Scalable Global Optimization via Local Bayesian Optimization

- 作者：David Eriksson, Michael Pearce, Jacob R Gardner, et al
- 机构：uber，warwick大学
- 会议：NIPS2019
- 地址：http://papers.nips.cc/paper/8788-scalable-global-optimization-via-local-bayesian-optimization.pdf
- 代码：https://github.com/uber-research/TuRBO

论文主要内容

摘要

Bayesian optimization在上千observation、high dim情况仍然面临挑战。本文中认为这是由global acquisition的**同质性**和**过分探索**导致。本文提出TurBO算法来拟合一堆local model并在这些local models上通过隐式bandit方法采样。通过综合比较显示TurBO取得SOTA水平。

贡献

1. 通过构造local model而不是global model，解决在large scale下的BO的performance问题

研究内容

Local model

选择GP 作为local model：

- 简单的model对noisy observation效果不好
- 简单的model需要的trust regin过小(保证准确的model)

Trust Regin

选择超矩形，超矩形的每个dim的长度为 L_i

- noise-free：设置 x^* 为目前最佳observation(如果存在噪声设为surrogate model的最小后验均值)，把该点作为超矩形的center
- base side length：初始化 $L \leftarrow L_{\text{init}}$

- $L_i = \lambda_i L / \left(\prod_{j=1}^{d} \lambda_j \right)^{1/d}$ 保持整个volume还是 L^d ， λ_i 是第 i 个高斯过程模型的length scale
- 如果能够连续**成功**找到比当前观测更好的candidate，则增大TR；如果连续**失败**，则减小TR
- 设定阈值 L_{min} ,如果比阈值小则discard TR，并重新初始化
- 设定阈值 L_{max} ,不让TR比阈值还大

Trust region Bayesian optimization

- 总体会**同时**维护 m 个TR，类似于多臂老虎机，形成探索开发的trade-off
- 每次迭代都需要从所有TR中选择 q 个（a batch）candidates：
  $\mathbf{x}_i^{(t)} \in \underset{\ell}{\operatorname{argmin}} \underset{\mathbf{x} \in \mathrm{TR}_\ell}{\operatorname{argmin}} f_\ell^{(i)}$ where $f_\ell^{(i)} \sim \mathcal{GP}_\ell^{(t)} \left(\mu_\ell(\mathbf{x}), k_\ell \left(\mathbf{x}, \mathbf{x}' \right) \right)$
- acquisition max过程具体通过tompson采样得到

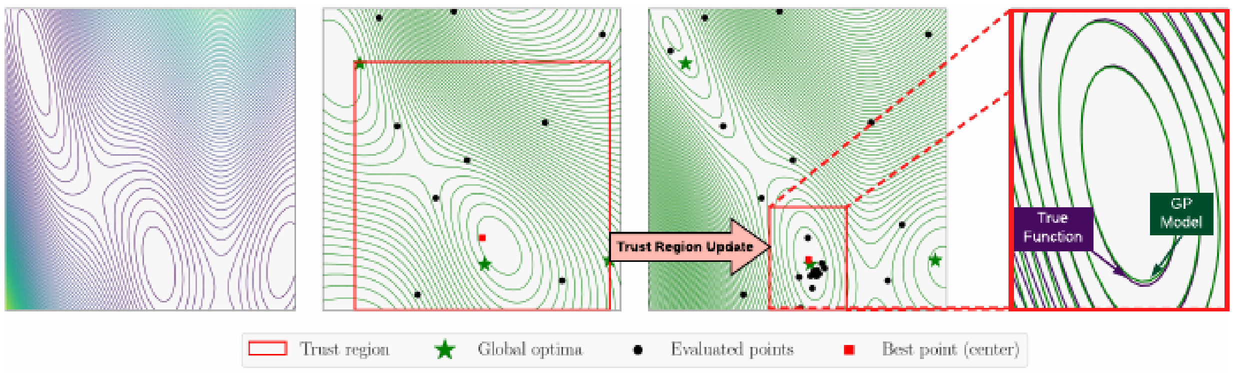

Figure 1: Illustration of the TuRBO algorithm. **(Left)** The true contours of the Branin function. **(Middle left)** The contours of the GP model fitted to the observations depicted by black dots. The current TR is shown as a red square. The global optima are indicated by the green stars. **(Middle right)** During the execution of the algorithm, the TR has moved towards the global optimum and has reduced in size. The area around the optimum has been sampled more densely in effect. **(Right)** The local GP model almost exactly fits the underlying function in the TR, despite having a poor global fit.

伪代码：

Python

```python
def suggest():
    for b in 1:batch_size
        for i in 1:m_trustRegin
            sample_func = gp_model[i].posterior.sample() # sample func inst
            x_candidates[b,i,...] = gen_candidates_grid(trustRegin[i], dim) # (candidates_num, dim)
            y_candidates[b,i,...] = sample_func(x_candidates) # (candidates_num, 1)
    # x_candidates: (b, m, c, dim), y_candidates: (b, m, c, 1)
    #idx = argmin(y_candidates, dim=[1,2]) # 最promising的前b个作为suggest结果
    for b in 1:batch_size
        idx_m, idx_c = y_candidates[b].argmin(dim=[1,2])
        idx[b].append(idx_m) # 标识该suggest来自哪个trust regin
        x_suggests[b, :] = x_candidates[b, idx_m, idx_c, :] # (b, dim)
    return x_suggests, idx

def observe(batch_x, batch_y, idx): # batch_x: (b, dim), batch_y: (b, 1)
    for i in 1:m_trustRegin
        x, y = batch_x[idx==i], batch_y[idx==i]
        gp_model[i].fit(x, y)
        trustRegin[i].update(y)
```

实验

实验结果

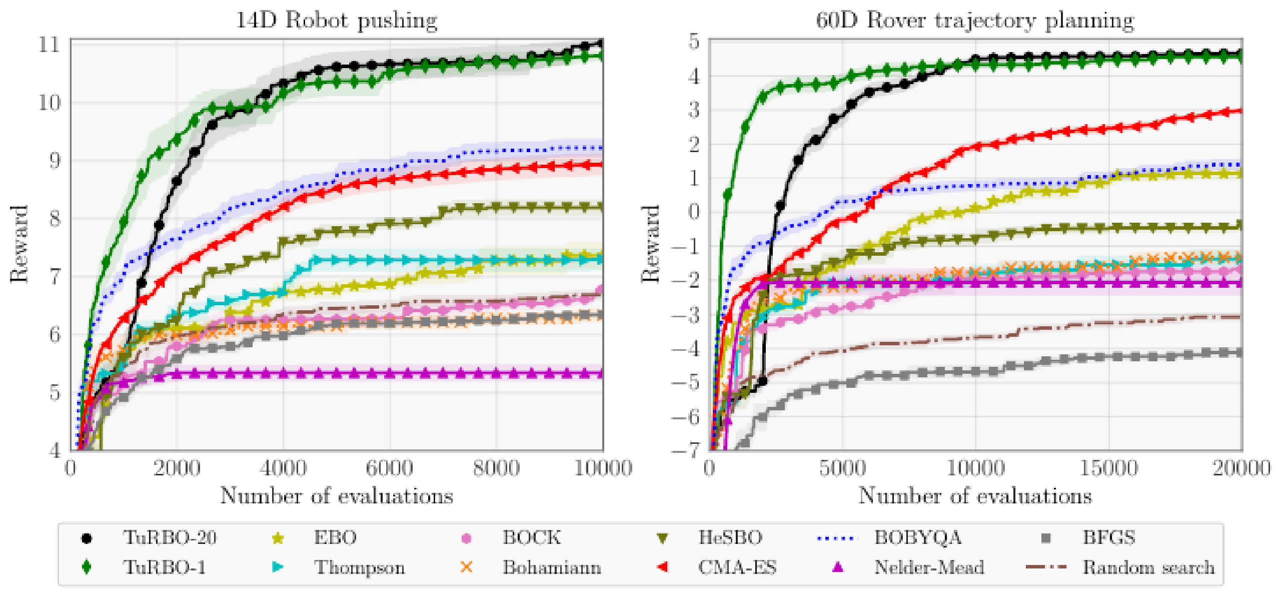

Figure 2: **14D Robot pushing (left):** TuRBO-1 and TuRBO-20 perform well after a few thousand evaluations. **60D Rover trajectory planning (right):** TuRBO-1 and TuRBO-20 achieve close to optimal objective values after 10K evaluations. In both experiments CMA-ES and BOBYQA are the runners up, and HeSBO-TS and EBO perform best among the other BO methods.

TuRBO-1和TuRBO-20的优势明显

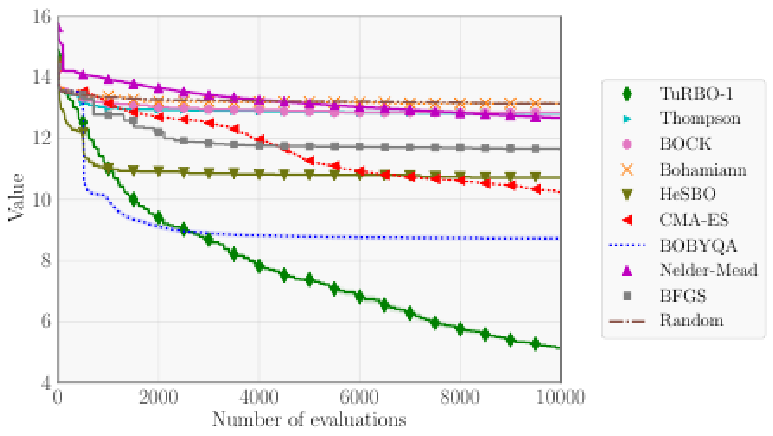

Figure 4: **200D Ackley function:** TuRBO-1 clearly outperforms the other baselines. BOBYQA makes good initial progress but consistently converges to sub-optimal local minima.

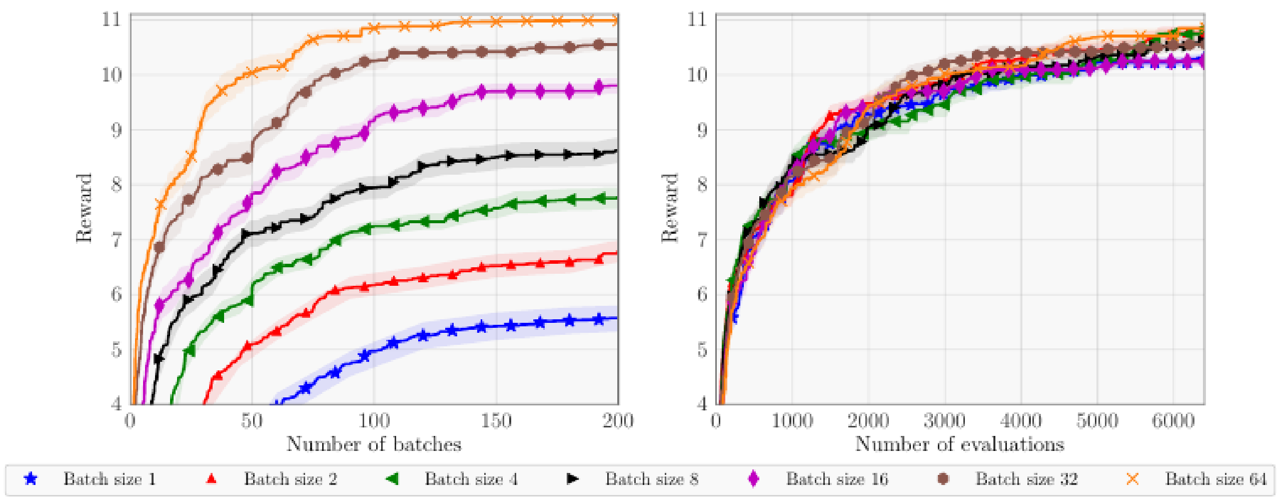

Figure 7: We evaluate TuRBO for different batch sizes. On the left, we see that larger batches provide better solutions at the same number of steps. On the right, we see that this reduction in wall-clock time does not come at the expense of efficacy, with large batches providing nearly linear speed up.

左：更大的batch size表现更好

右：这个实验结果说明，TurBO并不因为batch size增加降低sample efficiency

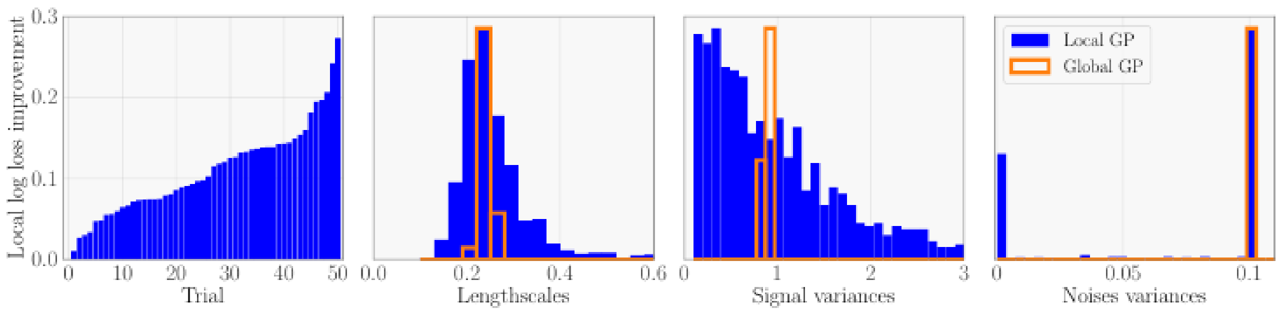

Figure 5: **Local and global GPs on log loss (left):** We show the improvement in test set log loss (nats/test point) of the local model over the global model by repeated trial. The local GP increases in performance in every trial. Trials are sorted in order of performance gain. This shows a substantial mean improvement of 0.110 nats. **Learned hypers (right three figures):** A histogram plot of the hyperparameters learned by the local (blue) and global (orange) GPs pooled across all repeated trials. The local GPs show a much wider range of hyperparameters that can specialize per region.

- 这是一个bimodel例子，从图最右边可以看到，local model能适应异方差性