的想法和问题：

问题：

问题1：GMM模型的初始化：GMM的初始化训练比GP训练所需的先验（仿真）更多，先验越多，模型越准确，在后续的AcquisitionFunction寻找最优点的时候收敛时间也越小。训练样本（大小和分布）怎么样才合适？

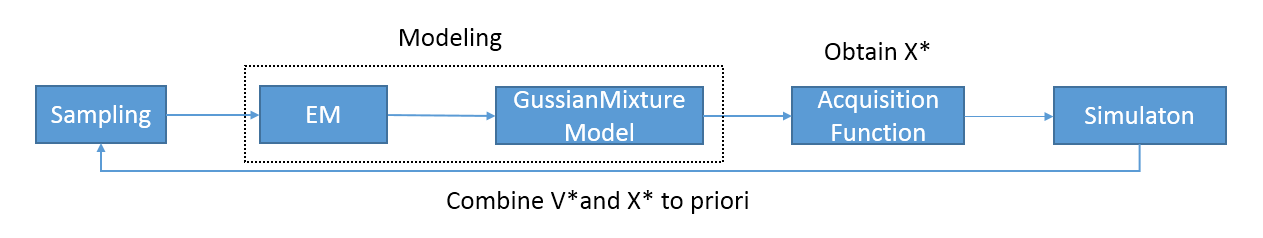
问题2：如果利用GMM模型，AcquisitionFnction如何设置（如何根据GMM模型的权重，均值，方差进行设计？），可以更快的收敛？

问题3：最终达到收敛的条件如何确定？（如何设计阈值，达到阈值则停止优化过程）

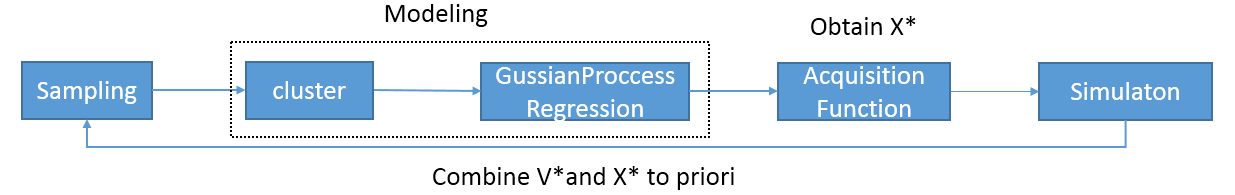
问题4：暂时还没有

改进的BOA方法：（待定）

PLAN A:



PLAN B:



**输入输出：**

方法超参数：components个数（聚类个数）、重复仿真次数、GPR核（包括核的各种参数）、AF函数等

设计参数：superapp包大小、superapp发包间隔、vbr包大小、vbr发包间隔、trafficgen包大小、trafficgen发包间隔等…

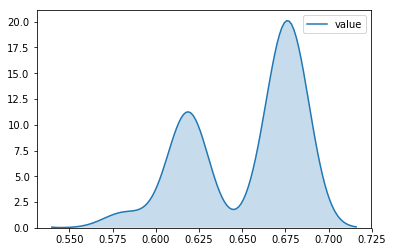
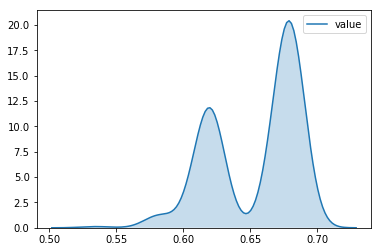
输出指标：三种业务（superapp、vbr、trafficgen）的吞吐量、时延、时延抖动、丢包率

最终的输出：网络评估值

区别是在建模过程中是先对数据进行聚类，然后将高斯分布线性组合还是直接利用GMM模型的EM算法得到GMM模型估计的参数

相当于牺牲了计算资源和时间，对网络模型进行更精确的刻画

对某两组随意的参数设计组合X\*分别进行了240次重复仿真和评估，得到了核密度估计曲线：



上左图为2400\_18000\_20000,

上右图为12000\_16000\_30000

第一个参数为superapp包大小，第二个参数为VBR报大小，第三个参数为TrafficGenerator包大小，可以看出两个比较明显的高斯分布（components）

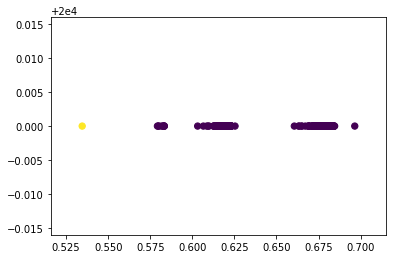
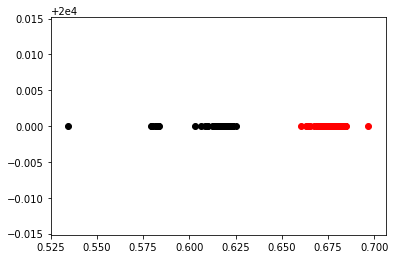
**基于GMM的贝叶斯优化：**

1. 采样：得到先验仿真数据结果和评估结果（目前正在进行先验数据的生成，目前是6个参数的组合中修改一个参数，修改15次，修改一次重复仿真120次）
2. 聚类：选择GMM模型中的component个数（因为上核密度估计图，暂定为2），利用K-means或EM算法思想将数据分类（两类）
3. 建立GMM模型：得到每一个component的权重、均值与方差（得到聚类结果之后对数据进行分类分别进行处理，权重跟数据分类至后的比重相关，均值方差跟GP模型类似）
4. AcquisitionFunction得到预测点：根据GMM模型中的权重均值方差设计预测函数，得到预测点（目前在BOA中用的主要是EI、PI、UCB方法，我觉得这里可以借鉴UCB方法）
5. 预测点迭代仿真：大量仿真（重复仿真的强度大概是多少，需要定量，强度大，模型准确；强度小，耗时短、迭代快）
6. 预测点数据加入先验数据：将仿真得到的预测点数据和先验数据合并，跳到步骤2）

PS：

聚类暂定Kmeans算法思想，不一定利用欧氏距离进行计算，EM算法对初始值（每个component的均值标准差的初值设定）要求高，拟合收敛所需的迭代次数高，作为第二选择进行考虑；

为了优化过程的迭代次数能够更小，前期仿真计算机院足够的情况下，系统能够支持大量的仿真，在AcquisiitonFunction进行选择的时候，拟根据不同的策略，进行多个点（每个策略选取一个点）的选择，并行仿真，加快模型的收敛，在后期采用Exploit策略，加快对优化结果的收敛；



第一个图为KMeans++（迭代300次）聚类（上图）：

第二个图是EM算法（迭代1000次）分类（上图）：

## 2018.06.06:

仿真数据跑了1800组（120\*15）从5月21号

新问题4：聚类的方式

1. 对每次仿真的数据进行评估之后，全部一起进行聚类（先验数据只进行一次聚类，但是在有新数据加入之后要重新进行全部数据的聚类）
2. 对相同的设计参数组合的仿真数据进行评估后进行聚类，最后把所有的数据再合并（先验数据的聚类次数多，但是对后续新数据加入之后，只需要对新数据进行聚类）

而且不同的聚类方式对先验数据的聚类结果有影响

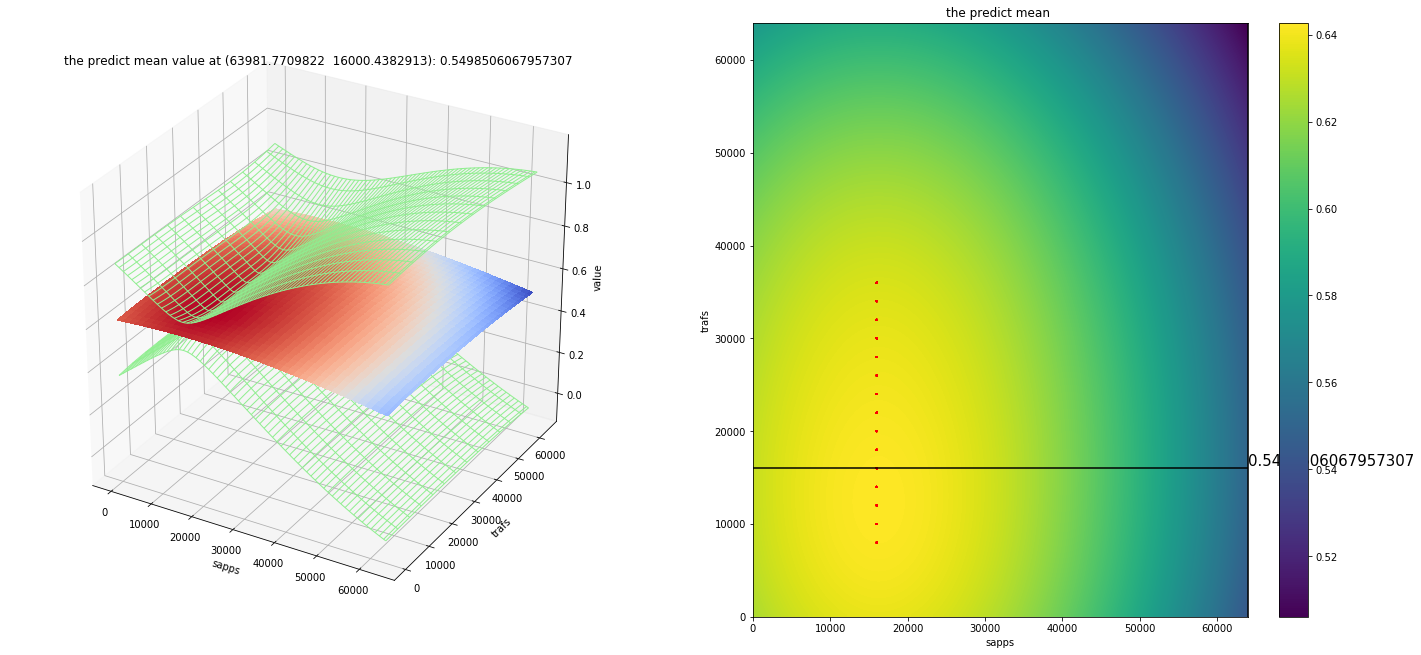
新问题5：仿真数据生成时间过长

仿真120\*15组数据大概花了16天一次仿真跑15-20分钟，需要提高仿真的效率

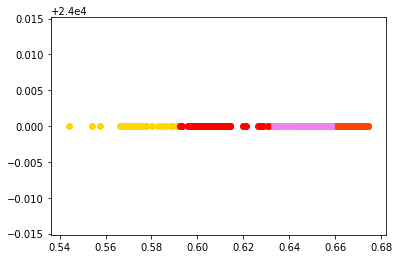
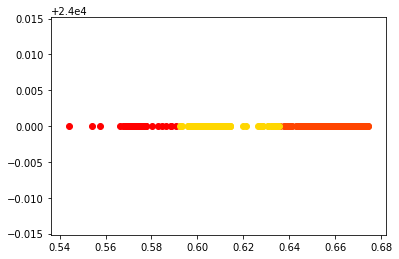
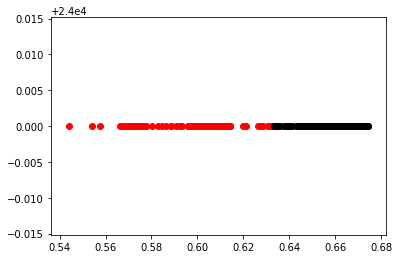
问题2：采集函数的设计

下一步的目标是采集函数的设计和聚类方式的比较

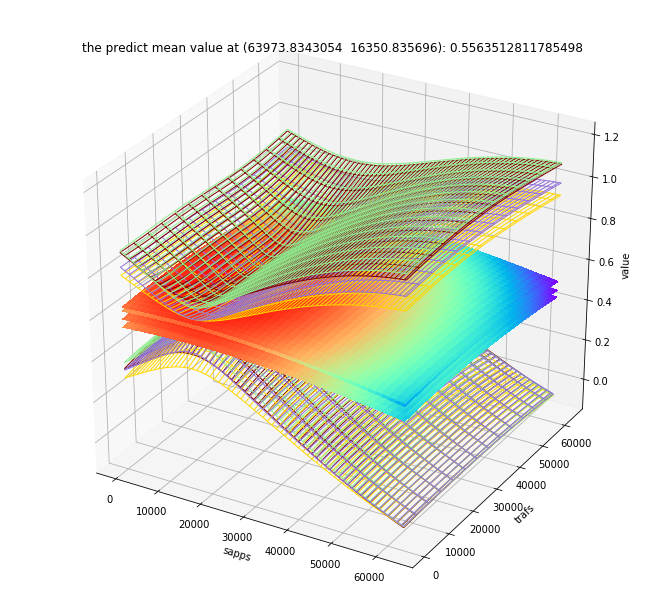
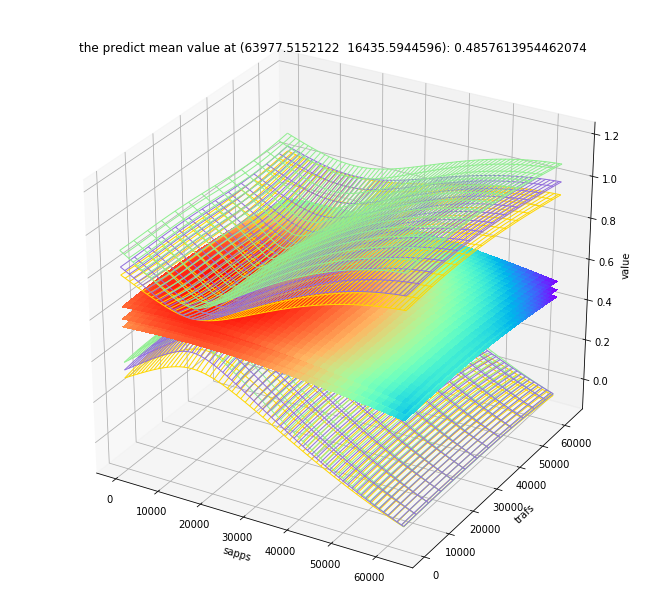
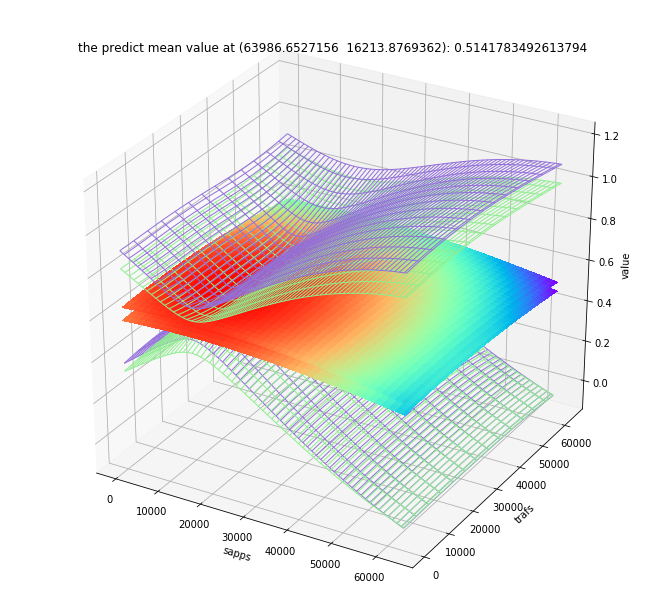
暂时可以做到4个components的高斯混合模型的三维热力图：



1 component高斯混合模型拟合结果，（上图）退化为高斯过程模型



上图2,3,4components聚类结果（采用的下列第一种聚类方式）



2 components高斯混合模型拟合结果（上左图）3 components高斯混合模型拟合结果（上中图）4 components高斯混合模型拟合结果（上右图）

现在目前的Acquisition Function有三种策略（主流）

1. expected improvement
2. probability of Improvement
3. upper confidence bound

## 2018.06.09:

讨论：

江老师：

1. 把评估去掉（评估的主观性太强，容易产生质疑），直接对指标与输入参数之间进行分析，然后对各个分布模型进行统计评估
2. 需要考虑如何验证优化的结果，如何用小空间样本验证全空间中的最优解
3. 需要对仿真速度进行优化

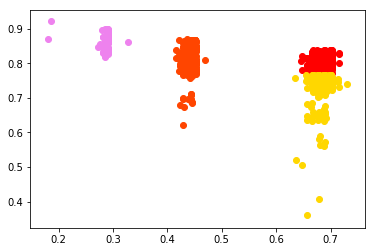
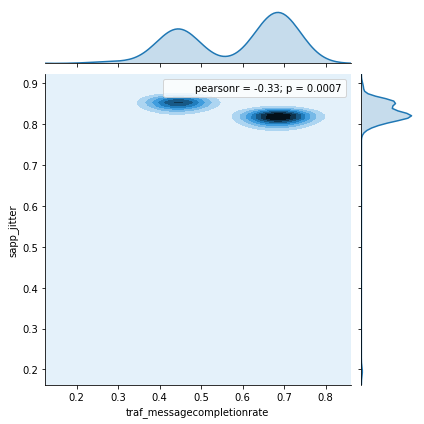
针对3）问题：现在把业务流从5\*4条改为3\*4条；同时在仿真程序的流程中，把动态生成stk报表的过程省略，变成直接读取本地的已有的报表。大概综合下来能够省略50%的时间（原来一次仿真过程大概在20分钟左右）但是在有些仿真中，数据有些问题，这里我就没管了，当做异常值处理。

初始采样方式\*：（Latin Hypercube Sampling）文献LHSMDU方法：Latin Hypercube Sampling with Multidimensional Uniformity

准备尝试

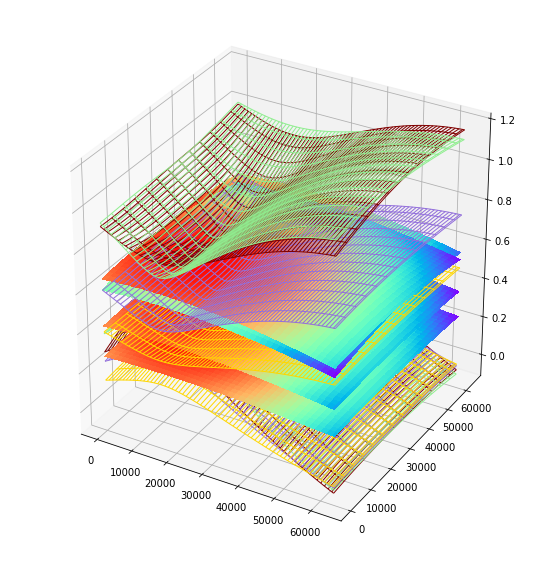
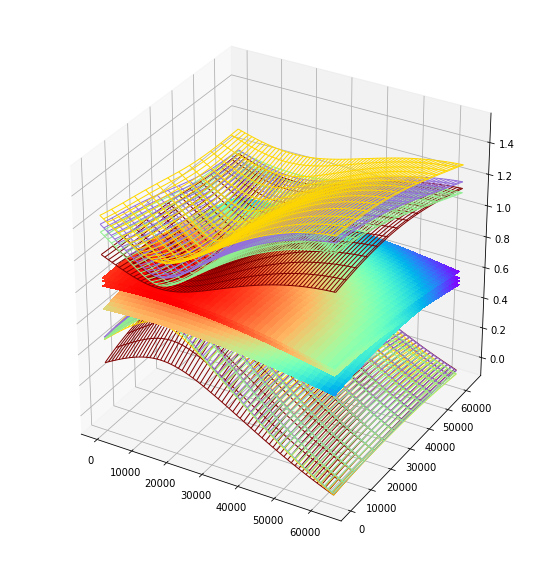
针对1）问题：现在如果选取一个qos指标目前是sapp\_jitter，做优化，则是与原来的优化方式没有区别，若是选取多个qos指标，（这里选取了两个：sapp\_jitter, traf\_messagecompletionrate）则会出现下列问题：

1. 聚类方式：聚类采用的是Kmeans++算法，在低纬度状态下聚类效果较好，在高维状态则很大可能没有好的结果，需要重新考虑
2. 聚类个数，针对不同的具体qos指标，分布有可能不同具体聚类的个数需要统计估计



上左图为120个仿真结果的sapp\_jitter联合traf\_messagecompletionrate的KDE分布图

上右图为sapp\_jitter联合traf\_messagecompletionrate 的Kmeans 聚类结果（4 cluster）



上图为sapp\_jitter的 GMM模型（4 clusters） 上图为traf\_messagecompletionrate的GMM模型（4 clusters）

**对于单一value（或指标）的GMM模型AF函数方法**

对于acquisition function的设计，借鉴UCB的策略

设计1：（已实现）

对聚类的个数分别对每个拟合的components预测一个Query point

与BOA中的UCB原理一样，有几个GMM成分，就有几个Query point

设计2：（已实现）

对于acquisition function的设计，借鉴UCB的策略

1. 首先将每个components的权重确定（通过聚类结果，对每一类的权重进行确定）
2. 每一个components对于每一个query point都会有一个预测的均值与标准差，利用UCB的思想，简单高效：

设权重为n为components个数（聚类个数）有

即第n个components采样点的个数与样本总体个数之比

且­

3）设第n个components对于query point的预测均值与标准差为、

则第n个components的upper confidence bound（UCB）为

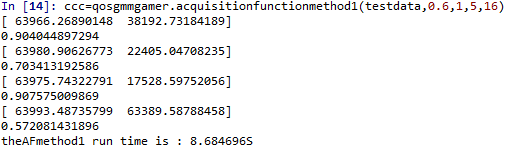
理解为标准差的重要程度，kappa值越高quarypoint越倾向于选择探索（explore）这个值需要人为设定

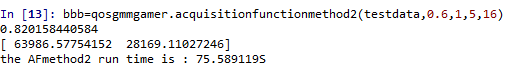
4）则总体上的UCB为

5）对于取到最大UCB的点即是下一次仿真过程的query point

最终得到一个Query point

对比：

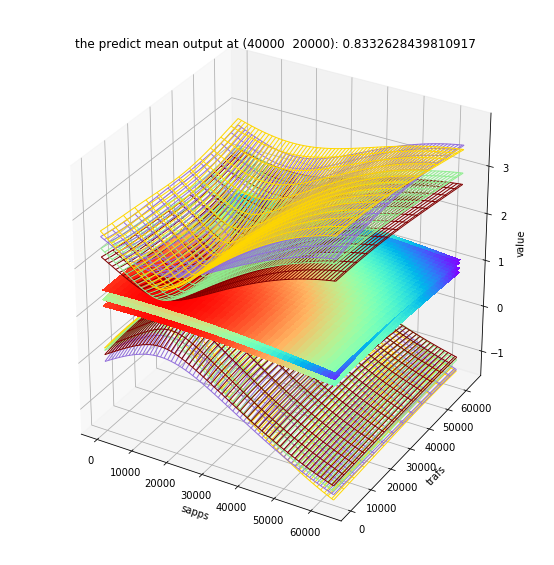
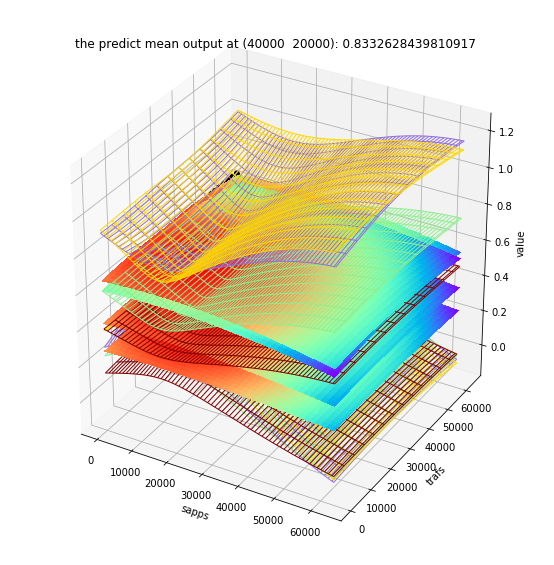
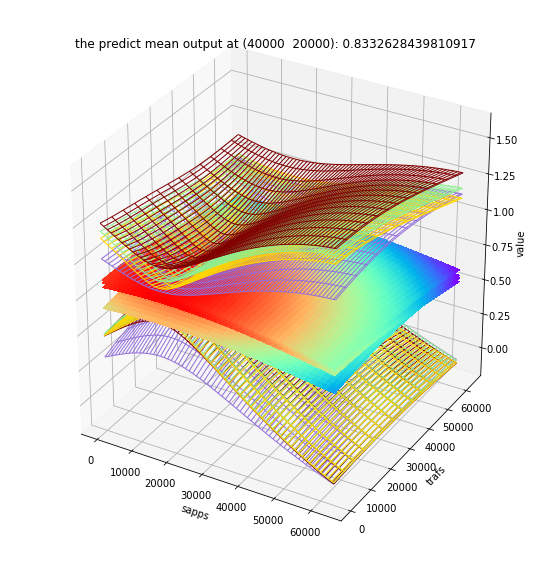
如图为设计1的实现效果，耗时少

如图对于设计2的实现效果

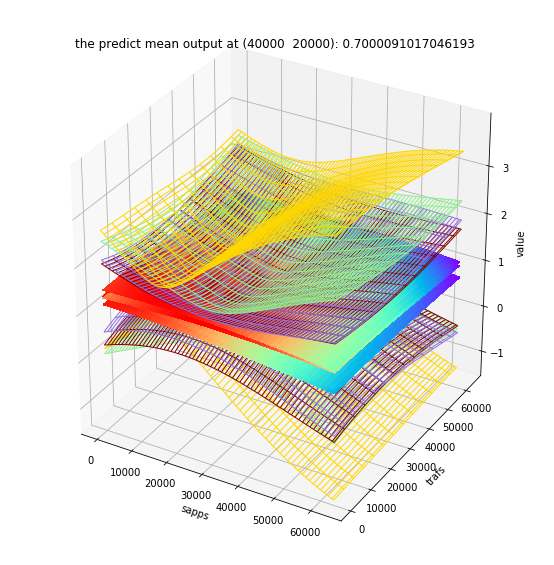
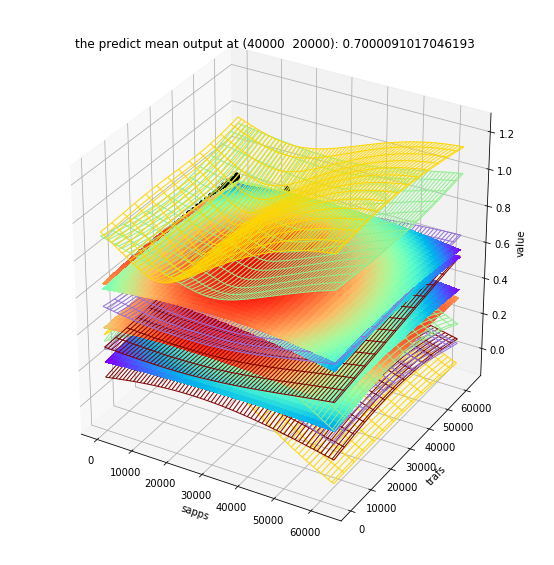
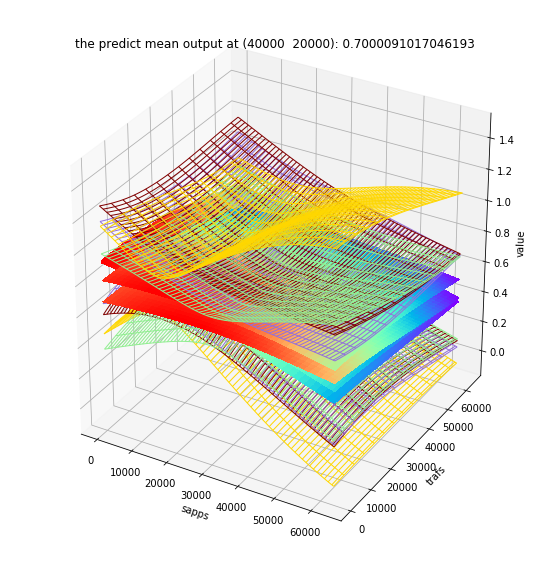
## 2018.06.16:

**多指标模型加权：**

现实现trafficgenerator丢包率和superapp抖动的模型加权（均值，标准差如下右图）（程序功能已实现，细节未优化）



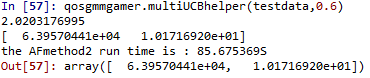
上面三个是原始训练集（16000\_24000\_8000-36000,每组120次仿真）的数据拟合的模型（sapp抖动，traf丢包率，指标加权）



上面是新训练集（16000\_24000\_8000-36000,30000\_24000\_8000-36000,每组60次仿真）数据拟合的模型（sapp抖动，traf丢包率，指标加权）

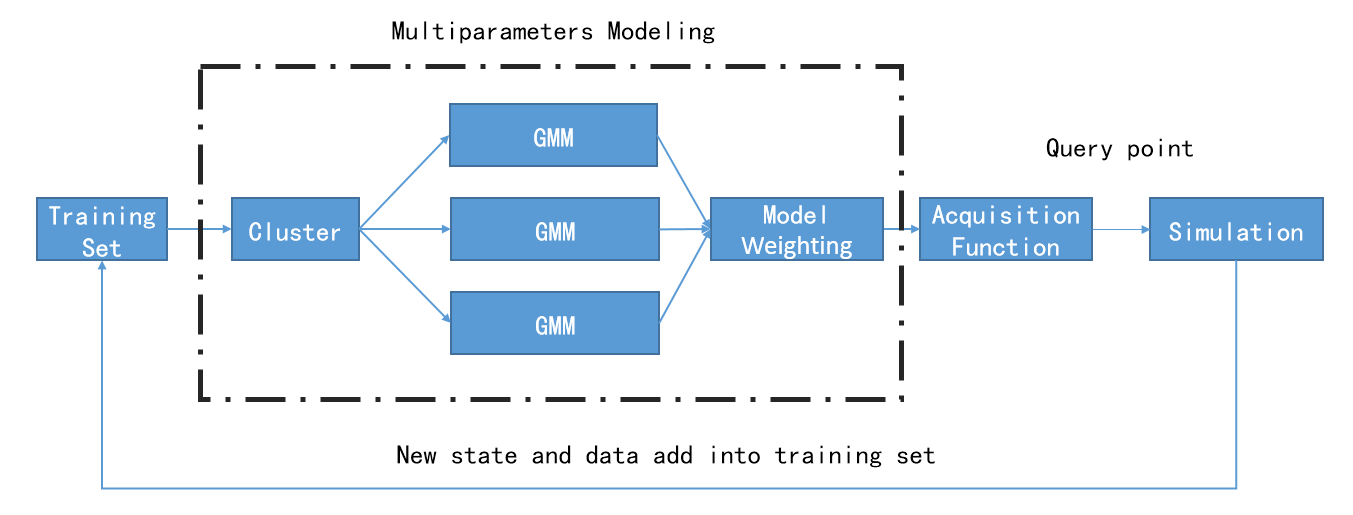
现在需要对多指标的模型加权进行acquisitionfunction的实现（已实现功能 细节未优化）

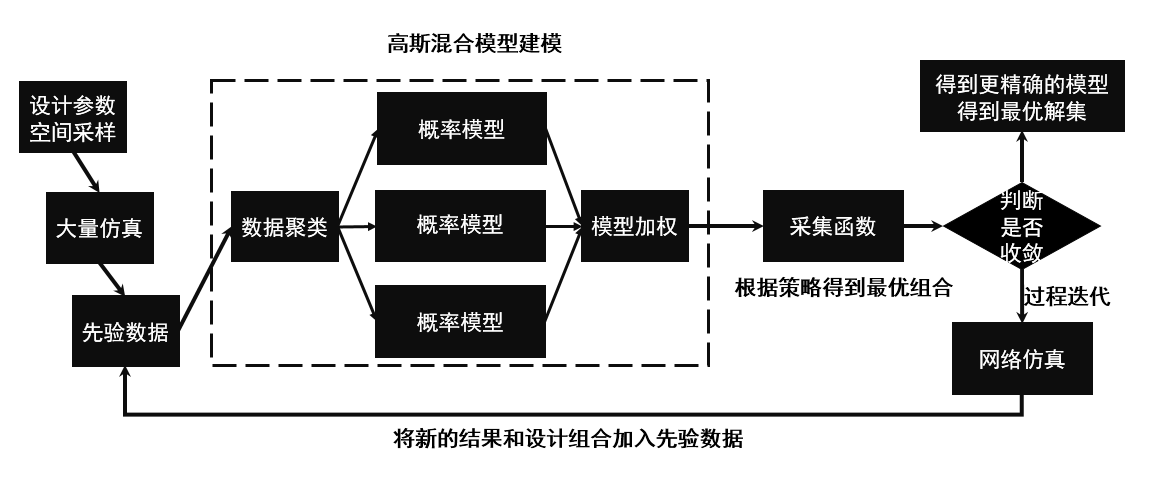
目前拟采用UCB方式设计

 左图为左束支综合的AF函数的结果

**进行流程的实现：**

新的动态优化的流程有些变动：





下一步是进行代码的优化，然后进行整个流程的自动化编程（现不影响反馈过程暂时注释了,目前主要的已经完成）

代码出了点问题，在加入新的数据的时候，出现错误（operands could not be broadcast together with shapes (500,) (502,)）

自动优化过程已经实现，正在试验（优化了仿真，大概7分钟一次仿真，进行一次反馈过程大概7\*30+5mins ）

为了快速出结果，现在新的query point仿真做30次

**问题：**

1）AF函数的参数设置（目前使用的UCB，还存在很多POF,EHI等，如何确定，需要比较，但是仿真实验太慢了）

2）最优解的收敛性（？最终可能得到的是一个pareto解集，在目前的实验里面两个指标之间的多目标优化逻辑已经可以收敛，更多的情况下需要实验）

3）聚类的方法的区别（目前使用k-means++，比较其他的同样是仿真太慢了）

4）多少次反馈之后的目标跟你和模型可以达到最优（不能自证最优）

目前完成了一次丢包率和时延抖动的联合自动建模和优化的过程

效果不是很明显

问题：在不同尺度的数据之间的模型比较，有些qos输出指标模型对设计指标不敏感，有一些比较敏感，怎么进行多目标优化