Scalable Global Optimization via Local Bayesian Optimization

David Eriksson, Michael Pearce, Jacob R Gardner,
Ryan Turner, Matthias Poloczek

MAMATHA YARRAMANENI

Road Map

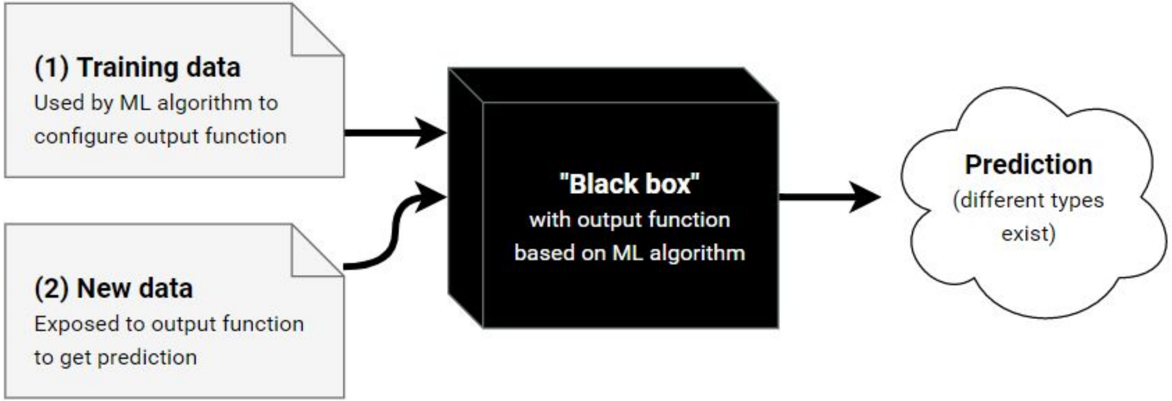
- Overview
- Basics
- Introduction
- Trust Region Bayesian Optimization Algorithm (TurBO)
- Numerical Experiments
- Conclusion

Problem Statement

- Bayesian Optimization(BO) has emerged as a popular optimization method for black box techniques
- Applying bayesian optimization to high-dimensional problems with thousands of observations is challenging.
- BO not competitive with other paradigms for difficult problems
- Introducing TurBO algorithm
- TurBO uses the design of a local probabilistic approach for global optimization of large-scale high-dimensional problems.
- TurBO uses implicit bandit approach.

What is a Black-box function?

- Only input and output are known to the user.

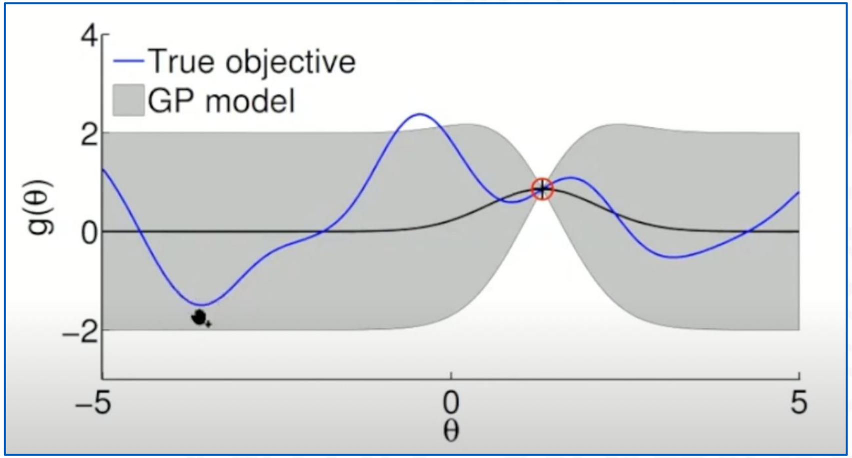


How to optimize?

- We need to optimize $f(x)$ (which is unknown)
- Cannot use convex optimization
- Cannot compute $f'(x)$ because $f(x)$ is unknown
- We use Bayesian optimization

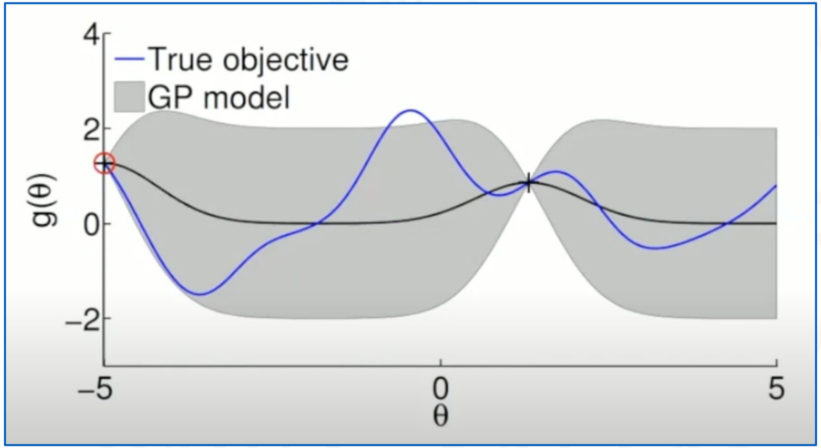
What is Bayesian Optimization?

- Objective function f is not known and needs to be evaluated in minimum number of function evaluations..
- We model function $f(x)$ using Gaussian Process (GP).
- GP acts as a surrogate to the model.



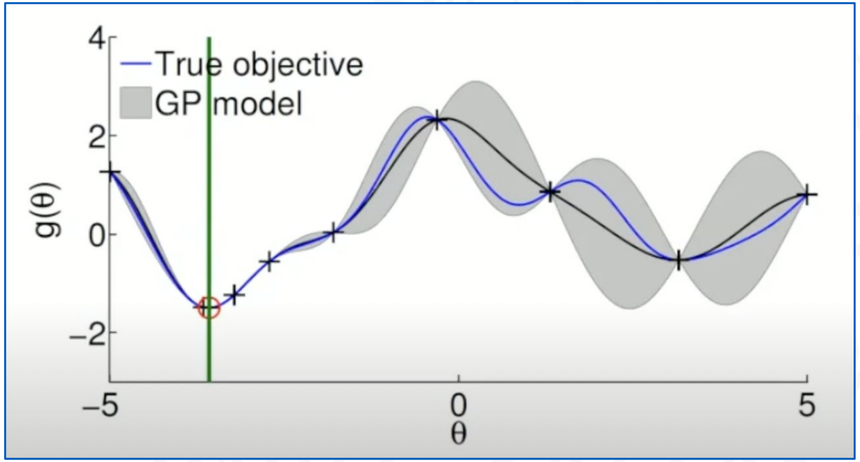
What is Bayesian Optimization?

- Iteratively sample points in the search space.
- GP learns more about the function as we evaluate it.
- Acquisition function uses GP to sample next point as a potential optimal point.
- Update the surrogate function.
- Continue the loop until:
  - Surrogate function maximum does not change.
  - variance is below threshold.
  - f is exhausted.



What is Bayesian Optimization?

- Iteratively sample points in the search space.
- GP learns more about the function as we evaluate it.
- Acquisition function uses GP to sample next point as a potential optimal point.
- Update the surrogate function.
- Continue the loop until:
    - Surrogate function maximum does not change.
    - variance is below threshold.
    - g is exhausted.



What is Multi-armed bandit approach?

- Maximize the total reward in the long run.
- Cannot access true bandit probability distributions
- Learned via trial and error
- Exploration
- Exploitation

Introduction

- Global optimization of high-dimensional black-box functions - important task in
  - Hyperparameter tuning
  - Searching for optimal parameterized policy in reinforcement learning
- BO is good for such problems, but scales poorly for high dimensions and large samples

Introduction

- Optimization of high-dimensional problems is hard:
  - Search space grows exponentially with dimension.
  - Faster search space growth due to the curse of dimensionality -> inherent presence of regions with large posterior uncertainty -> results in over exploration and fails to exploit promising areas.

Introduction

- To overcome these challenges:
  - Introducing TurBO - Trust Region Bayesian Optimization.
  - Simultaneous local optimization runs using independent probabilistic models.
  - Each local surrogate model does not suffer from over exploration problem.

TuRBO

- Trust region Bayesian Optimization algorithm.
- For optimizing high-dimensional black-box functions
- maintain multiple local models simultaneously
- allocate samples via implicit multi-armed bandit approach.
- Trust region → sphere or a polytope → centered at the best solution,

$$\text{Find } \mathbf{x}^* \in \Omega \text{ such that } f(\mathbf{x}^*) \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega,$$

$$f : \Omega \to \mathbb{R} \text{ and } \Omega = [0,1]^d$$

$$y(\mathbf{x}) = f(\mathbf{x}) + \varepsilon$$

$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$

TuRBO

- Use Gaussian Process surrogate model within a Trust Region(TR)
- TR be hyperrectangle
- best solution be x*
- Absence of noise -> set x* to best observation so far.
- Presence of noise -> x* to observation with smallest posterior mean.

TuRBO

- Initialize base side length of the TR to L ← Linit.
- Side length for each dimension is given by:

$$L_i = \lambda_i L / (\prod_{j=1}^{d} \lambda_j)^{1/d}$$

- Total volume is given by L^d.

TuRBO - TR rules for each local region

- single local optimization run, acquisition function at each iteration t to select a batch q candidates within TR. $\{\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_q^{(t)}\}$
- L is large enough for all points = global BO. So, L's evolution is critical.
- After sampling ->
  - "Success" -> x* improves
  - "Failure" -> no improvement in x*
- After Tsucc consecutive success → double the size of TR, $L \leftarrow \min\{L_{\max}, 2L\}$
- After Tfail consecutive failures → halve the size of TR, L ← L/2.
- Reset success and failure counters to zero after size of TR change.
- If L falls below Lmin, discard the TR and initialize a new one.

TuRBO

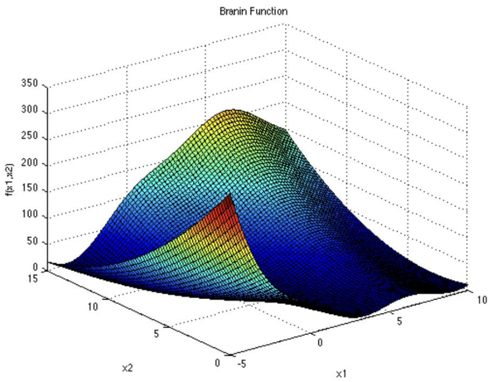
- Making this algorithm global using Random restarts → inefficient
- So, TuRBO maintains m trust regions simultaneously.
- Each TR ℓ with ℓ ∈ {1,…,m} is a hyperrectangle of base side length Lℓ <= Lmax, uses independent local GP model.
- In each iteration, select a batch of q candidates drawn from the union of all trust regions.
- Thompson sampling to select candidates within a TR and across the set of TRs simultaneously.
- $$f_\ell^{(i)} \sim \mathcal{GP}_\ell^{(t)}(\mu_\ell(\mathbf{x}), k_\ell(\mathbf{x}, \mathbf{x}'))$$

TuRBO

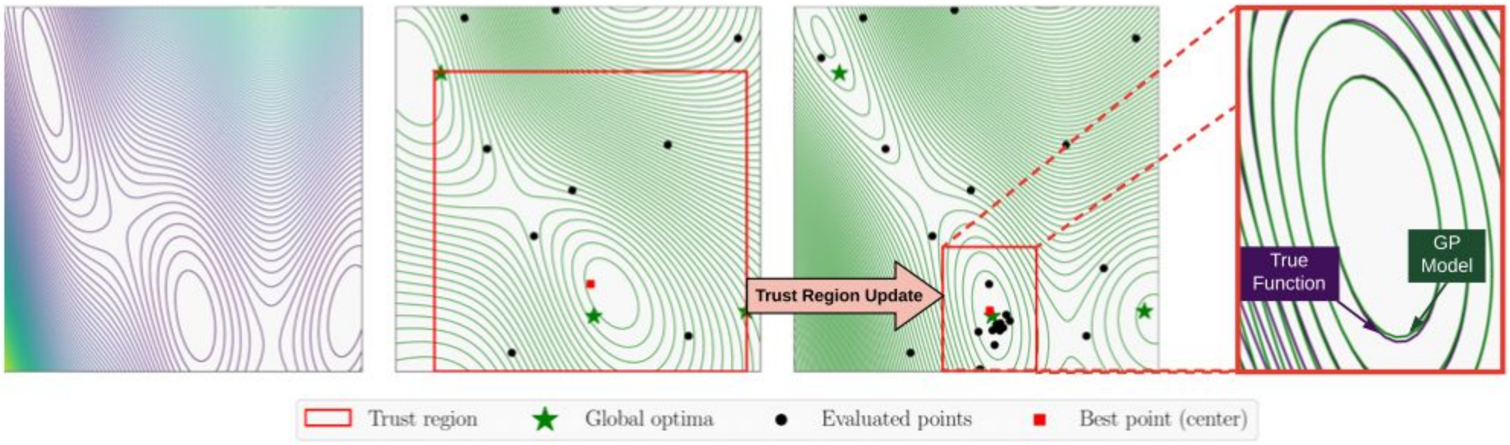
$$f_\ell^{(i)} \sim \mathcal{GP}_\ell^{(t)}(\mu_\ell(\mathbf{x}), k_\ell(\mathbf{x}, \mathbf{x}'))$$

$$\mathbf{x}_i^{(t)} \in \underset{\ell}{\operatorname{argmin}} \underset{\mathbf{x} \in \mathrm{TR}_\ell}{\operatorname{argmin}} f_\ell^{(i)} \text{ where } f_\ell^{(i)} \sim \mathcal{GP}_\ell^{(t)}(\mu_\ell(\mathbf{x}), k_\ell(\mathbf{x}, \mathbf{x}'))$$

TuRBO



$$f(\mathbf{x}) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1-t)\cos(x_1) + s$$



Numerical Experiments

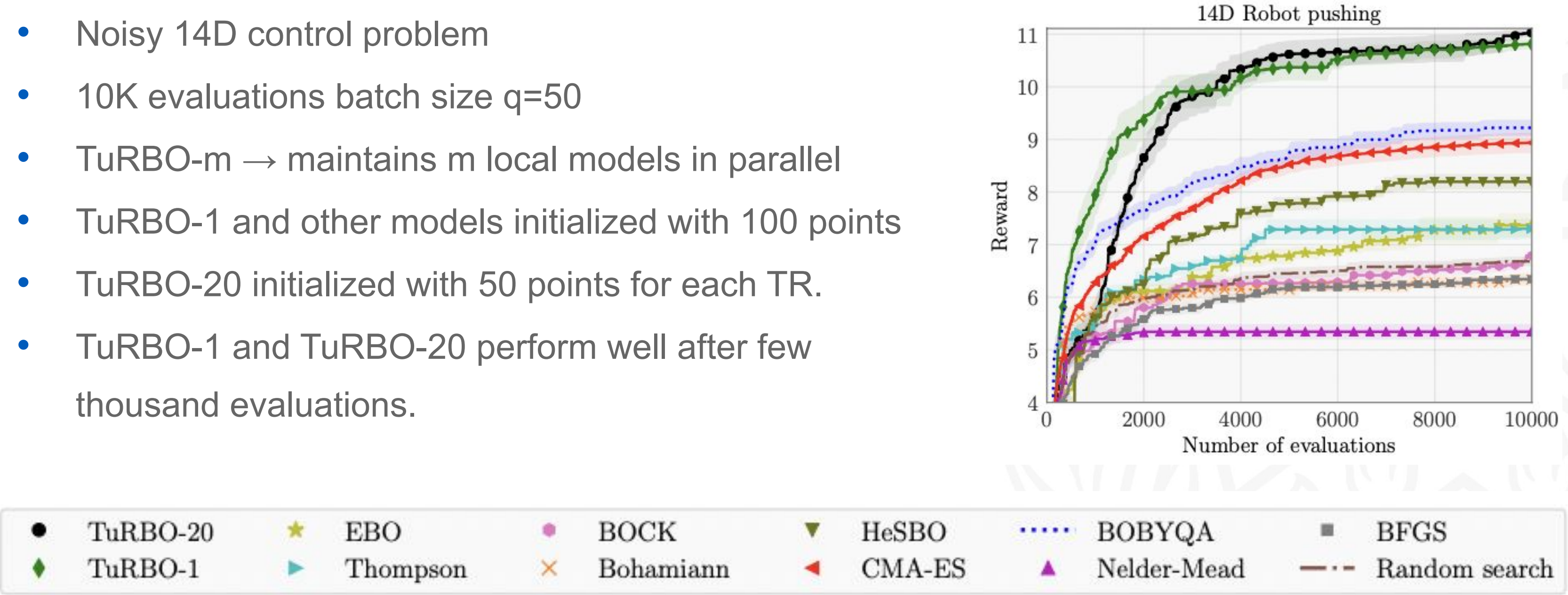
- Evaluate TuRBO on wide range of problems:

  14D robot pushing problem, 60D rover trajectory planning problem, 12D cosmological constant estimation problem, 12D lunar landing reinforcement problem, 200D synthetic problem.
- Compare TuRBO with:

  BFGS, BOCK, BOHAMIANN, CMA-ES, BOBYQA, EBO, GP-TS, HeSBO-TS, Nelder-Mead(NM), random search(RS)

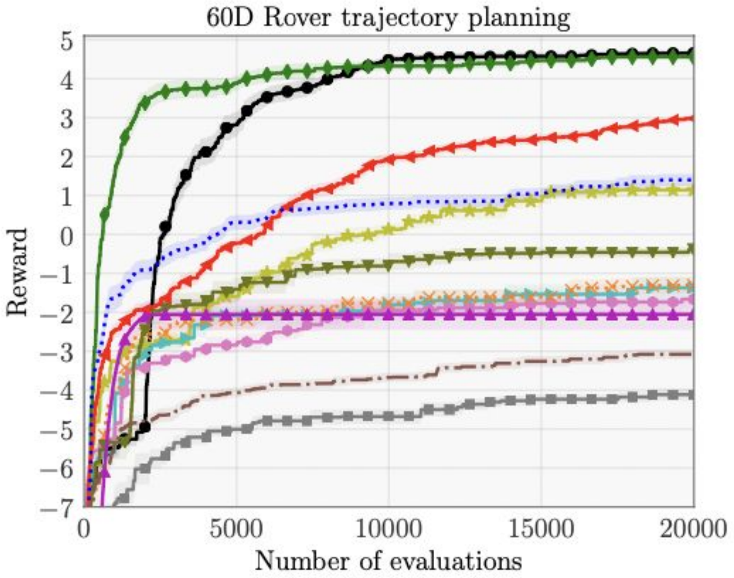
Robot pushing

- Noisy 14D control problem
- 10K evaluations batch size q=50
- TuRBO-m → maintains m local models in parallel
- TuRBO-1 and other models initialized with 100 points
- TuRBO-20 initialized with 50 points for each TR.
- TuRBO-1 and TuRBO-20 perform well after few thousand evaluations.



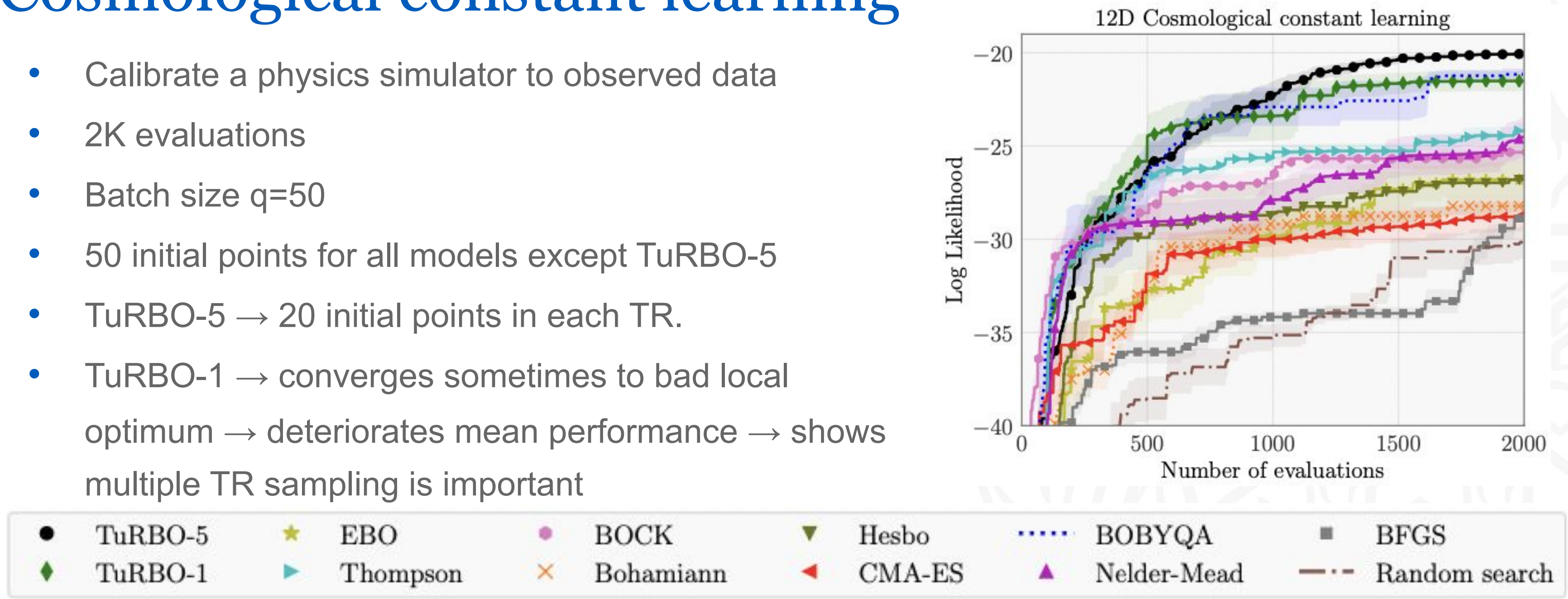
Rover Trajectory planning

- Optimize the locations of 30 points in the 2D-plane that determine the trajectory of a rover.
- 200 steps with a batch size of q-100, total 20K evaluations.
- TuRBO-1 and other models initialized with 200 points
- TuRBO-20 initialized with 100 points for each TR.
- TuRBO-1 and TuRBO-20 achieve close optimal objective values after 10K evaluations.



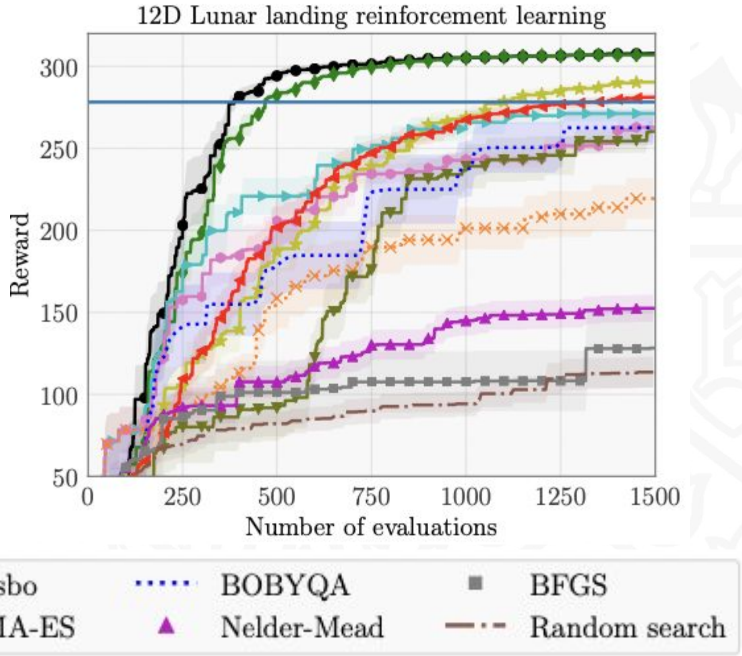
Cosmological constant learning

- Calibrate a physics simulator to observed data
- 2K evaluations
- Batch size q=50
- 50 initial points for all models except TuRBO-5
- TuRBO-5 → 20 initial points in each TR.
- TuRBO-1 → converges sometimes to bad local optimum → deteriorates mean performance → shows multiple TR sampling is important



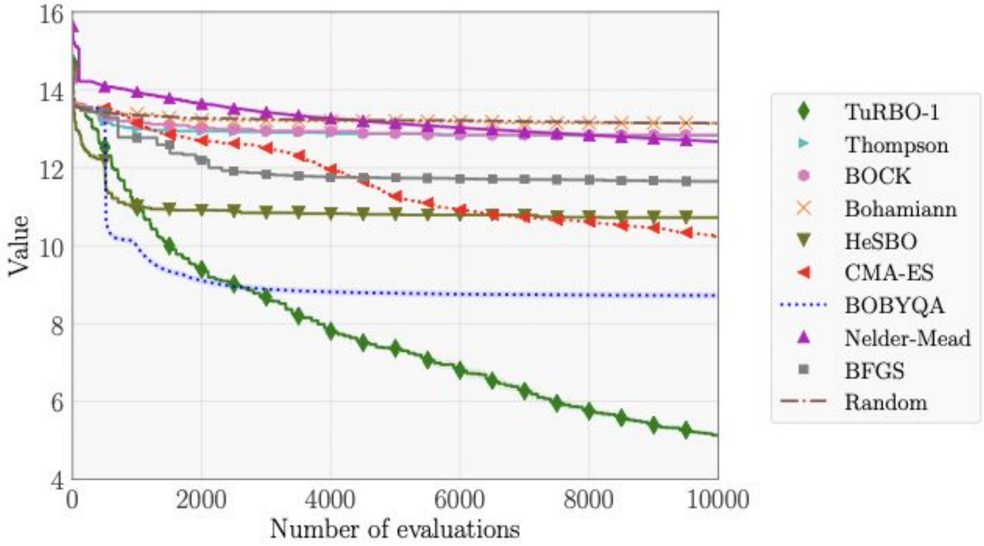
Lunar landing reinforcement learning

- Learn a controller for lunar lander
- Possible actions → left, right, up, nothing
- Maximize average final reward
- 1500 function evaluations, batch size q=50
- 50 initial points for all models except TuRBO-5
- TuRBO-5 → 20 initial points in each TR.
- TuRBO-5 and TuRBO-1 → learn best controllers



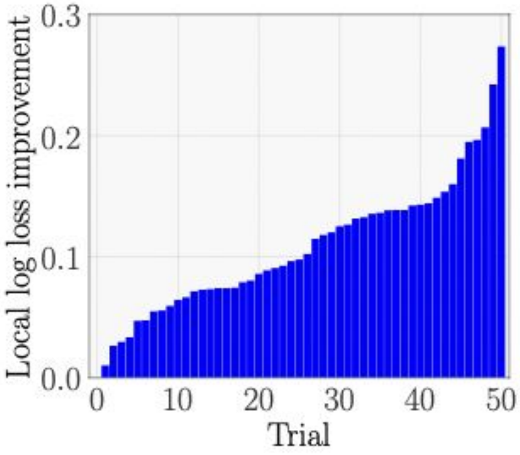
200-dimensional Ackley function

- Domain $[-5, 10]^{200}$
- Has too many local minima
- Total 10K evaluations
- Batch size q=100,
- 200 initial points for all algos
- TuRBO-1 performs well



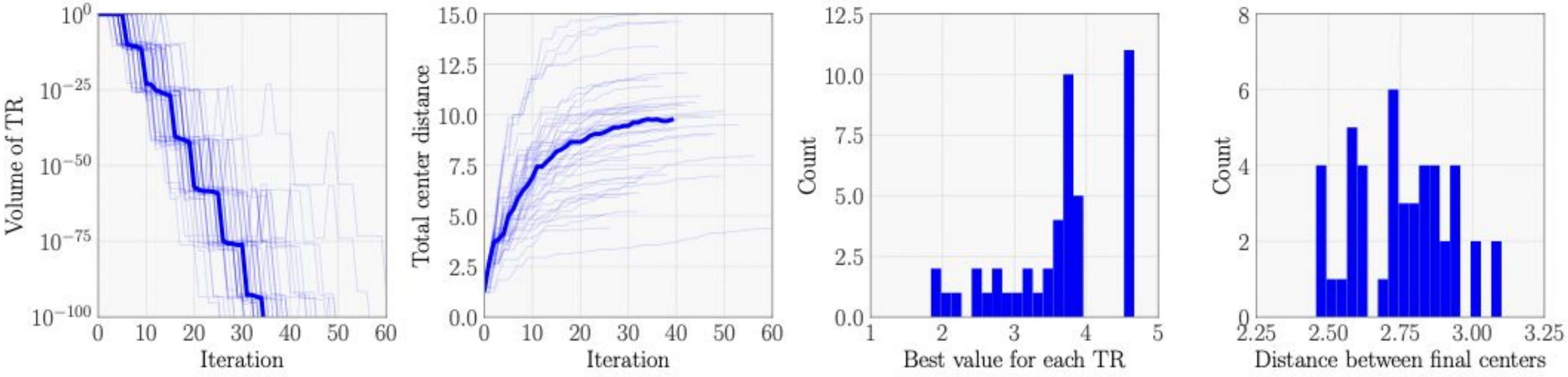
Advantage of local models over global models

- Performance evaluation of local and global GP models on the 14D robot pushing problem
- Global GP → all 4000 points
  Local GP → 20 hypercubes, each of side length 0.4 and 200 uniformly distributed training points.
- Global GP → can access all the data points
  Local GPs → can learn different hyperparameter in each region.
- Global GP → average log loss 1.284
  local model → average log loss 1.174 for 50 trails.
- Proves local approach improves predictive power and reduces computational overhead



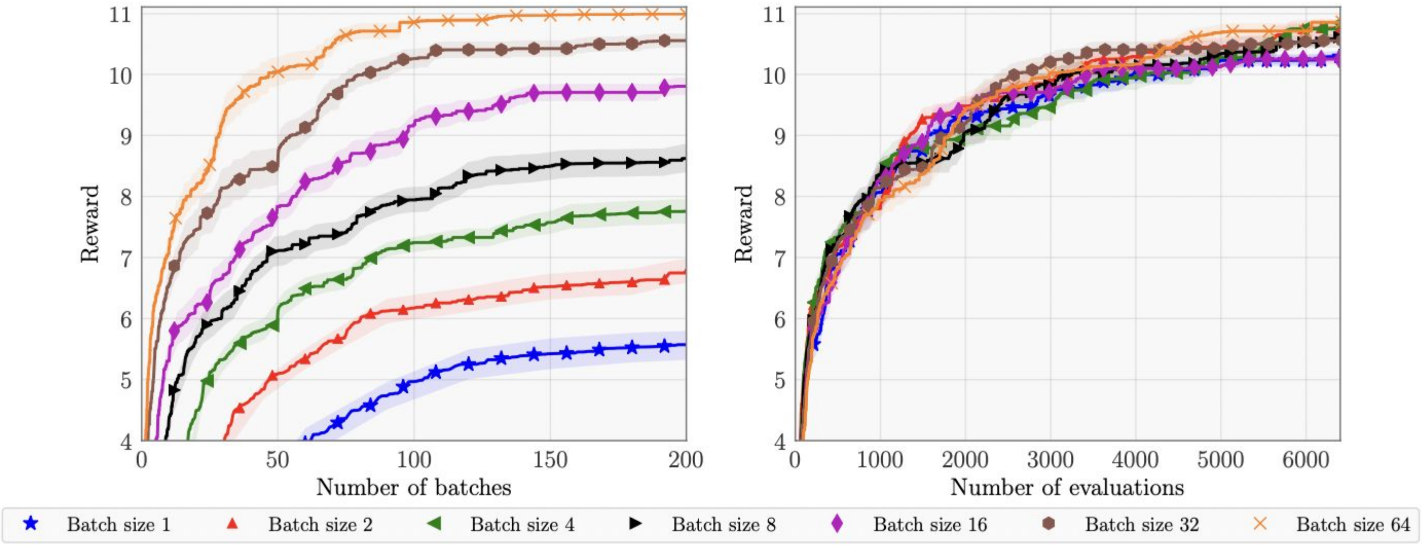
Why high-dimensional spaces are challenging

- TuRBO-1 for 50 restarts on the 60D rover trajectory planning.
- Within a TR, optimization is local, volume of any TR decreases rapidly



Efficiency of large batches

- Large batches obtain better results for the same number of iterations.
- Speed up is nearly linear.



Conclusion

- TuRBO uses novel local approach to global optimization.
- Multiple local models discover better objective values
- TuRBO outperform state-of-the-art techniques on a variety of real-world complex tasks.

References

- https://arxiv.org/pdf/1910.01739.pdf
- https://www.sfu.ca/~ssurjano/branin.html
- https://www.youtube.com/watch?v=Zgwfw3bzSmQ
- https://stats.stackexchange.com/questions/408690/what-is-the-relation-between-a-surrogate-function-and-an-acquisition-function

Thank you!