Outlines

- Parameter/Function Optimization
- Learning to Simulate
  - ICLR2019, https://arxiv.org/abs/1810.02513
- Simulating, Fast and Slow: Learning Policies for Black-Box Optimization
  - 2024, https://arxiv.org/abs/2406.04261
- Black-Box Optimization with Local Generative Surrogates
  - NeurIPS2020, https://proceedings.neurips.cc/paper/2020/hash/a878dbebc902328b41dbf02aa87abb58-Abstract.html

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- We adjust the parameters $\boldsymbol{x}$ *or* $\boldsymbol{\theta}$ to achieve desired output $\boldsymbol{y}$.

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- We adjust the parameters $\boldsymbol{x}$ *or* $\boldsymbol{\theta}$ to achieve desired output $\boldsymbol{y}$.
  - $\boldsymbol{x}$: Input to the function, $\boldsymbol{\theta}$: Mode parameters
    - In some cases, $\boldsymbol{x}$ and $\boldsymbol{\theta}$ may be indistinguishable.

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- We adjust the parameters $\boldsymbol{x}$ *or* $\boldsymbol{\theta}$ to achieve desired output $\boldsymbol{y}$.
  - $\boldsymbol{x}$: Input to the function, $\boldsymbol{\theta}$: Mode parameters
    - In some cases, $\boldsymbol{x}$ and $\boldsymbol{\theta}$ may be indistinguishable.
  - $\boldsymbol{y}$: Function output, $\boldsymbol{t}$: Desired target

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- We adjust the parameters $\boldsymbol{x}$ *or* $\boldsymbol{\theta}$ to achieve desired output $\boldsymbol{y}$.
  - $\boldsymbol{x}$: Input to the function, $\boldsymbol{\theta}$: Mode parameters
    - In some cases, $\boldsymbol{x}$ and $\boldsymbol{\theta}$ may be indistinguishable.
  - $\boldsymbol{y}$: Function output, $\boldsymbol{t}$: Desired target
  - L: Loss or objective function
    - Measures the difference between $\boldsymbol{y}$ and $\boldsymbol{t}$

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- Parameter Optimization: Focuses on optimizing $\boldsymbol{\theta}$ to achieve the desired $\boldsymbol{y}$ for a given $\boldsymbol{x}$.
  - Classification: For $\boldsymbol{x}$ =  and $\boldsymbol{t}$ = apple, adjust $\boldsymbol{\theta}$ so that $\boldsymbol{y}$ = apple.

Apple made by Vectors Market, www.flaticon.com

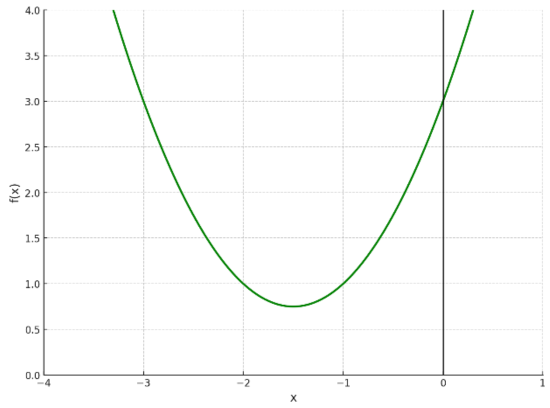
Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- Function Optimization: Focuses on finding $\boldsymbol{x}$ such that $\boldsymbol{y}$ is minimized or maximized.
  - Typical optimization problems.
  - $\boldsymbol{t}$ is implicitly given as $\pm\infty$.

$f(x) = x^2 + 3x + 3$



Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- In our case,
  - $\boldsymbol{f}$: A traffic simulator
  - $\boldsymbol{y}$: Simulation output
  - $\boldsymbol{t}$: Real traffic observation
  - We need to define $\boldsymbol{x}$, $\boldsymbol{\theta}$ and $L(\boldsymbol{y},\boldsymbol{t})$.

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- Technical Challenge
  - Difficulties arise when $\boldsymbol{f}$ is a black model and non-differentiable with respect to $\boldsymbol{x}$ or $\boldsymbol{\theta}$.

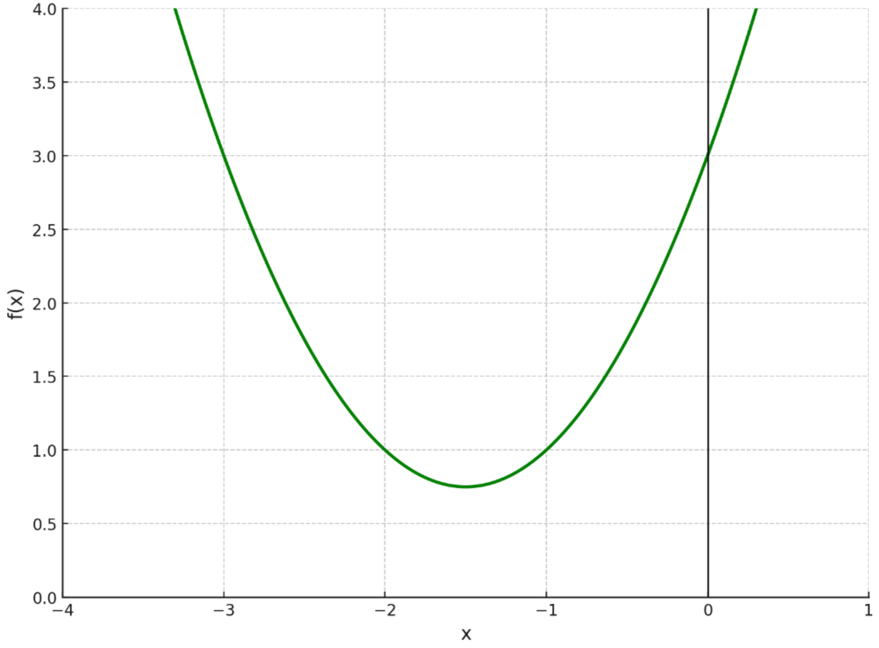
- Minimization of a function.

$$f(x) = x^2 + 3x + 3$$

$$\frac{\partial f}{\partial x} = 2x + 3 = 0$$

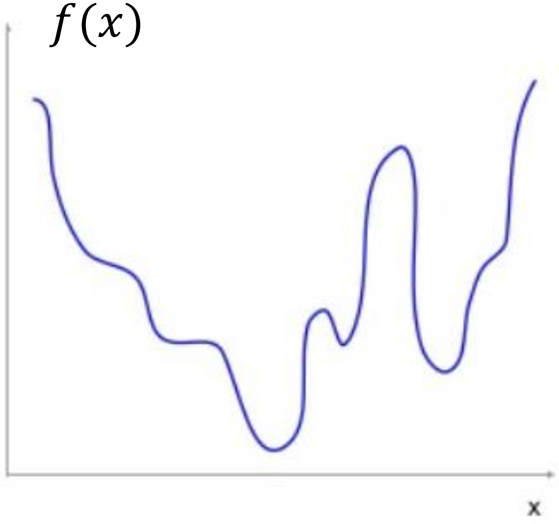
$$x = -\frac{3}{2}$$

$$f(x) = x^2 + 3x + 3$$



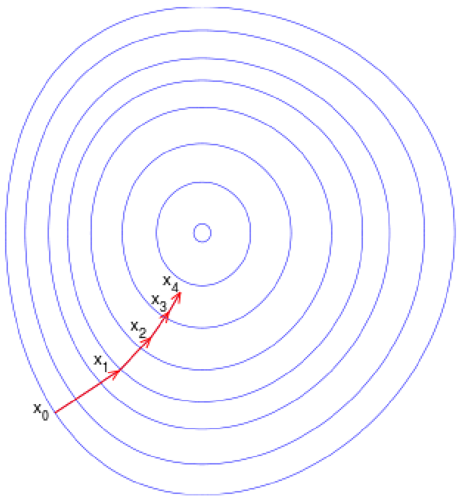
Gradient Descent/Ascent

$f(x)$



x

$$x_{n+1} = x_n - \lambda \left.\frac{\partial f}{\partial x}\right|_{x_n}$$



https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-optimization-upc-2016

https://en.wikipedia.org/wiki/Gradient_descent

Parameter/Function Optimization

$$\boldsymbol{x} \longrightarrow \boxed{\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})} \longrightarrow \boldsymbol{y} \longrightarrow L(\boldsymbol{y},\boldsymbol{t})$$

Ex. $y = f(x;\boldsymbol{\theta}) = ax^2 + bx + c$

- There are several approaches to optimize a non-differentiable black box function.
  - Policy Gradient/REINFORCE
  - Bayesian Optimization
  - Local Surrogate-based Approach
  - Genetic Algorithm
  - Particle Swam Optimization

Outlines

- Parameter/Function Optimization
- Learning to Simulate
  - ICLR2019, https://arxiv.org/abs/1810.02513
- Simulating, Fast and Slow: Learning Policies for Black-Box Optimization
  - 2024, https://arxiv.org/abs/2406.04261
- Black-Box Optimization with Local Generative Surrogates
  - NeurIPS2020, https://proceedings.neurips.cc/paper/2020/hash/a878dbebc902328b41dbf02aa87abb58-Abstract.html

Learning to Simulate

- ICLR2019, https://arxiv.org/abs/1810.02513

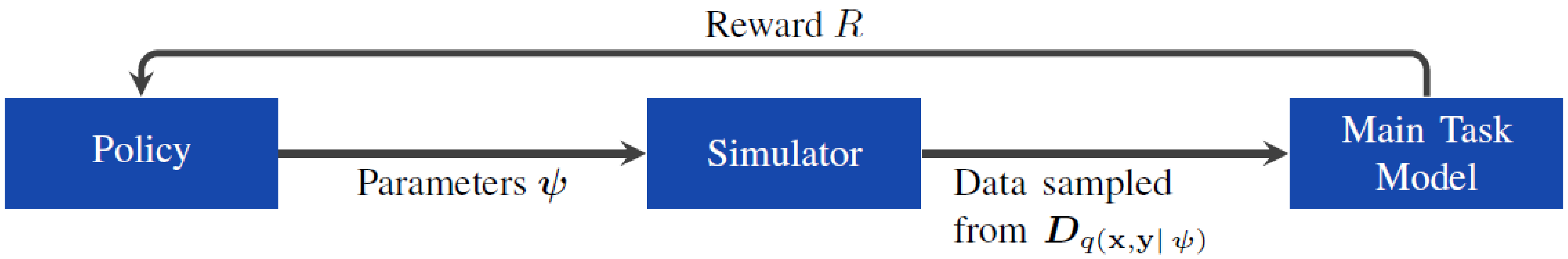


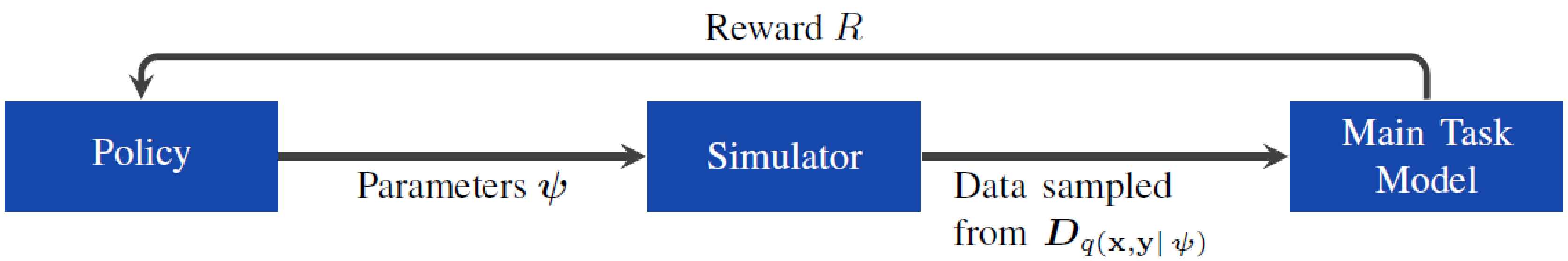
Figure 1: A high-level overview of our "learning to simulate" approach. A policy π_ω outputs parameters ψ which are used by a simulator to generate a training dataset. The main task model (MTM) is then trained on this dataset and evaluated on a validation set. The obtained accuracy serves as reward signal R for the policy on how good the synthesized dataset was. The policy thus learns *how to generate data to maximize the validation accuracy.*

- Simulator: Traffic Scene Generation
  - Car Counting, Segmentation

Figure 3: Example of rendered traffic scene with CARLA (Dosovitskiy et al., 2017) and the Unreal engine (Epic-Games, 2018).

- A straight road of variable length.
- Either an L, T or X intersection at the end of the road.
- Cars of 5 different types which are spawned randomly on the straight road.
- Houses of a unique type which are spawned randomly on the sides of the road.
- Four different types of weather.

Learning to Simulate



Policy: Determines the simulation setting parameters ψ.

Simulator: Generate a dataset based on parameters ψ.

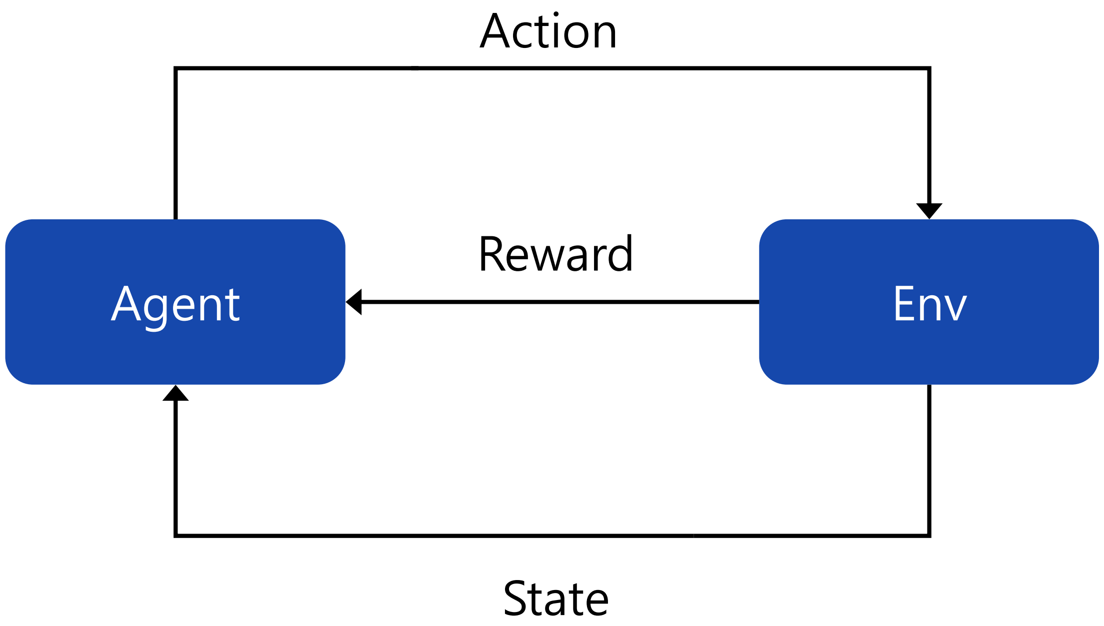
Main Task Model: Trains on the data generated from the simulator for task like:

Classification, Regression, Segmentation

Reward ψ : The feedback signal is based on the performance of the **Main Task** model (Accuracy/Error).

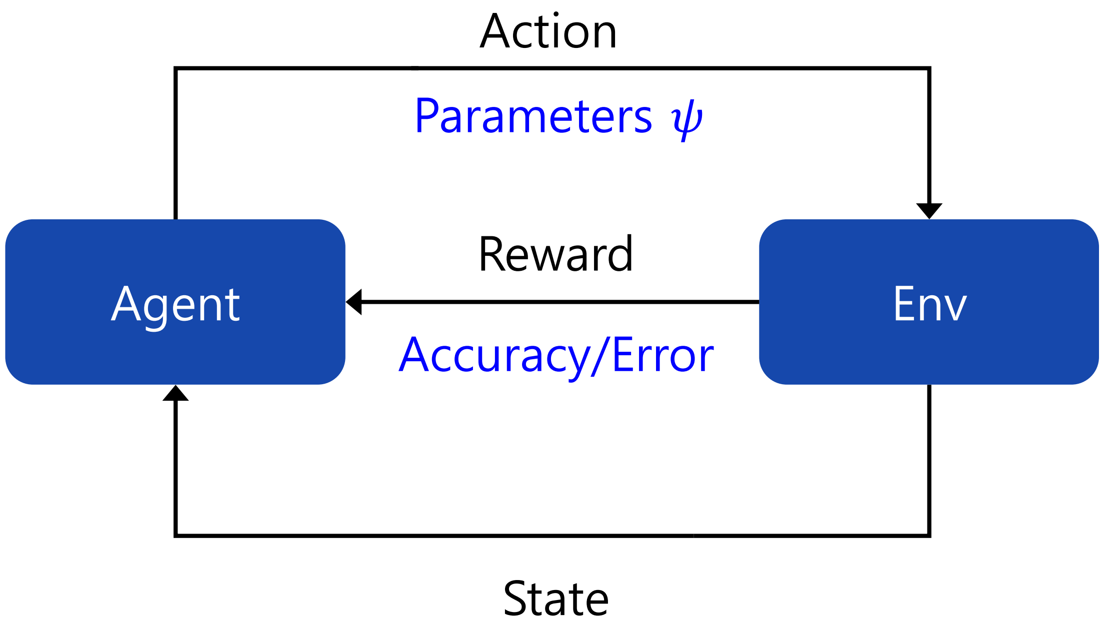
Learning to Simulate

- RL-based Approach



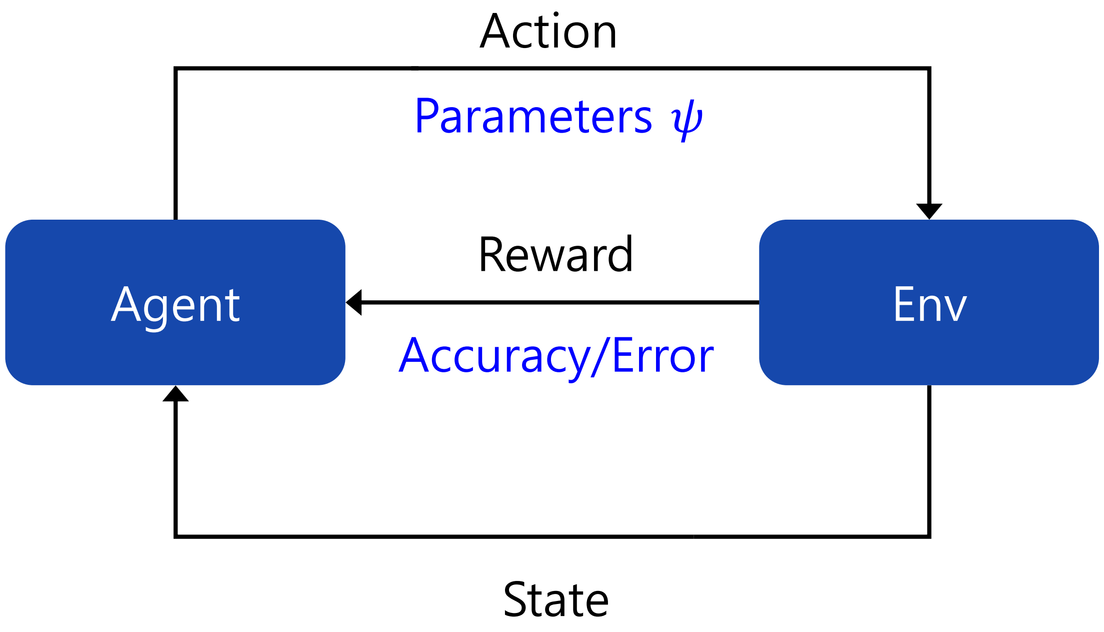
Learning to Simulate

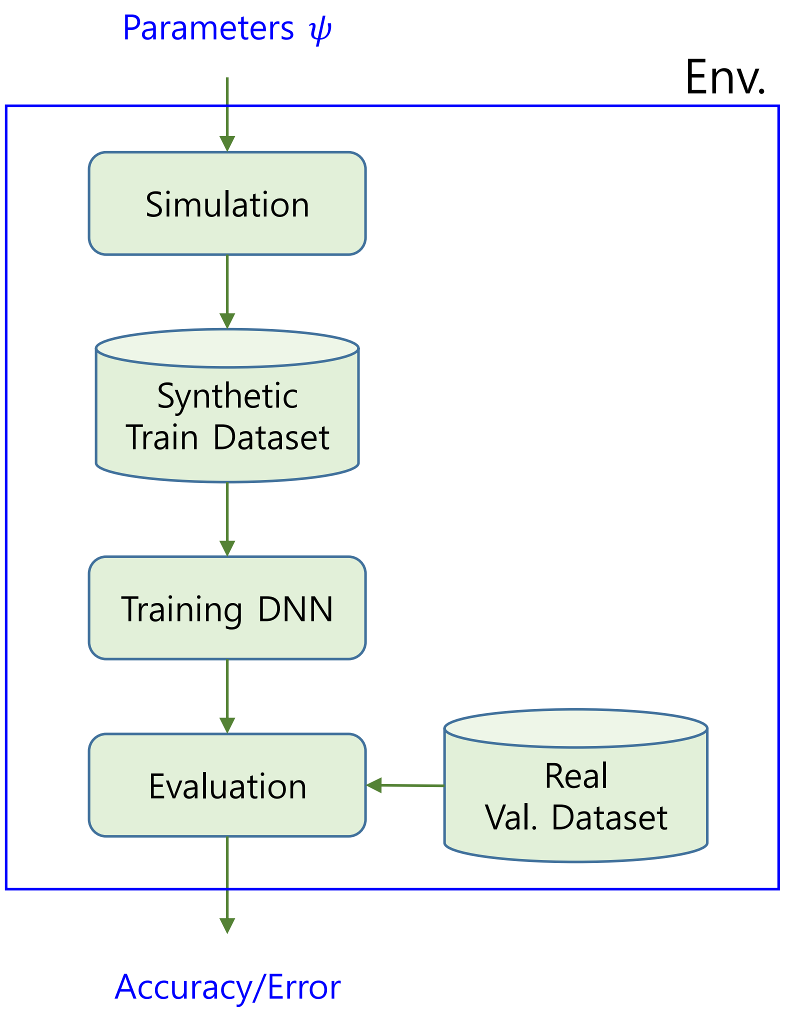
- RL-based Approach



Learning to Simulate

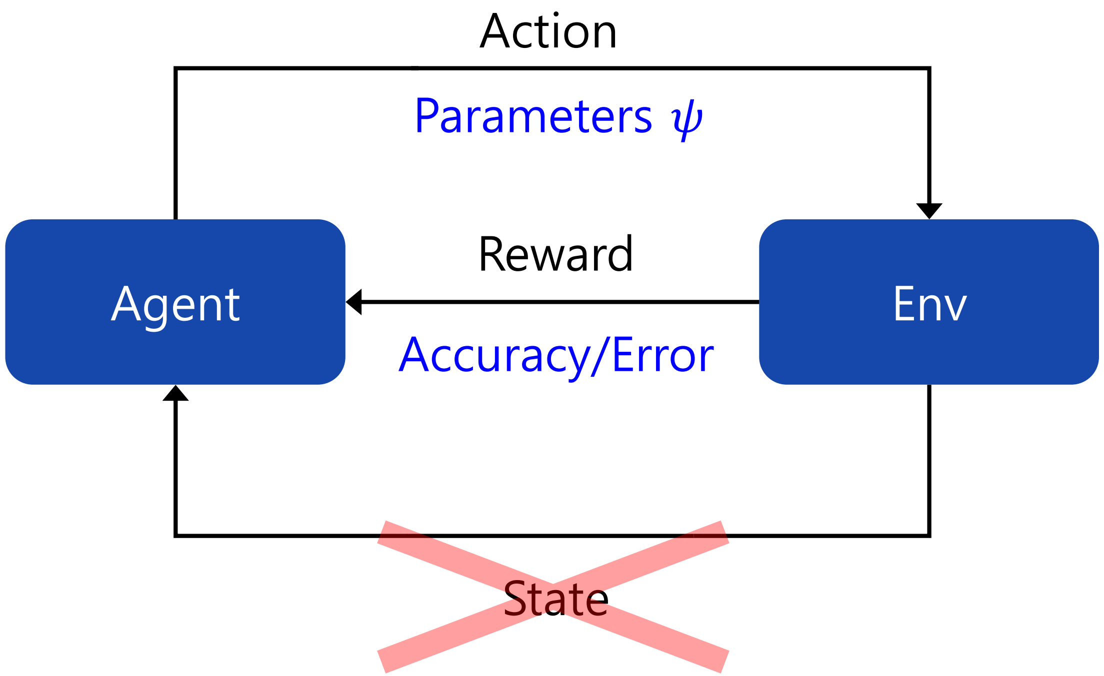
- RL-based Approach

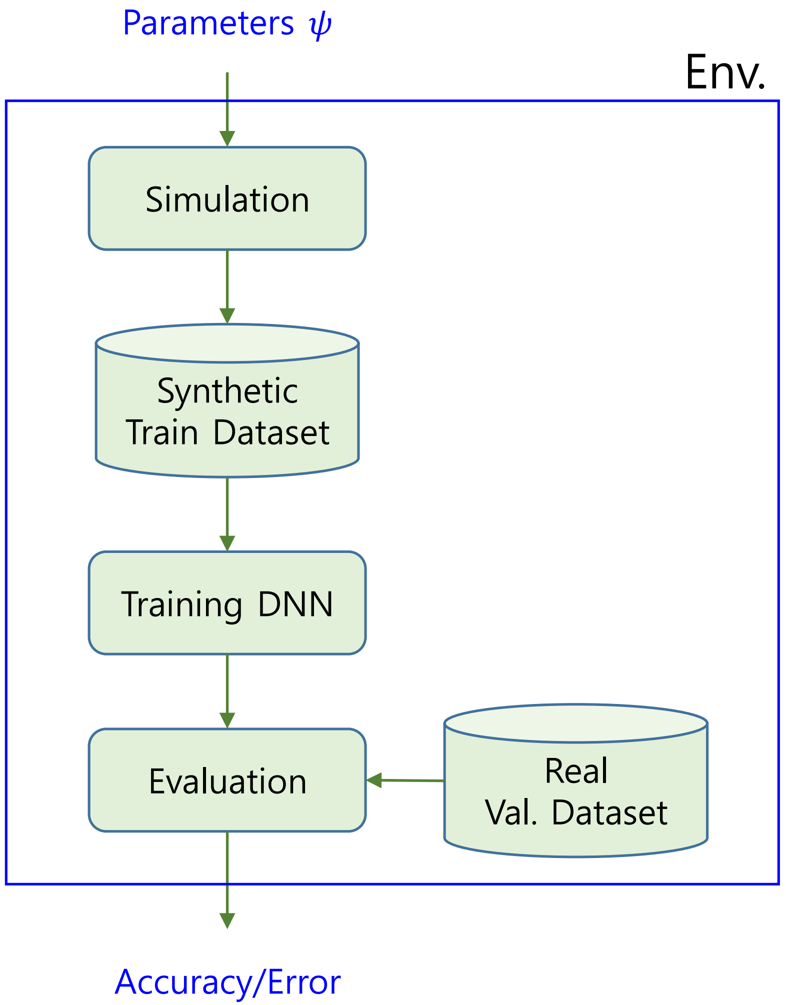




Learning to Simulate

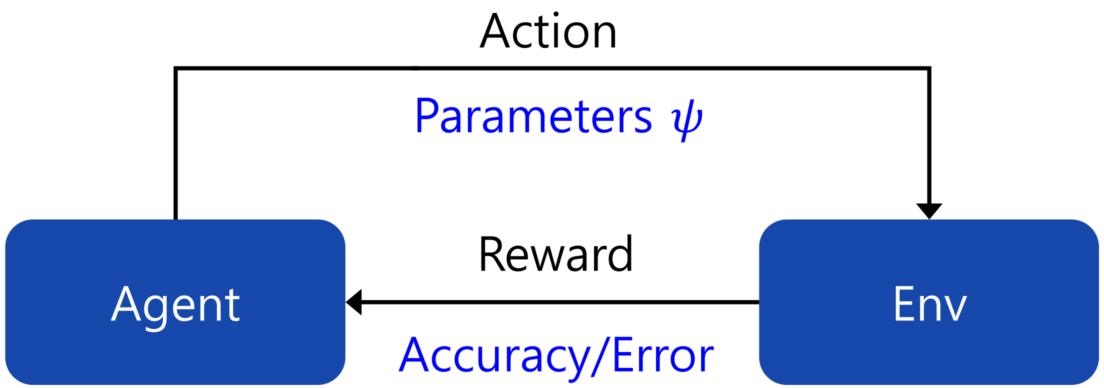
- RL-based Approach

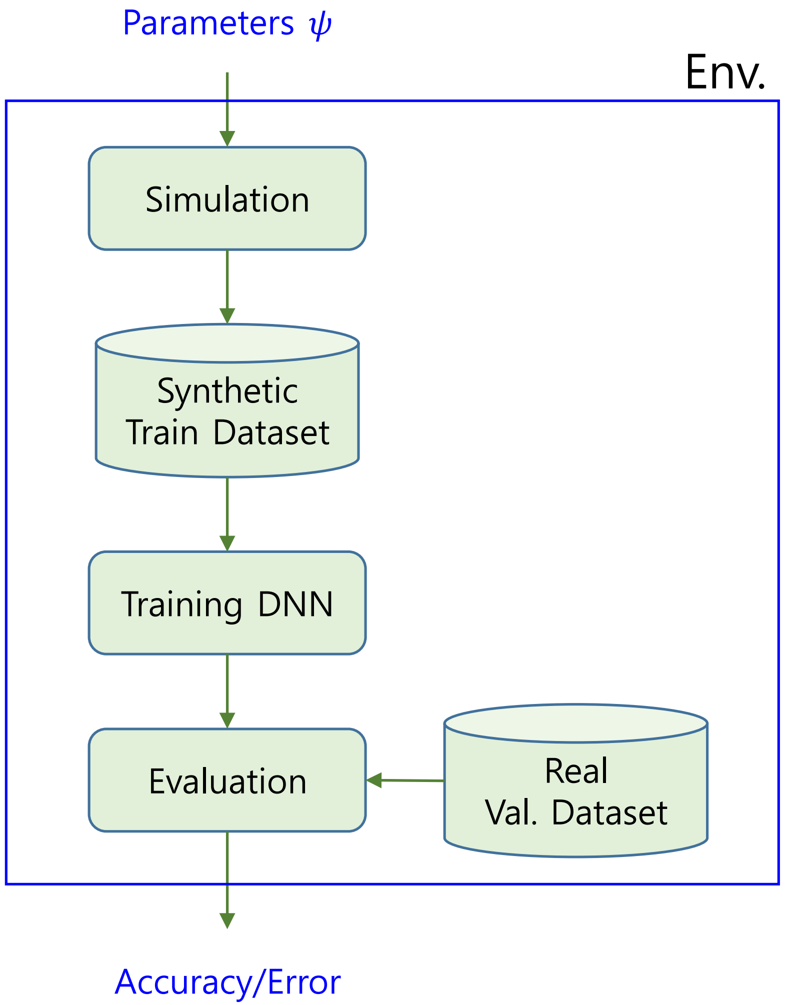


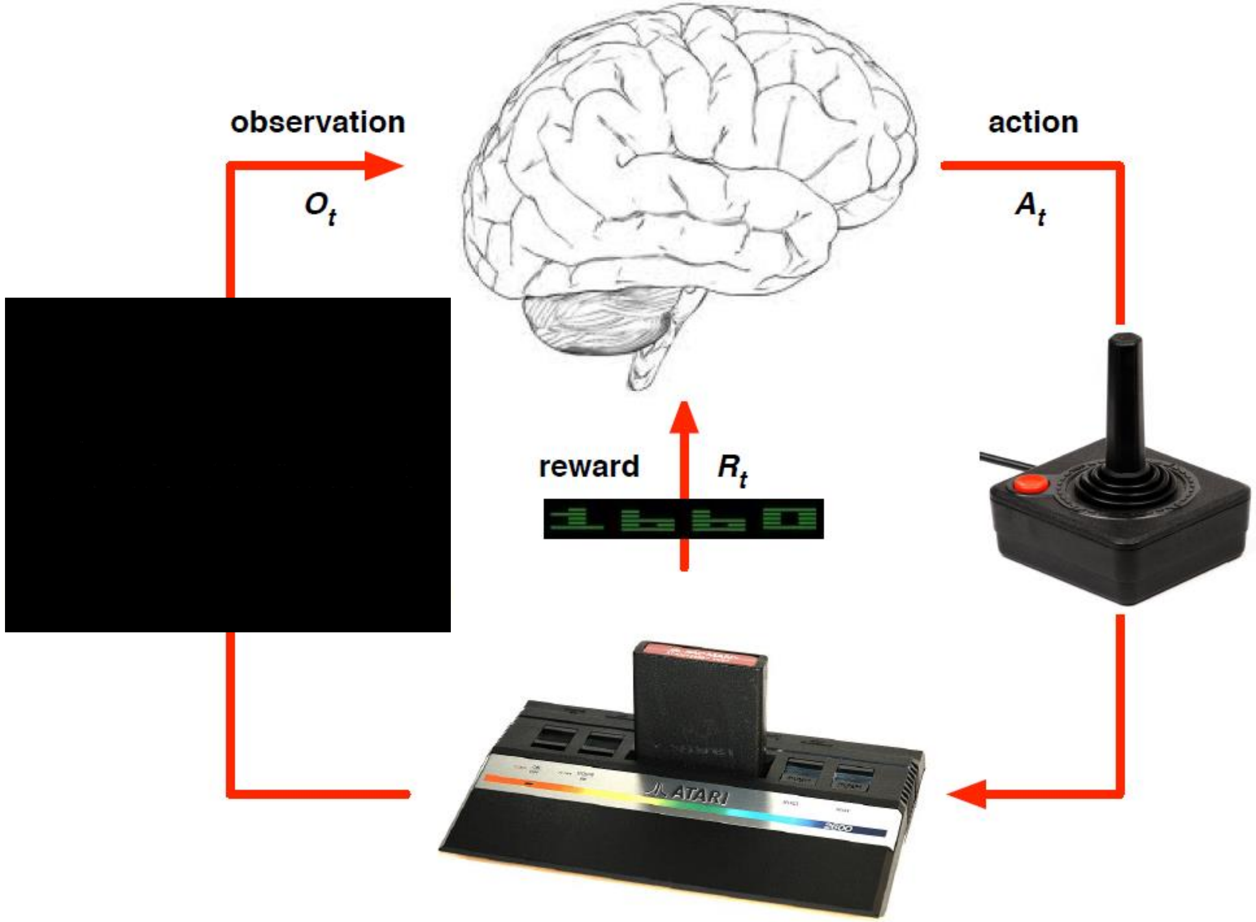


Learning to Simulate

- RL-based Approach

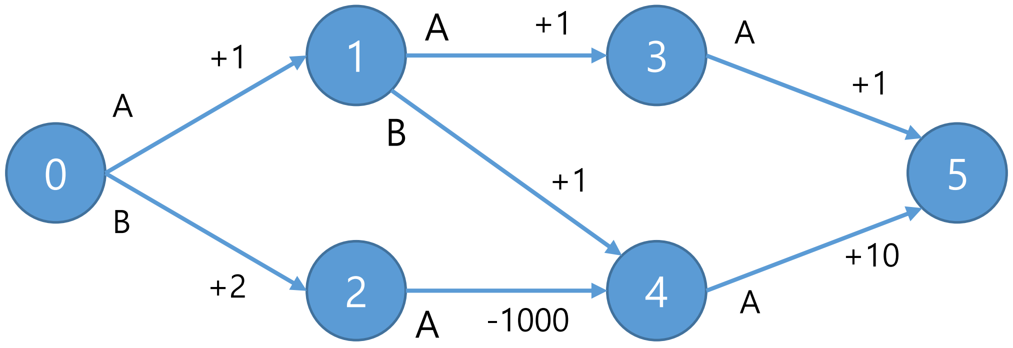





observation
O_t
action
A_t
reward
R_t
ATARI

Sequential Decision Problem

- Generalizes the previous example.
  - State, Action, Reward.
  - Maximize the sum of rewards.
  - Each decision affects subsequent decisions.



Slide modified from Reinforcement Learning: A User's Guide, Bill Smart, Washington University in St. Louis

Learning to Simulate

- Reinforcement Learning (RL) is overkill for this application.
  - Non-sequential problem
    - The simulation optimization problem is **not a sequential decision-making task**.
    - Since the problem doesn't involve sequential decisions, there's no strong reason to use RL.
  - Challenges with RL
    - **Complex training**: Training an RL agent requires significant care to ensure convergence, avoid overfitting, and balance exploration-exploitation.
    - **High computational cost**: RL training often involves extensive simulation calls, making it time-consuming and expensive.
    - **On-policy constraints**: Practical and popular methods like **PPO** require new simulation samples at each iteration, further increasing cost and complexity.

Outlines

- Parameter/Function Optimization
- Learning to Simulate
  - ICLR2019, https://arxiv.org/abs/1810.02513
- Simulating, Fast and Slow: Learning Policies for Black-Box Optimization
  - 2024, https://arxiv.org/abs/2406.04261
- Black-Box Optimization with Local Generative Surrogates
  - NeurIPS2020, https://proceedings.neurips.cc/paper/2020/hash/a878dbebc902328b41dbf02aa87abb58-Abstract.html

Simulating, Fast and Slow: Learning Policies for Black-Box Optimization

- 2024, https://arxiv.org/abs/2406.04261
  - Surrogate-based Approach

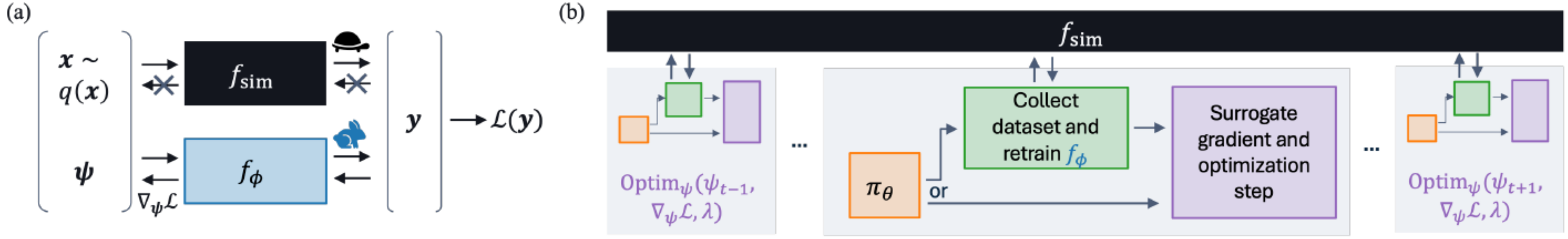
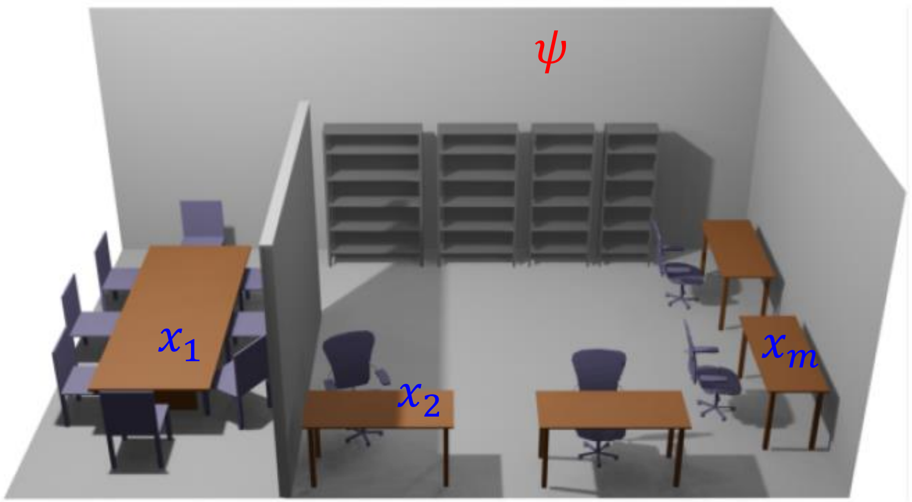


Figure 1: Schematic view of our approach. (a) We study black-box optimization problem (over parameters ψ), with an emphasis on using gradient information from a fast differentiable surrogate f_ϕ (b) To optimize ψ sample-efficiently, we employ a policy π_θ to actively determine whether retraining the surrogate is necessary before using the gradient information.

- Wireless Communication: Indoor Transmitting Antenna Placement

Find the transmitting antenna location (ψ)
to maximize signal strength (y) at receiver locations (x).

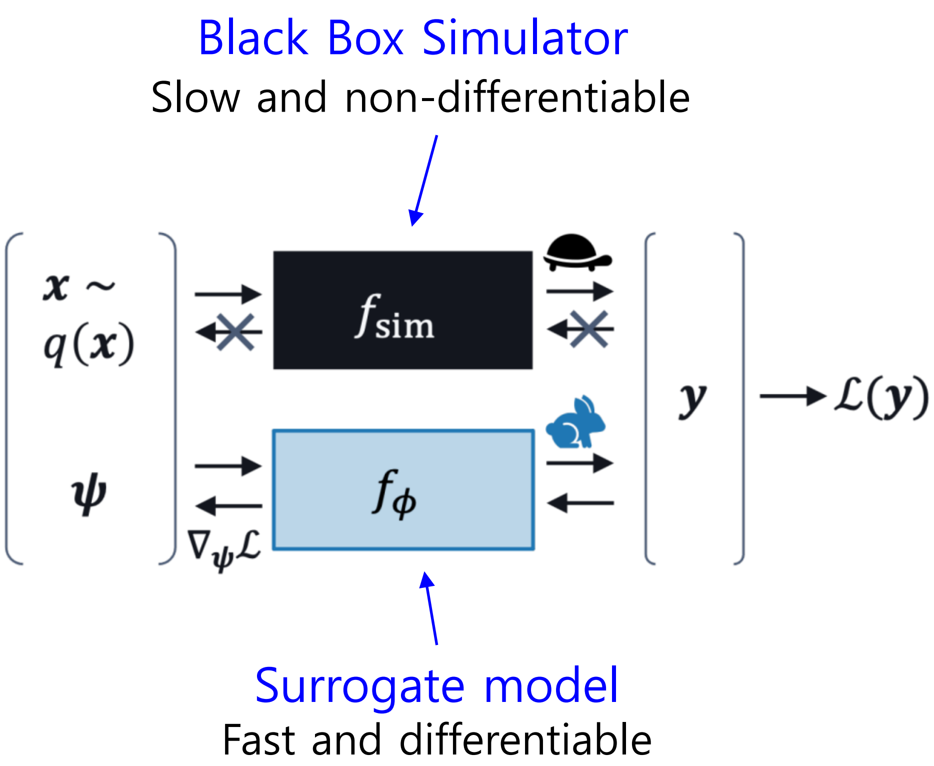


ψ: Transmitting antenna location.

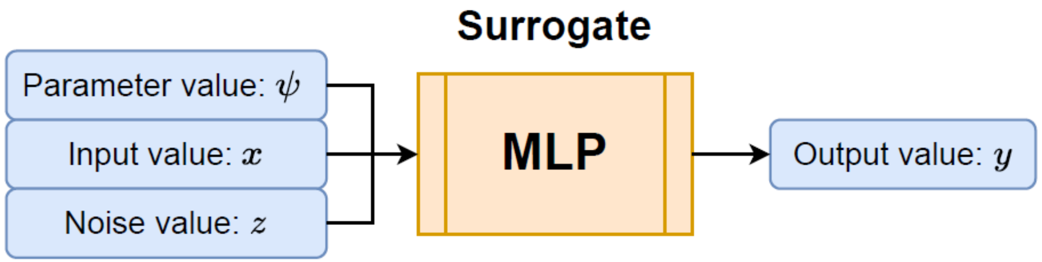
x: Receiving antenna locations.

y: Signal strength at receiver's locations.

Surrogate Model

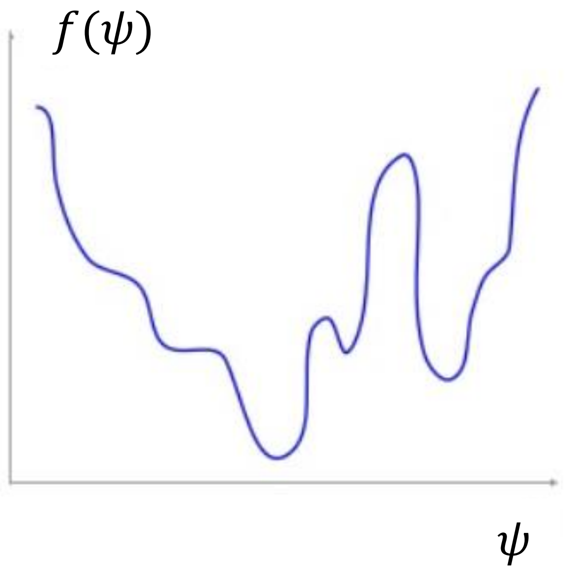


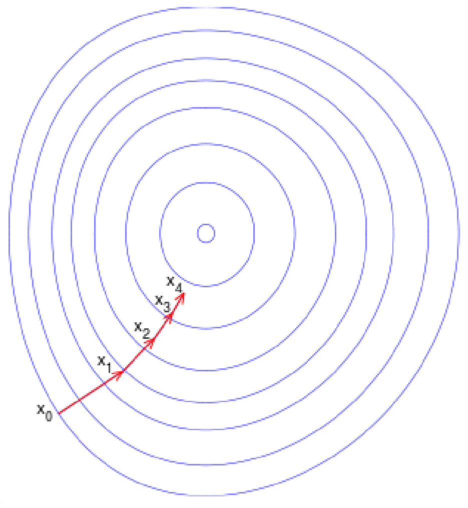
- Train a DNN surrogate to approximate the simulator.
- Then, gradients from the surrogate are used for optimizing ψ.



Gradient Descent/Ascent

$$\psi_{n+1} = \psi_n - \lambda \left.\frac{\partial f}{\partial \psi}\right|_{\psi_n}$$





https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-optimization-upc-2016

https://en.wikipedia.org/wiki/Gradient_descent

Learning Policy to train a surrogate model

- Goal: Minimize # simulation calls
  - When to call a simulation
  - How to sample training data.

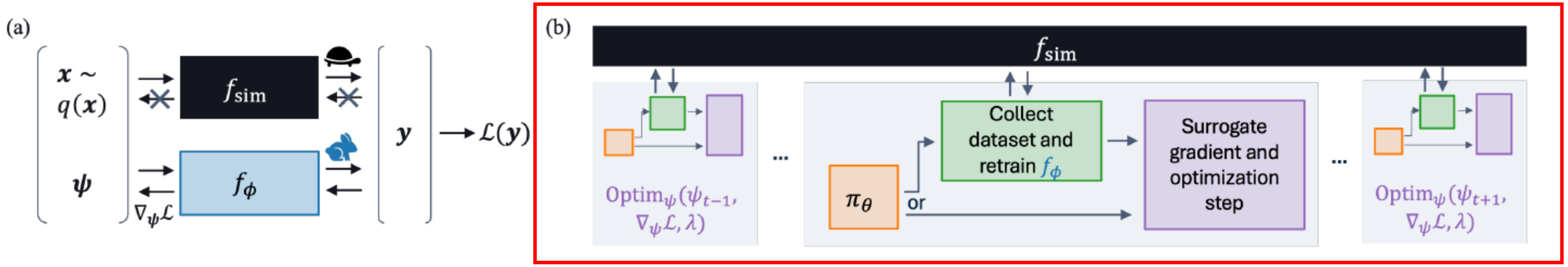
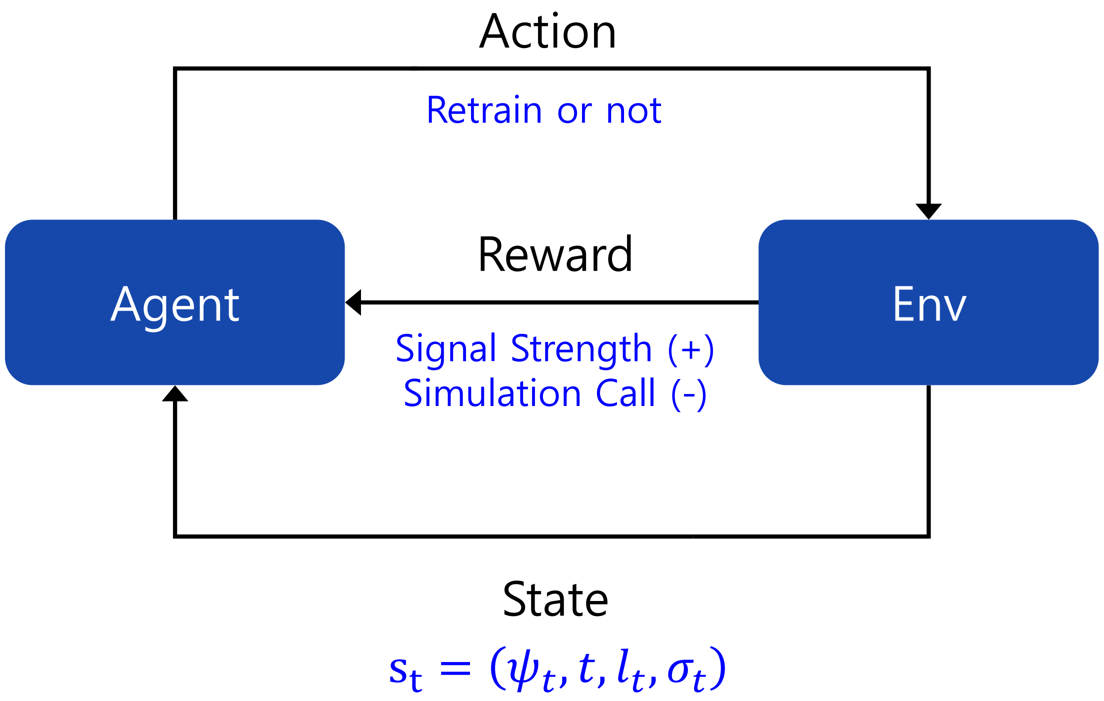


Figure 1: Schematic view of our approach. (a) We study black-box optimization problem (over parameters ψ), with an emphasis on using gradient information from a fast differentiable surrogate f_ϕ (b) To optimize ψ sample-efficiently, we employ a policy π_θ to actively determine whether retraining the surrogate is necessary before using the gradient information.

Learning Policy to train a surrogate model

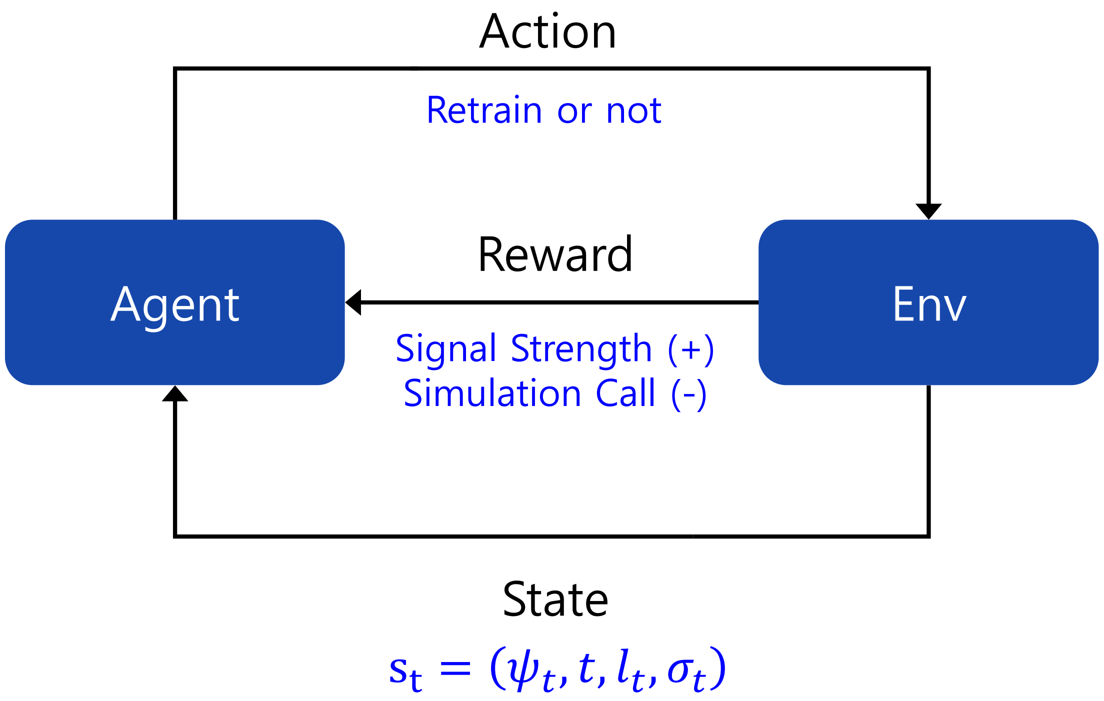


State

- ψ_t: The current parameter values being optimized.
- t: The current timestep in the optimization process.
- l_t: The number of simulator calls that have already been made during the current episode.
- σ_t: A measure of the uncertainty of the surrogate model, indicating how reliable its predictions are at the current ψ_t.

Note: Agent ≠ Surrogate

Learning Policy to train a surrogate model



Action: Retrain

- 1. Determine the sampling boundary for ψ_t.
- 2. Run simulations
- 3. Collect training data
- 4. Retrain a surrogate model

Note: Agent ≠ Surrogate

Simulating, Fast and Slow: Learning Policies for Black-Box Optimization

- ~~Reinforcement Learning (RL) is overkill for this application~~.
  - ~~Non-sequential problem~~
    - ~~The simulation optimization problem is **not a sequential decision-making task**~~.
    - ~~Since the problem doesn't involve sequential decisions, there's no strong reason to use RL~~.
  - Challenges with RL
    - **Complex training**: Training an RL agent requires significant care to ensure convergence, avoid overfitting, and balance exploration-exploitation.
    - ~~**High computational cost**: RL training often involves extensive simulation calls, making it time-consuming and expensive.~~
    - **On-policy constraints**: Practical and popular methods like **PPO** require new simulation samples at each iteration, further increasing cost and complexity.

Outlines

- Parameter/Function Optimization

- Learning to Simulate
    - ICLR2019, https://arxiv.org/abs/1810.02513

- Simulating, Fast and Slow: Learning Policies for Black-Box Optimization
    - 2024, https://arxiv.org/abs/2406.04261

- Black-Box Optimization with Local Generative Surrogates
    - NeurIPS2020, https://proceedings.neurips.cc/paper/2020/hash/a878dbebc902328b41dbf02aa87abb58-Abstract.html

Local Generative Surrogate Optimization

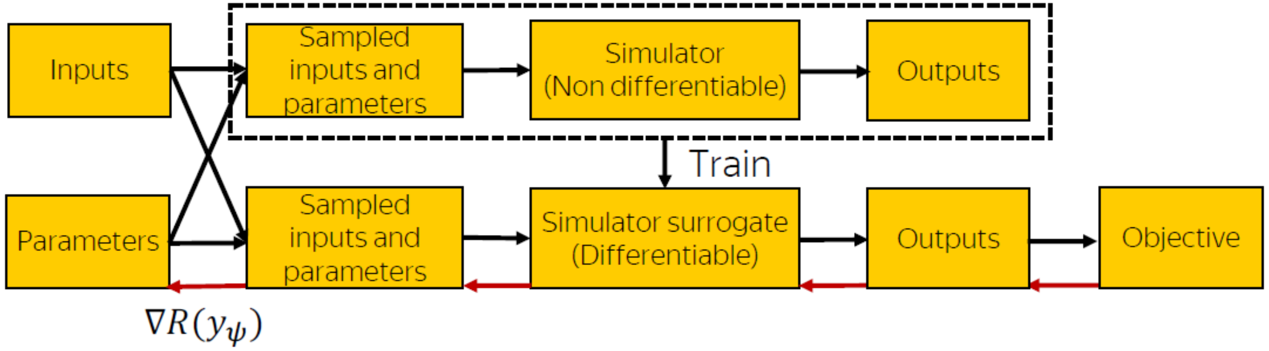
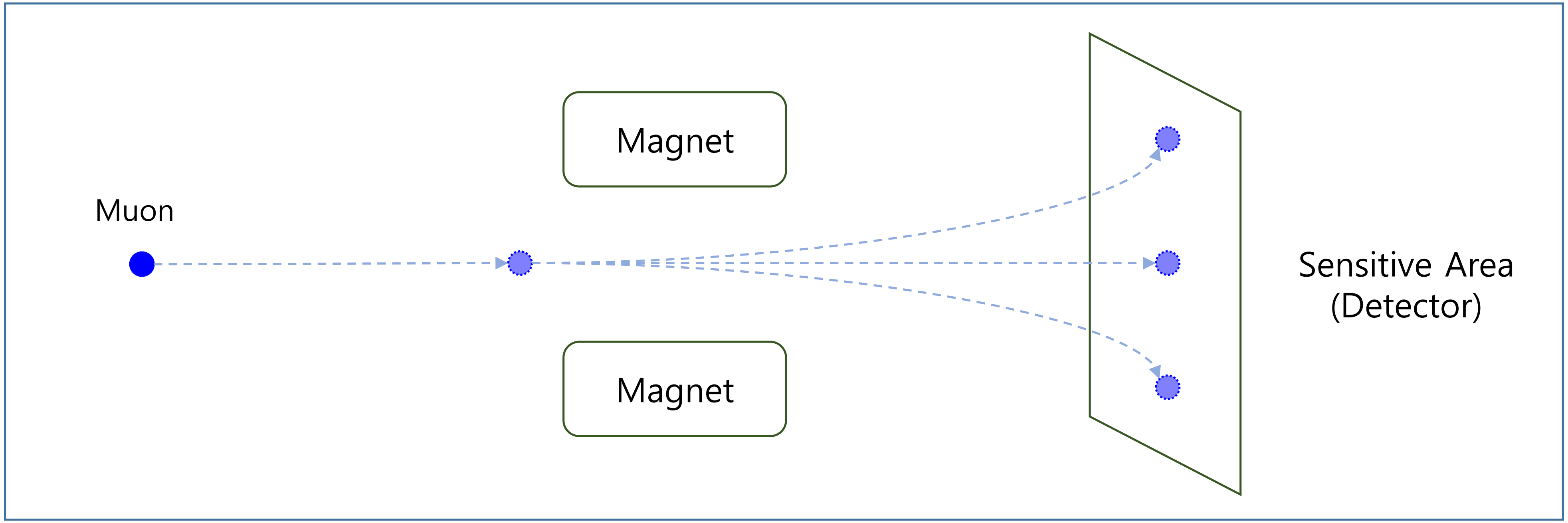


Figure 1: Simulation and surrogate training. *Black:* forward propagation. *Red:* error backpropagation.

Muon Background Reduction

Minimize # muon hits on sensitive area

$$\boldsymbol{y} = F(\boldsymbol{x}; \boldsymbol{\psi})$$



Momentum (P)
Azimuthal (ϕ)
Polar (θ)
Charge (Q)
Coordinate (C)

$\boldsymbol{x} \in R^7$

$\boldsymbol{\psi} \in R^{42}$
Geometries of 6 Magnets

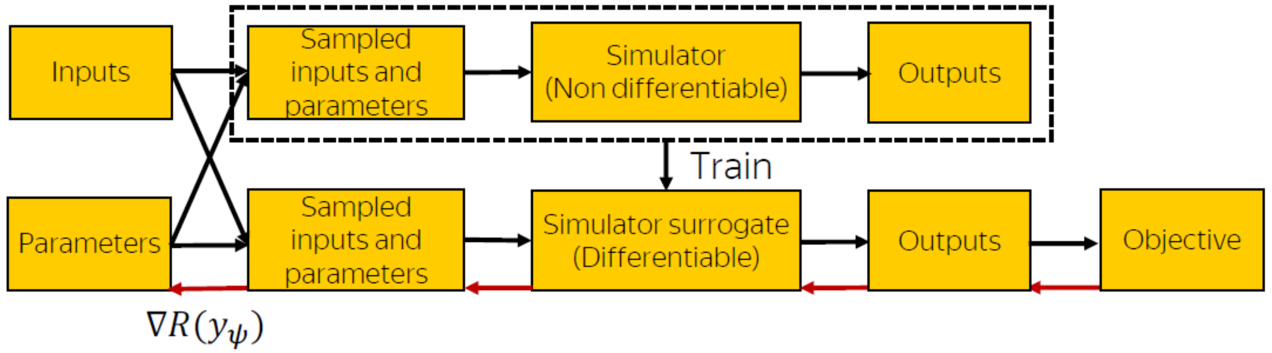
$\boldsymbol{y} \in R^2$
Hit Coordinate in Sensitive Area

Local Generative Surrogate Optimization

Momentum (P)
Azimuthal (ϕ)
Polar (θ)
Charge (Q)
Coordinate (C)

$\boldsymbol{x} \in R^7$

$\boldsymbol{y} \in R^2$
Hit Coordinate in Sensitive Area



\# muon hits on sensitive area

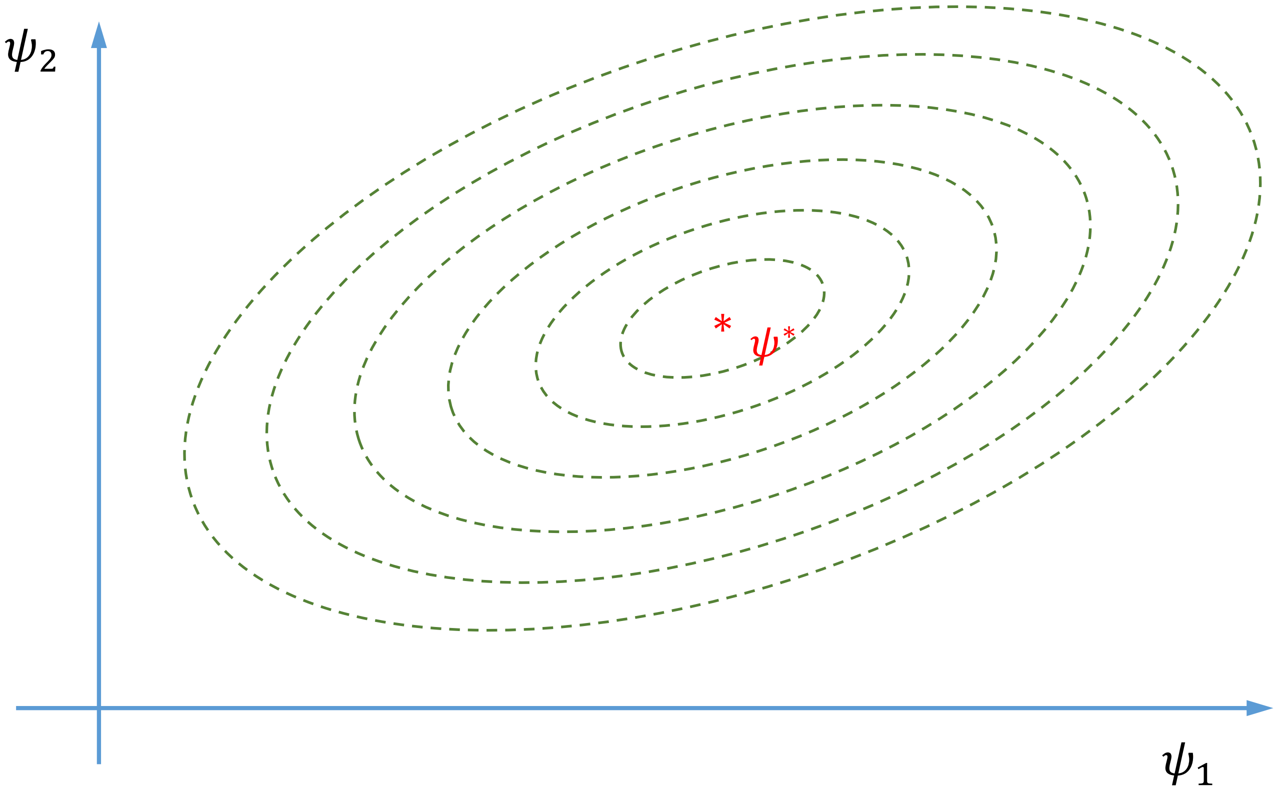
$\boldsymbol{\psi} \in R^{42}$
Geometries of 6 Magnets

Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of x for surrogate training, number K of x for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter ψ
2: **while** ψ has not converged **do**
3: Sample ψ'_i in the region U_ϵ^ψ, $i = 1, \ldots, N$
4: For each ψ'_i, sample inputs $\{x_j^i\}_{j=1}^M \sim q(x)$
5: Sample $M \times N$ training examples from simulator $y_{ij} = F(x_j^i; \psi'_i)$
6: Store y_{ij}, x_j^i, ψ'_i in history H $i = 1, \ldots, N; j = 1, \ldots, M$
7: Extract all y_l, x_l, ψ'_l from history H, iff $d(\psi, \psi'_l) < \epsilon$
8: Train generative surrogate model $S_\theta(z_l, x_l; \psi'_l)$, where $z_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{y}_k = S_\theta(z_k, x_k; \psi)$, $z_k \sim \mathcal{N}(0, 1)$, $x_k \sim q(x)$, $k = 1, \ldots, K$
11: $\nabla_\psi \mathbb{E}[\mathcal{R}(\bar{y})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{y}_k} \frac{\partial S_\theta(z_k, x_k; \psi)}{\partial \psi}$
12: $\psi \leftarrow \mathrm{SGD}(\psi, \nabla_\psi \mathbb{E}[\mathcal{R}(\bar{y})])$
13: **end while**

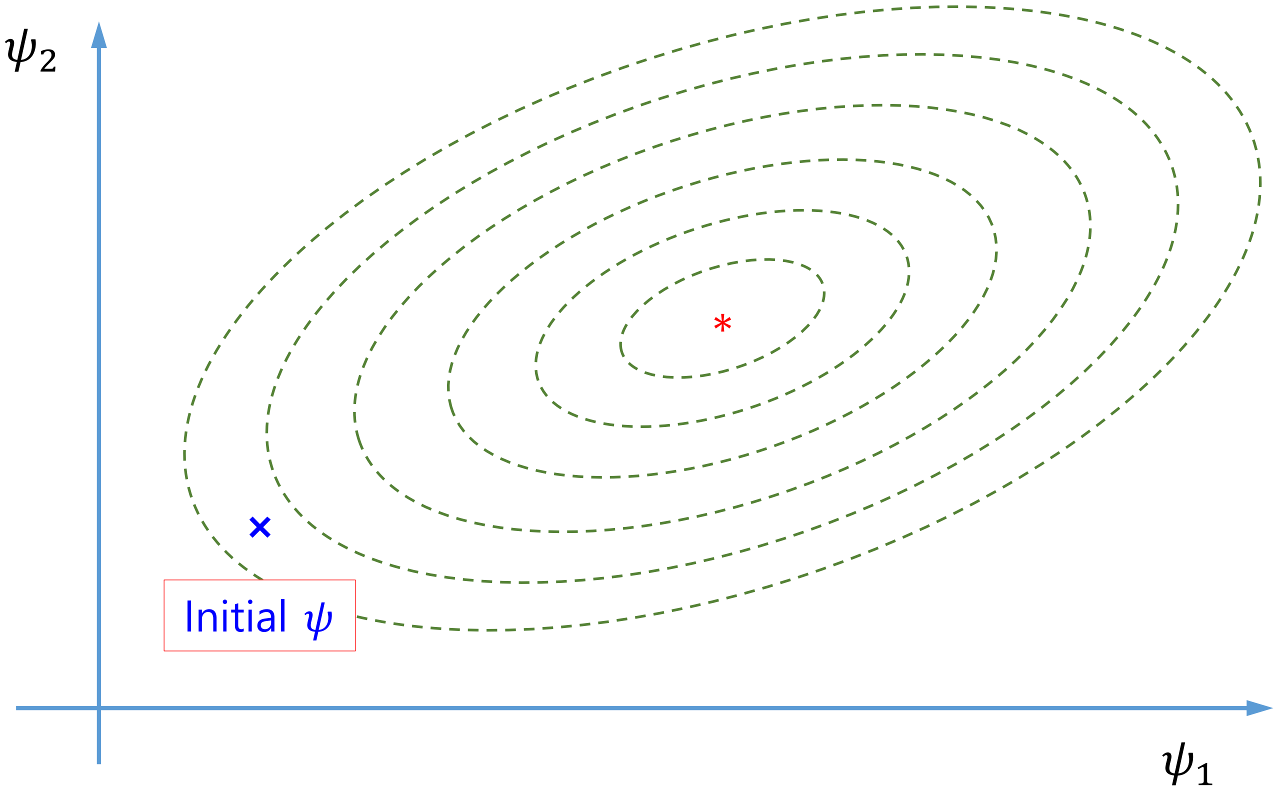


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter ψ
while ψ has not converged **do**
  Sample $\boldsymbol{\psi}'_i$ in the region U^{ψ}_{ϵ}, $i = 1, \ldots, N$
  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
  Fix weights of the surrogate model θ
  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})$, $\boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \ldots, K$
  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
end while

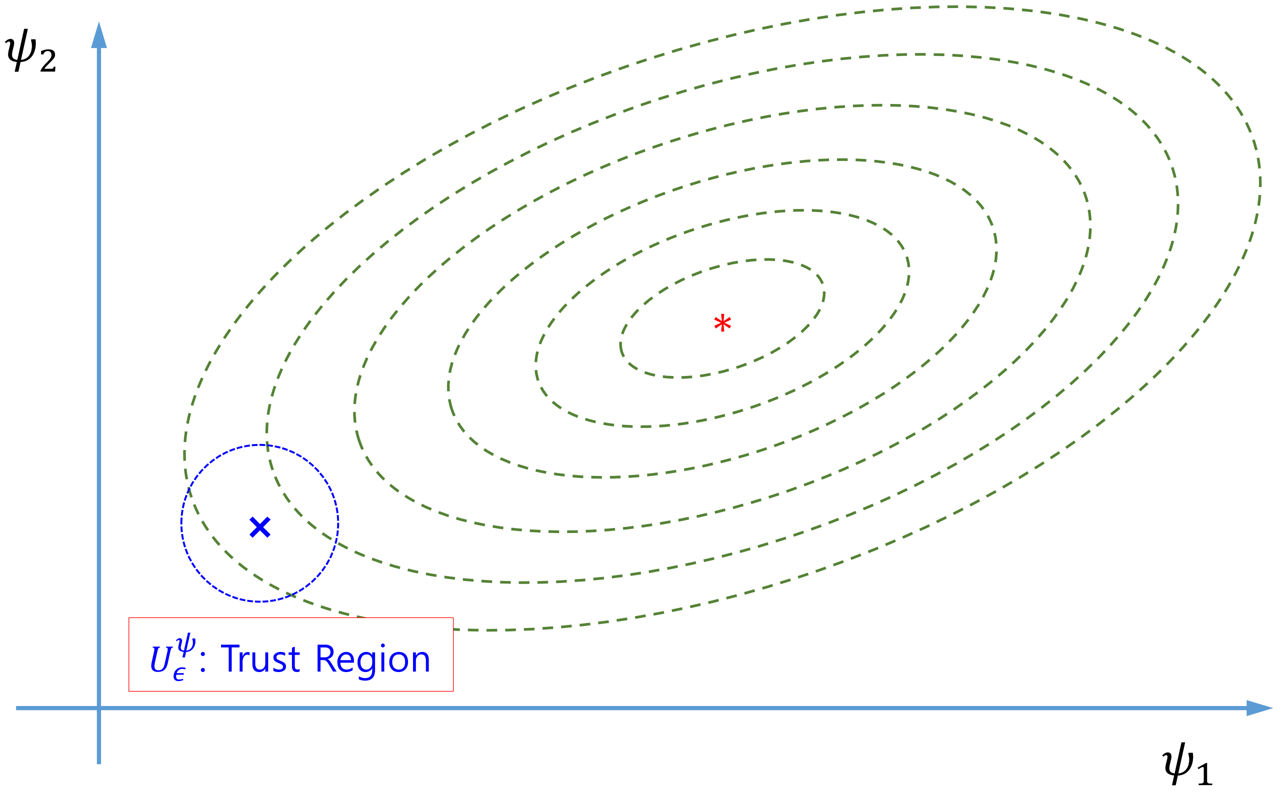


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of x for surrogate training, number K of x for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter ψ
while ψ has not converged **do**
    Sample ψ'_i in the region U_ϵ^ψ, $i = 1, \ldots, N$
    For each ψ'_i, sample inputs $\{x_j^i\}_{j=1}^M \sim q(x)$
    Sample $M \times N$ training examples from simulator $y_{ij} = F(x_j^i; \psi'_i)$
    Store y_{ij}, x_j^i, ψ'_i in history H $i = 1, \ldots, N; j = 1, \ldots, M$
    Extract all y_l, x_l, ψ'_l from history H, iff $d(\psi, \psi'_l) < \epsilon$
    Train generative surrogate model $S_\theta(z_l, x_l; \psi'_l)$, where $z_l \sim \mathcal{N}(0, 1)$
    Fix weights of the surrogate model θ
    Sample $\bar{y}_k = S_\theta(z_k, x_k; \psi)$, $z_k \sim \mathcal{N}(0, 1)$, $x_k \sim q(x)$, $k = 1, \ldots, K$
    $\nabla_\psi \mathbb{E}[\mathcal{R}(\bar{y})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{y}_k} \frac{\partial S_\theta(z_k, x_k; \psi)}{\partial \psi}$
    $\psi \leftarrow \mathrm{SGD}(\psi, \nabla_\psi \mathbb{E}[\mathcal{R}(\bar{y})])$
end while



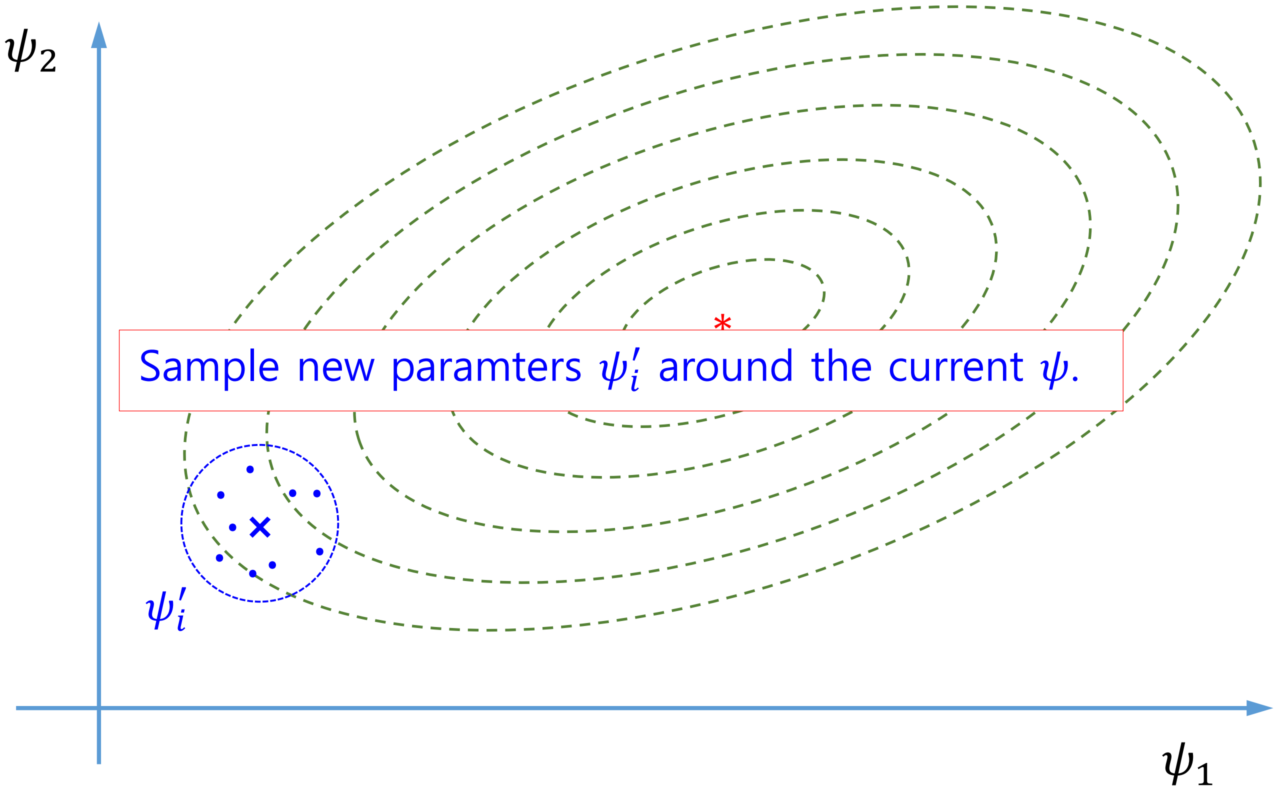
Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

---

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter $\boldsymbol{\psi}$
2: **while** $\boldsymbol{\psi}$ has not converged **do**
3:     Sample $\boldsymbol{\psi}'_i$ in the region $U_\epsilon^{\boldsymbol{\psi}}$, $i = 1, \dots, N$
4:     For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}_j^i\}_{j=1}^M \sim q(\boldsymbol{x})$
5:     Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}_j^i; \boldsymbol{\psi}'_i)$
6:     Store $\boldsymbol{y}_{ij}, \boldsymbol{x}_j^i, \boldsymbol{\psi}'_i$ in history H $i = 1, \dots, N; j = 1, \dots, M$
7:     Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
8:     Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9:     Fix weights of the surrogate model θ
10:     Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})$, $\boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \dots, K$
11:     $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12:     $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
13: **end while**

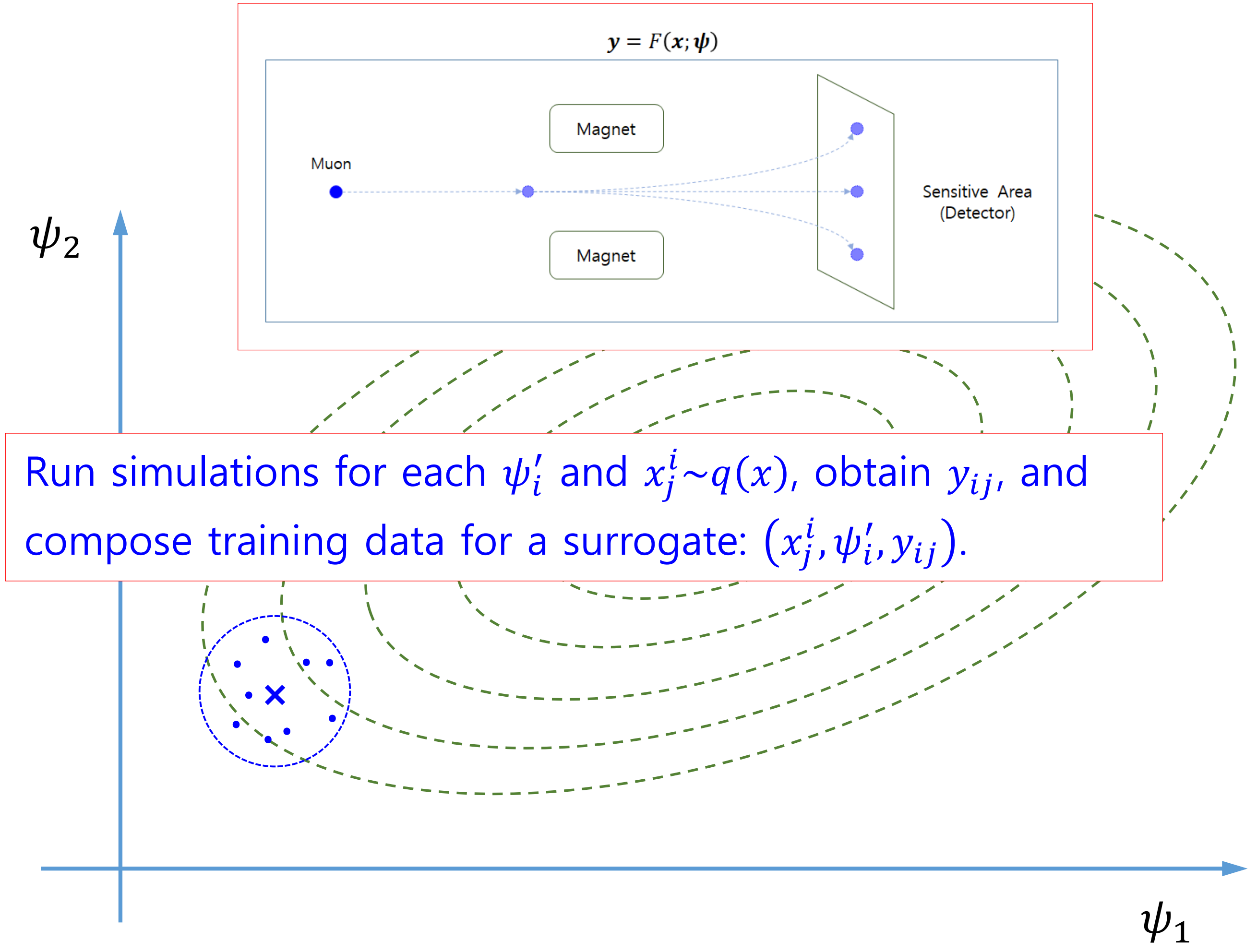


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter $\boldsymbol{\psi}$
while $\boldsymbol{\psi}$ has not converged **do**
  Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \ldots, N$
  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
  Fix weights of the surrogate model θ
  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \ldots, K$
  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
end while



Run simulations for each ψ'_i and $x^i_j \sim q(x)$, obtain y_{ij}, and compose training data for a surrogate: (x^i_j, ψ'_i, y_{ij}).

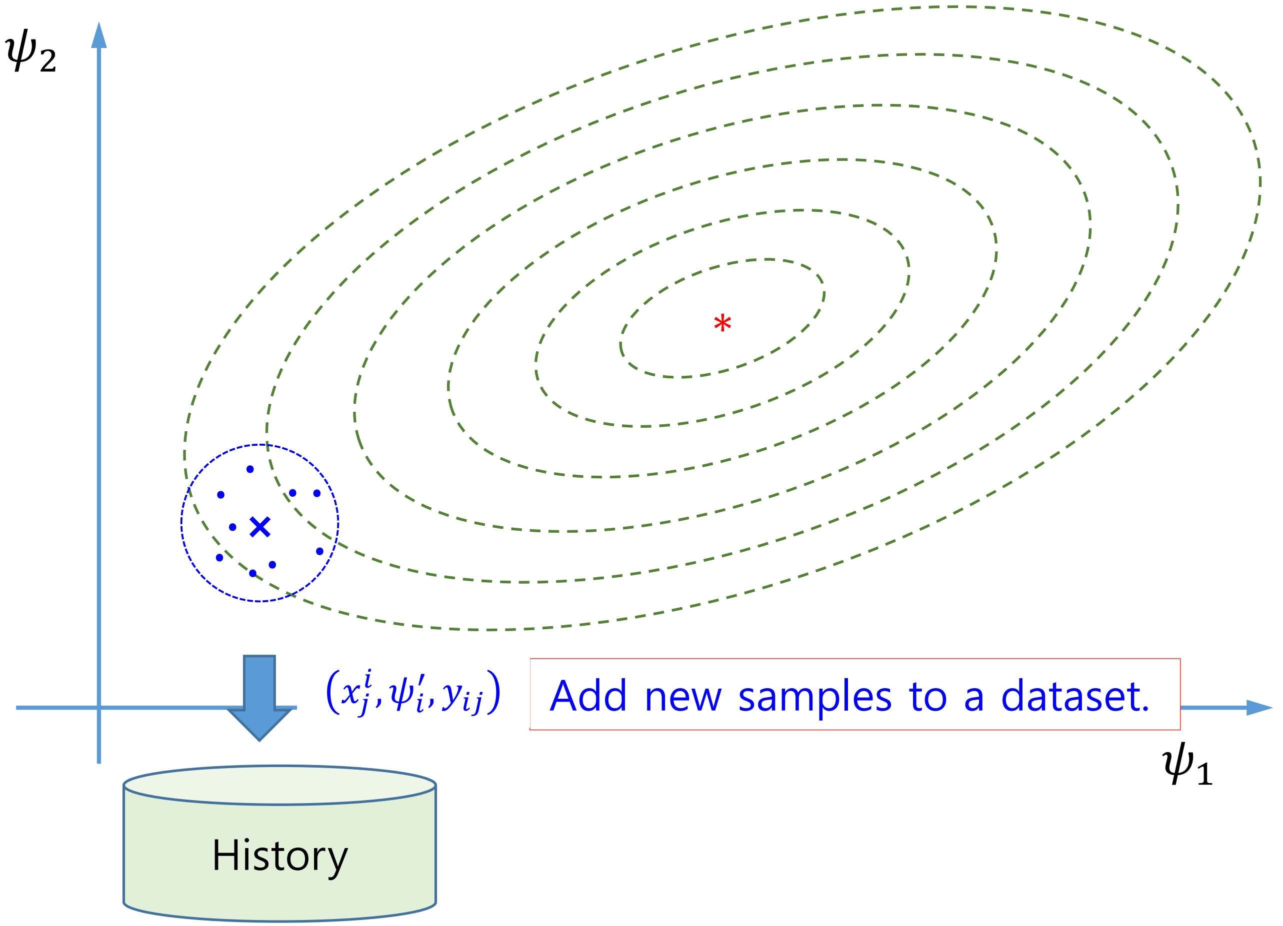
Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

---

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter $\boldsymbol{\psi}$
2: **while** $\boldsymbol{\psi}$ has not converged **do**
3: Sample $\boldsymbol{\psi}'_i$ in the region $U_\epsilon^{\boldsymbol{\psi}}$, $i = 1, \ldots, N$
4: For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}_{j=1}^M \sim q(\boldsymbol{x})$
5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})$, $\boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \ldots, K$
11: $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12: $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
13: **end while**

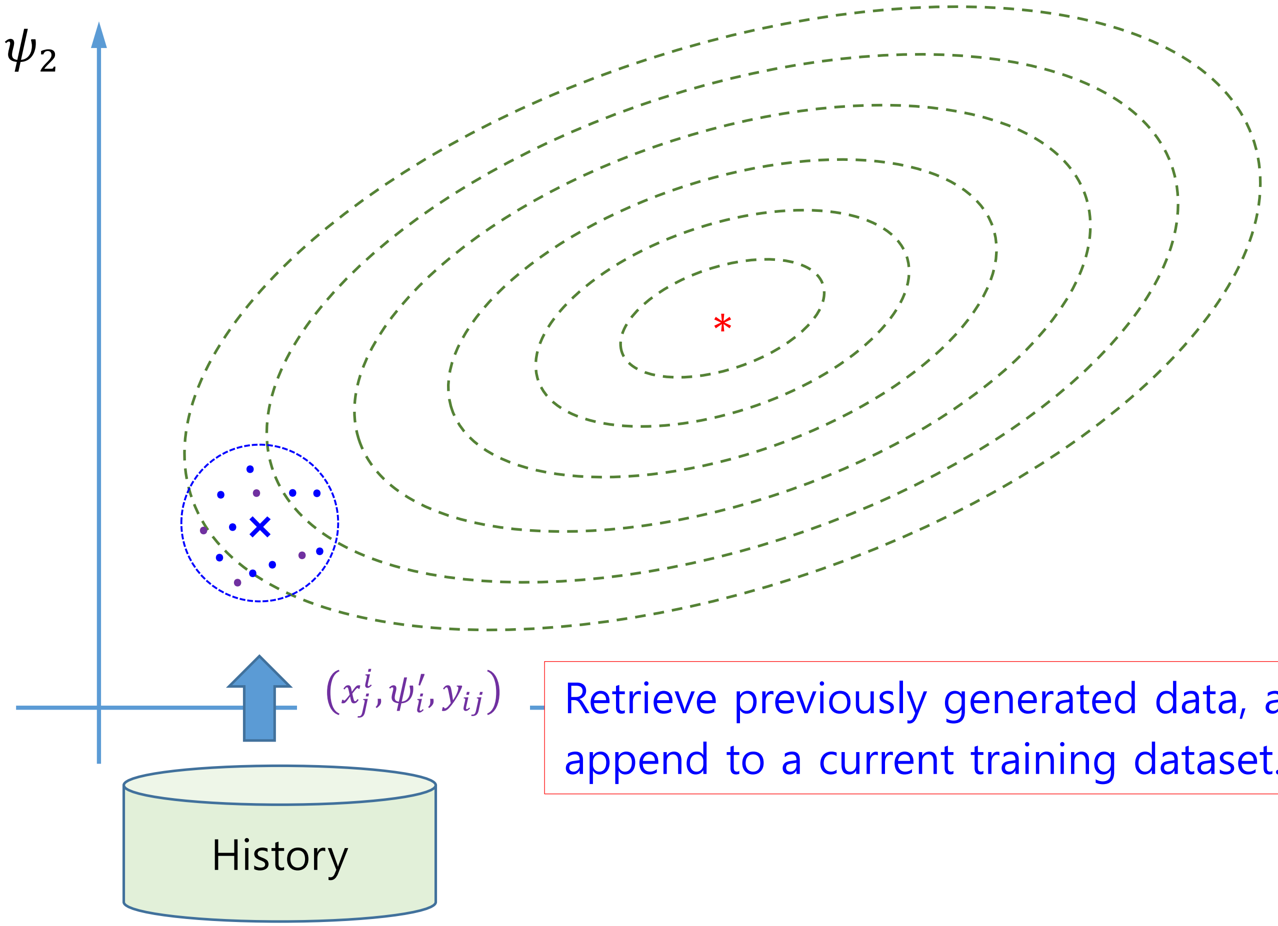


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter ψ
2: **while** ψ has not converged **do**
3: Sample $\boldsymbol{\psi}'_i$ in the region U^{ψ}_{ϵ}, $i = 1, \ldots, N$
4: For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \ldots, K$
11: $\nabla_\psi \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12: $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_\psi \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
13: **end while**



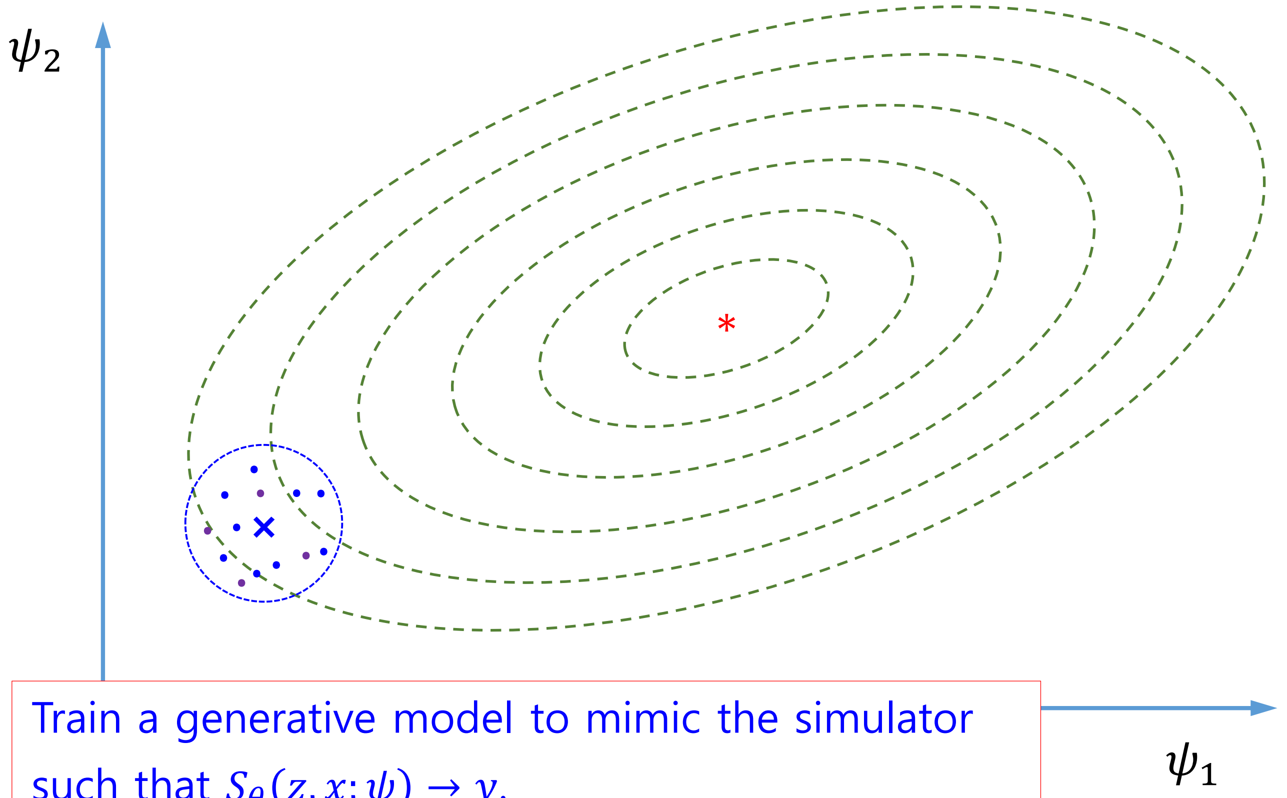
Retrieve previously generated data, and append to a current training dataset.

Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter ψ
while ψ has not converged **do**
  Sample $\boldsymbol{\psi}'_i$ in the region $U_\epsilon^{\boldsymbol{\psi}}$, $i = 1, \dots, N$
  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}_j^i\}_{j=1}^M \sim q(\boldsymbol{x})$
  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}_j^i; \boldsymbol{\psi}'_i)$
  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}_j^i, \boldsymbol{\psi}'_i$ in history H $i = 1, \dots, N; j = 1, \dots, M$
  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0,1)$
  Fix weights of the surrogate model θ
  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})$, $\boldsymbol{z}_k \sim \mathcal{N}(0,1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \dots, K$
  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
end while



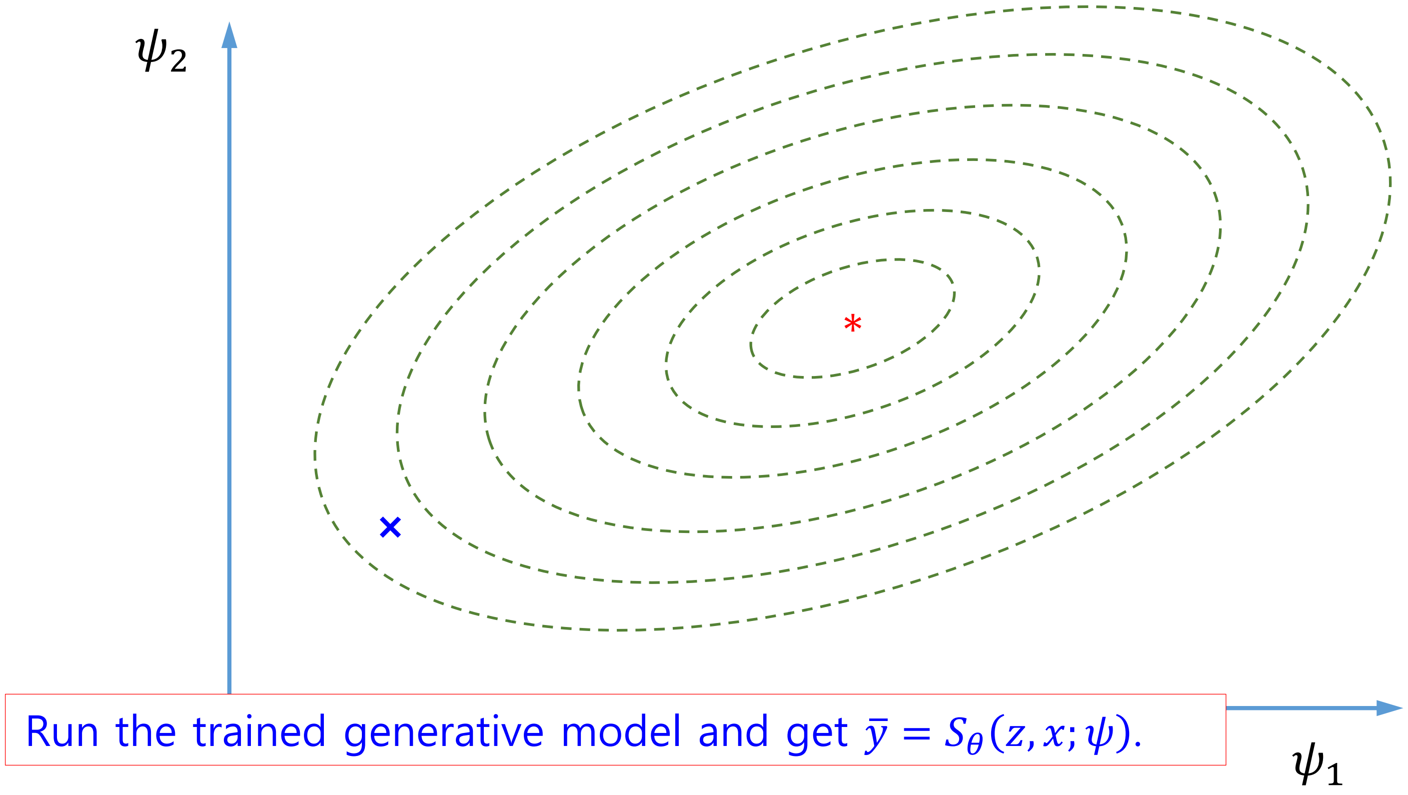
Train a generative model to mimic the simulator such that $S_\theta(z, x; \psi) \to y$.

Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter $\boldsymbol{\psi}$
2: **while** $\boldsymbol{\psi}$ has not converged **do**
3: Sample $\boldsymbol{\psi}'_i$ in the region $U_\epsilon^{\boldsymbol{\psi}}$, $i = 1, \dots, N$
4: For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}_j^i\}_{j=1}^M \sim q(\boldsymbol{x})$
5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}_j^i; \boldsymbol{\psi}'_i)$
6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}_j^i, \boldsymbol{\psi}'_i$ in history H $i = 1, \dots, N; j = 1, \dots, M$
7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})$, $\boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \dots, K$
11: $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12: $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
13: **end while**



Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

---

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter $\boldsymbol{\psi}$

while $\boldsymbol{\psi}$ has not converged **do**

  Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \ldots, N$

  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$

  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$

  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$

  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$

  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$

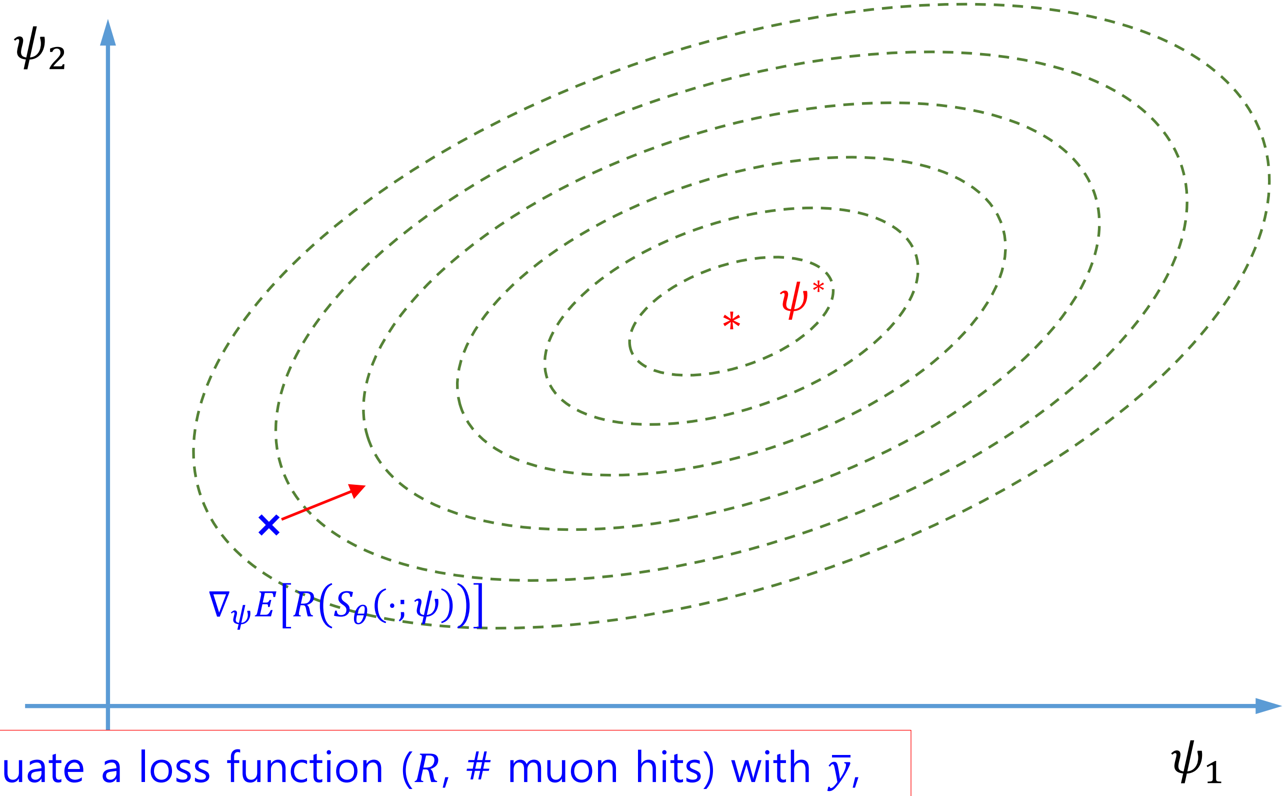
  Fix weights of the surrogate model θ

  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \ldots, K$

  $\boxed{\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}}$

  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$

end while



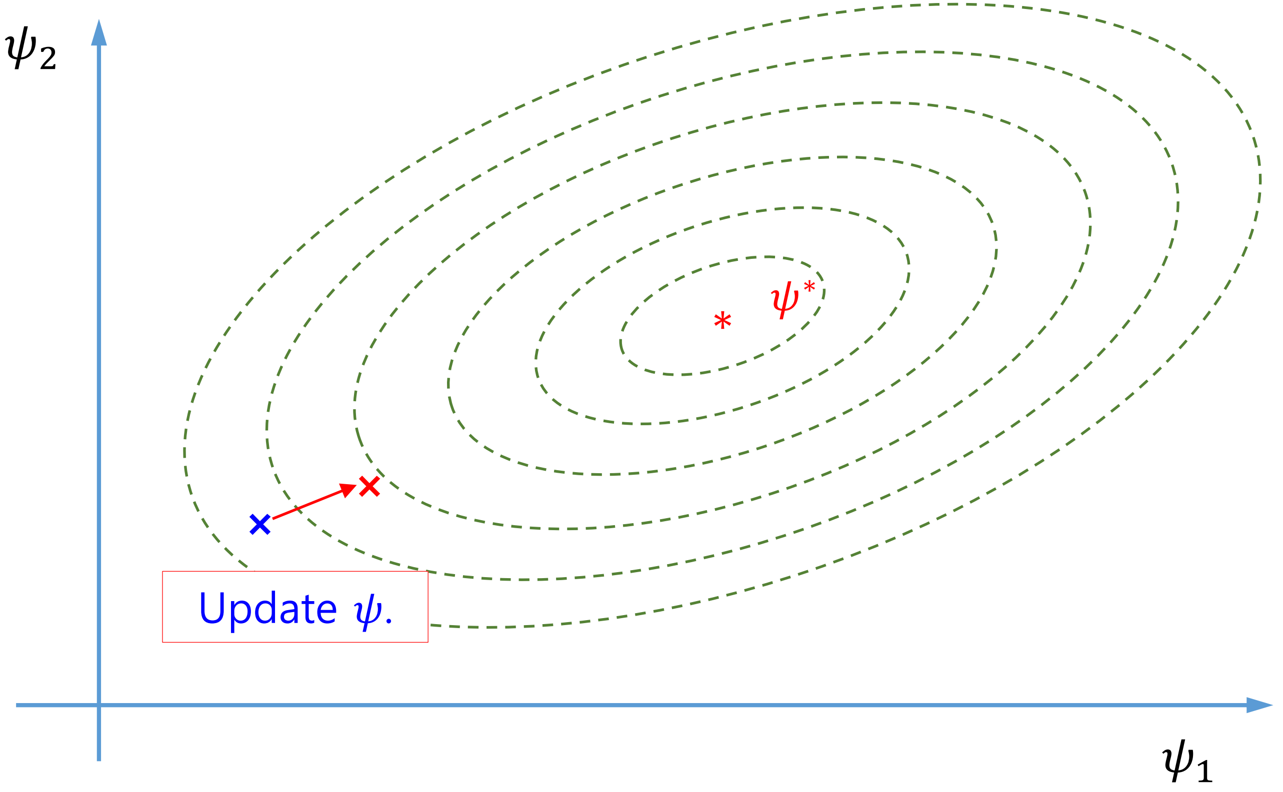
Evaluate a loss function (R, # muon hits) with $\bar{y}$, and compute the gradient of R w.r.t ψ.

Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter $\boldsymbol{\psi}$
2: **while** $\boldsymbol{\psi}$ has not converged **do**
3: Sample $\boldsymbol{\psi}_i'$ in the region $U_\epsilon^{\boldsymbol{\psi}}$, $i = 1, \dots, N$
4: For each $\boldsymbol{\psi}_i'$, sample inputs $\{\boldsymbol{x}_j^i\}_{j=1}^M \sim q(\boldsymbol{x})$
5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}_j^i; \boldsymbol{\psi}_i')$
6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}_j^i, \boldsymbol{\psi}_i'$ in history H $i = 1, \dots, N; j = 1, \dots, M$
7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}_l'$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}_l') < \epsilon$
8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}_l')$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})$, $\boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \dots, K$
11: $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12: $\boxed{\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])}$
13: **end while**

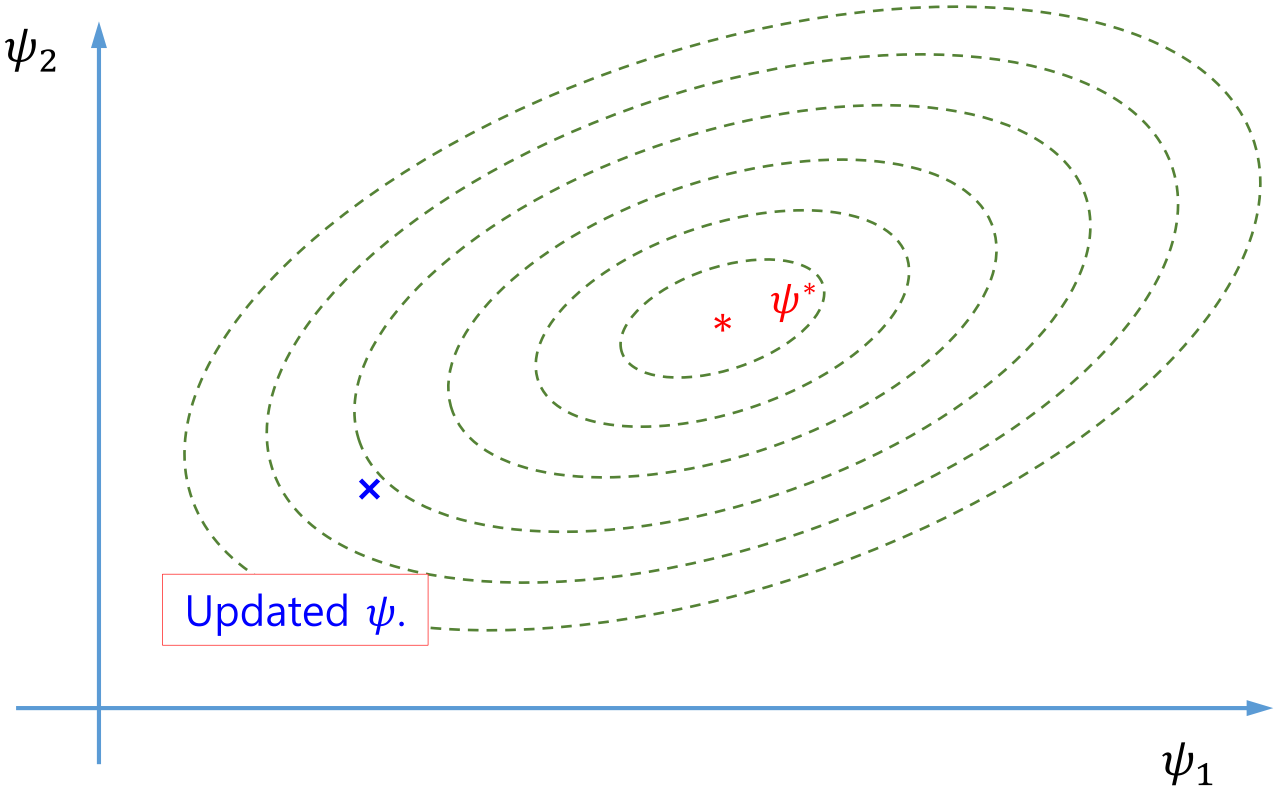


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of x for surrogate training, number K of x for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter ψ
while ψ has not converged **do**
  Sample ψ'_i in the region U^ψ_ϵ, $i = 1, \ldots, N$
  For each ψ'_i, sample inputs $\{x^i_j\}^M_{j=1} \sim q(x)$
  Sample $M \times N$ training examples from simulator $y_{ij} = F(x^i_j; \psi'_i)$
  Store y_{ij}, x^i_j, ψ'_i in history H $i = 1, \ldots, N; j = 1, \ldots, M$
  Extract all y_l, x_l, ψ'_l from history H, iff $d(\psi, \psi'_l) < \epsilon$
  Train generative surrogate model $S_\theta(z_l, x_l; \psi'_l)$, where $z_l \sim \mathcal{N}(0, 1)$
  Fix weights of the surrogate model θ
  Sample $\bar{y}_k = S_\theta(z_k, x_k; \psi)$, $z_k \sim \mathcal{N}(0, 1)$, $x_k \sim q(x)$, $k = 1, \ldots, K$
  $\nabla_\psi \mathbb{E}[\mathcal{R}(\bar{y})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{y}_k} \frac{\partial S_\theta(z_k, x_k; \psi)}{\partial \psi}$
  $\psi \leftarrow \mathrm{SGD}(\psi, \nabla_\psi \mathbb{E}[\mathcal{R}(\bar{y})])$
end while

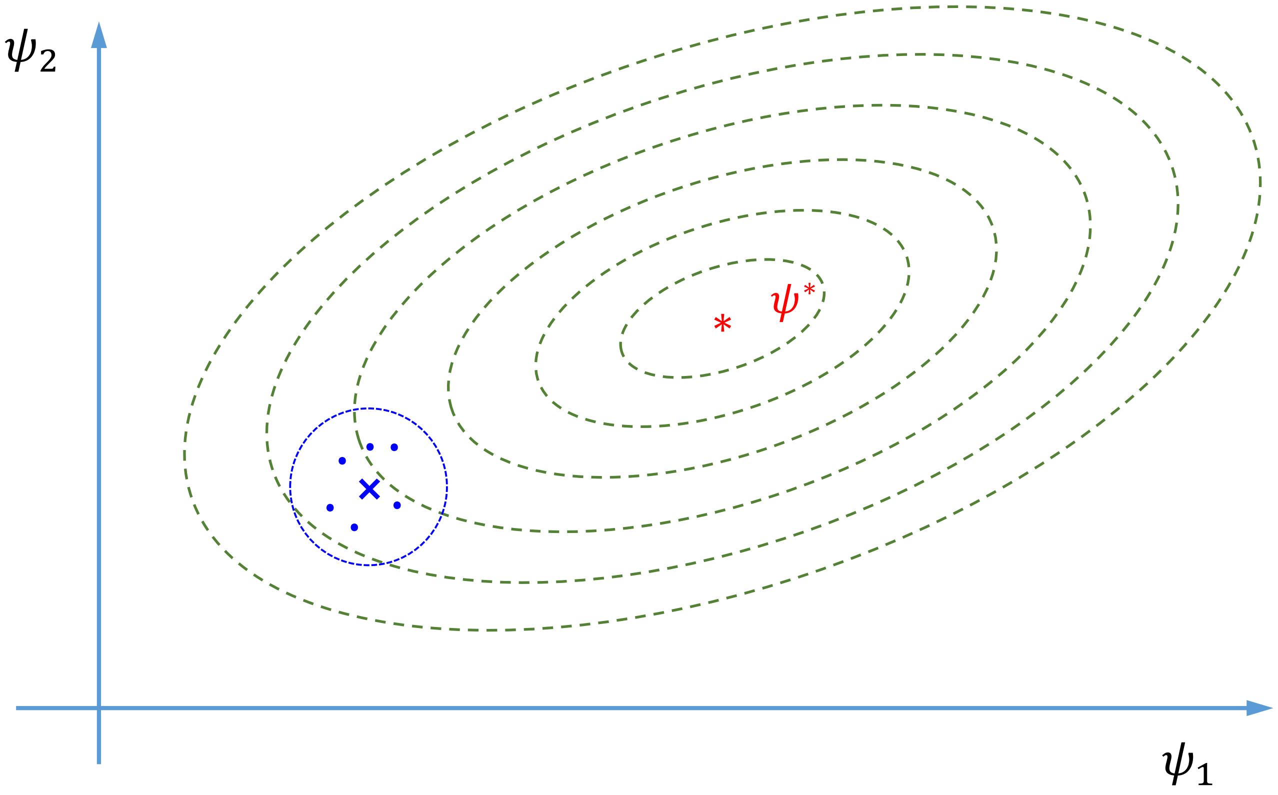


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter $\boldsymbol{\psi}$
while $\boldsymbol{\psi}$ has not converged **do**
  Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \ldots, N$
  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
  Fix weights of the surrogate model θ
  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \ldots, K$
  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
end while



Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter $\boldsymbol{\psi}$

while $\boldsymbol{\psi}$ has not converged **do**

  Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \dots, N$

  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$

  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$

  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \dots, N; j = 1, \dots, M$

  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$

  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$

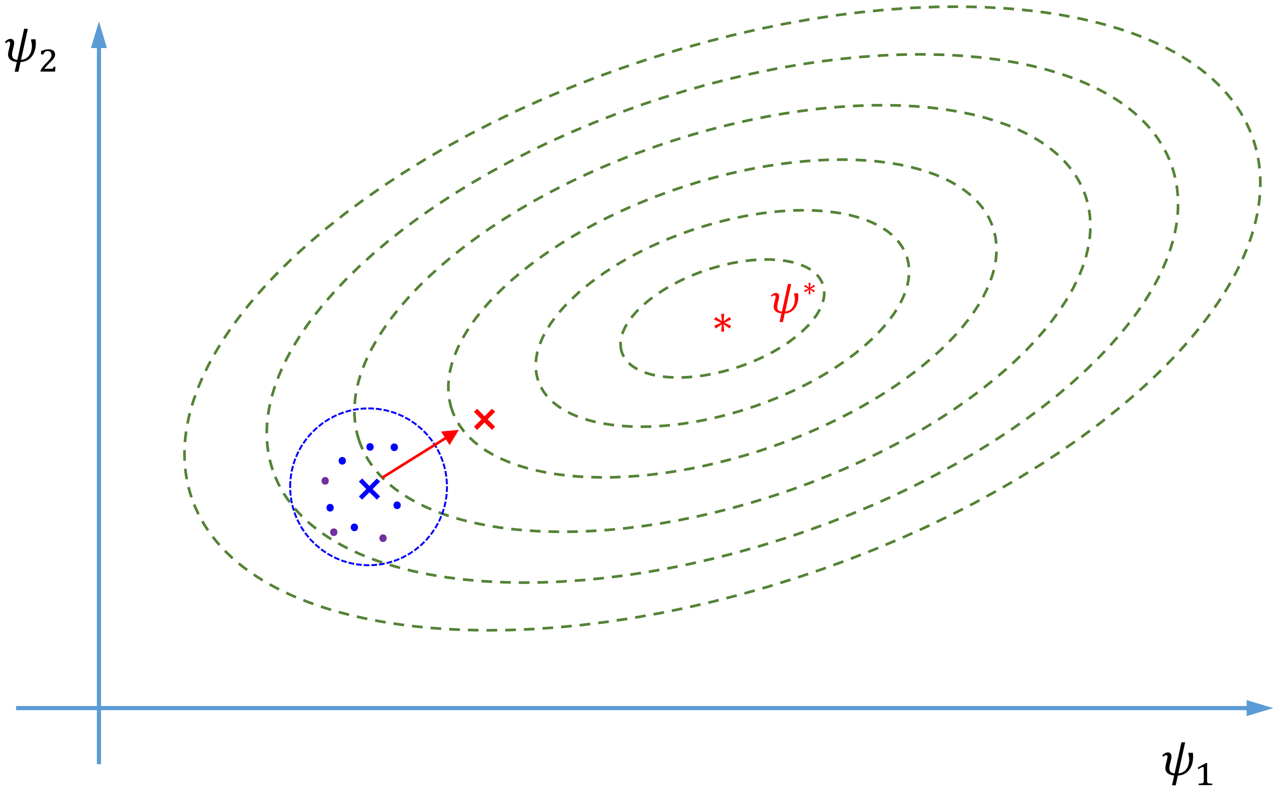
  Fix weights of the surrogate model θ

  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \dots, K$

  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$

  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$

end while

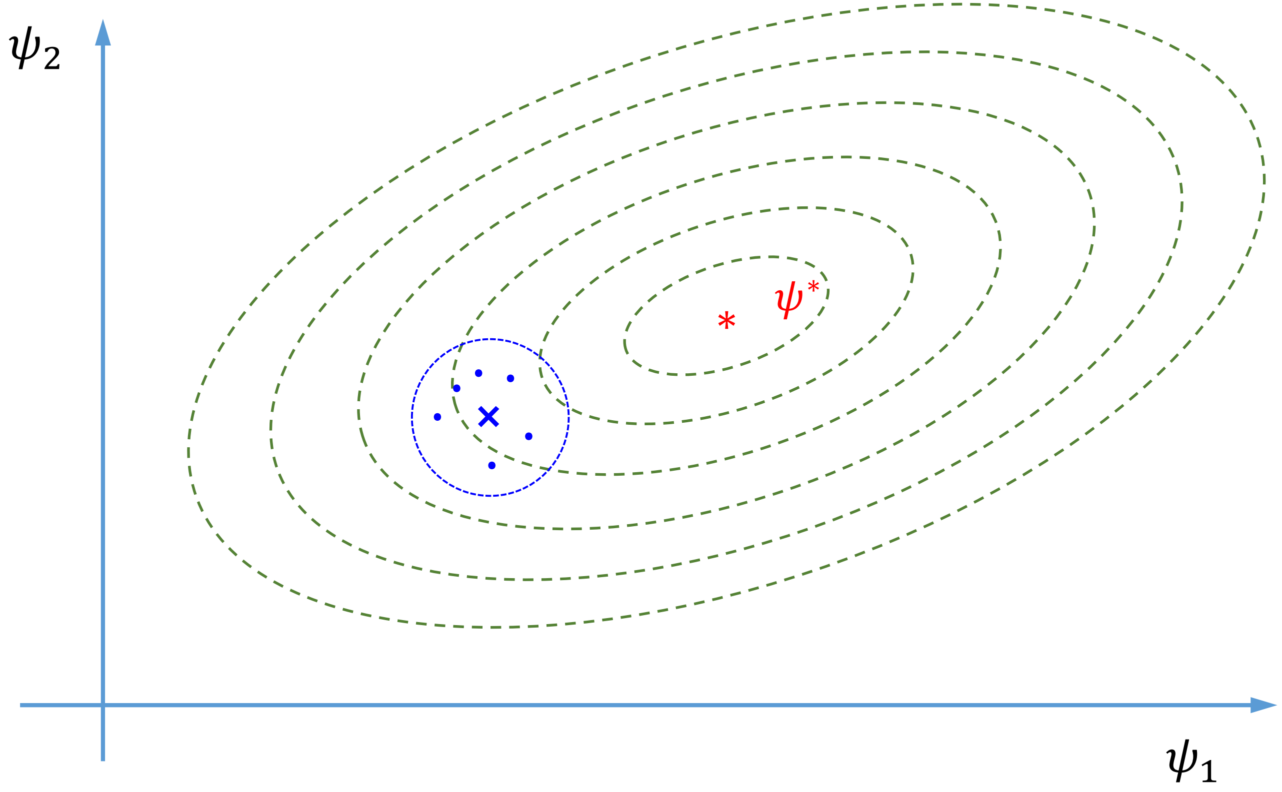


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter $\boldsymbol{\psi}$
2: **while** $\boldsymbol{\psi}$ has not converged **do**
3: Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \ldots, N$
4: For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \ldots, K$
11: $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12: $\boldsymbol{\psi} \leftarrow \text{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
13: **end while**



Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter $\boldsymbol{\psi}$

while $\boldsymbol{\psi}$ has not converged **do**

  Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \dots, N$

  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$

  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$

  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \dots, N; j = 1, \dots, M$

  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$

  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$

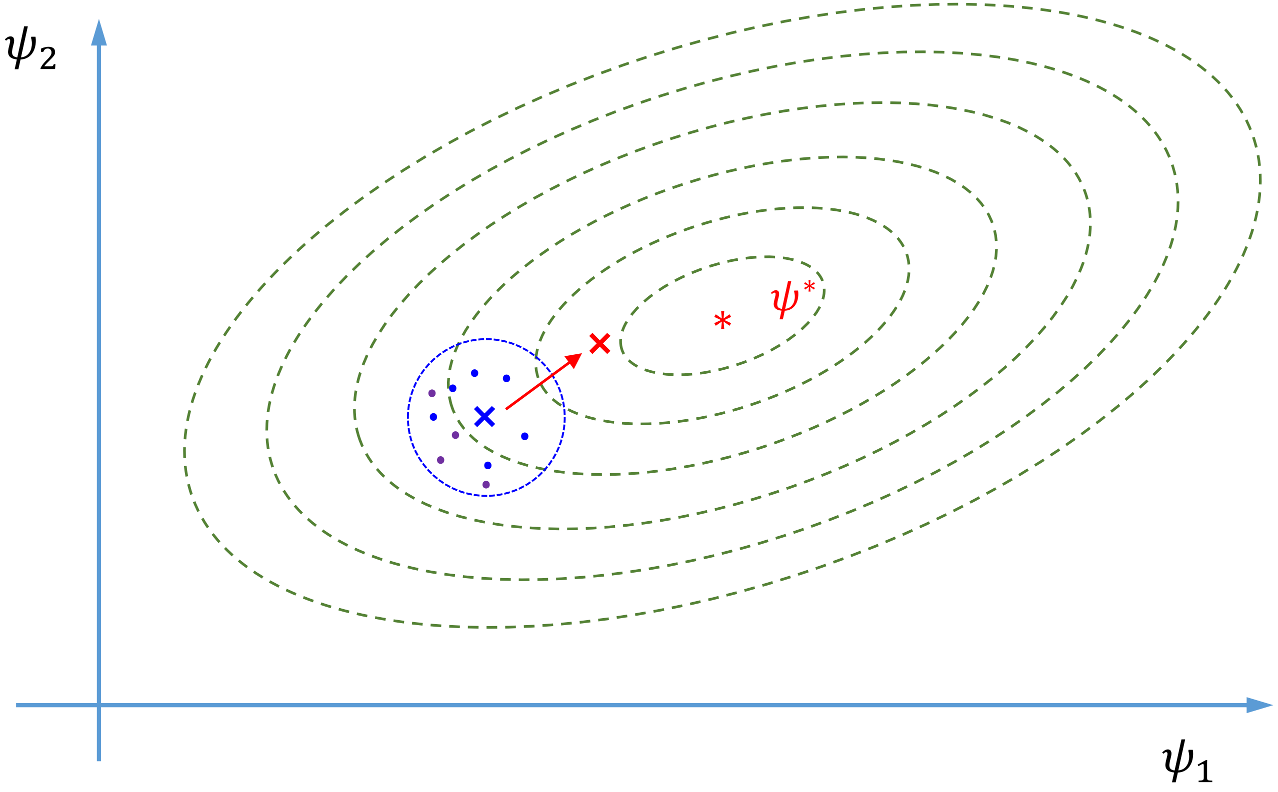
  Fix weights of the surrogate model θ

  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x})$, $k = 1, \dots, K$

  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$

  $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$

end while

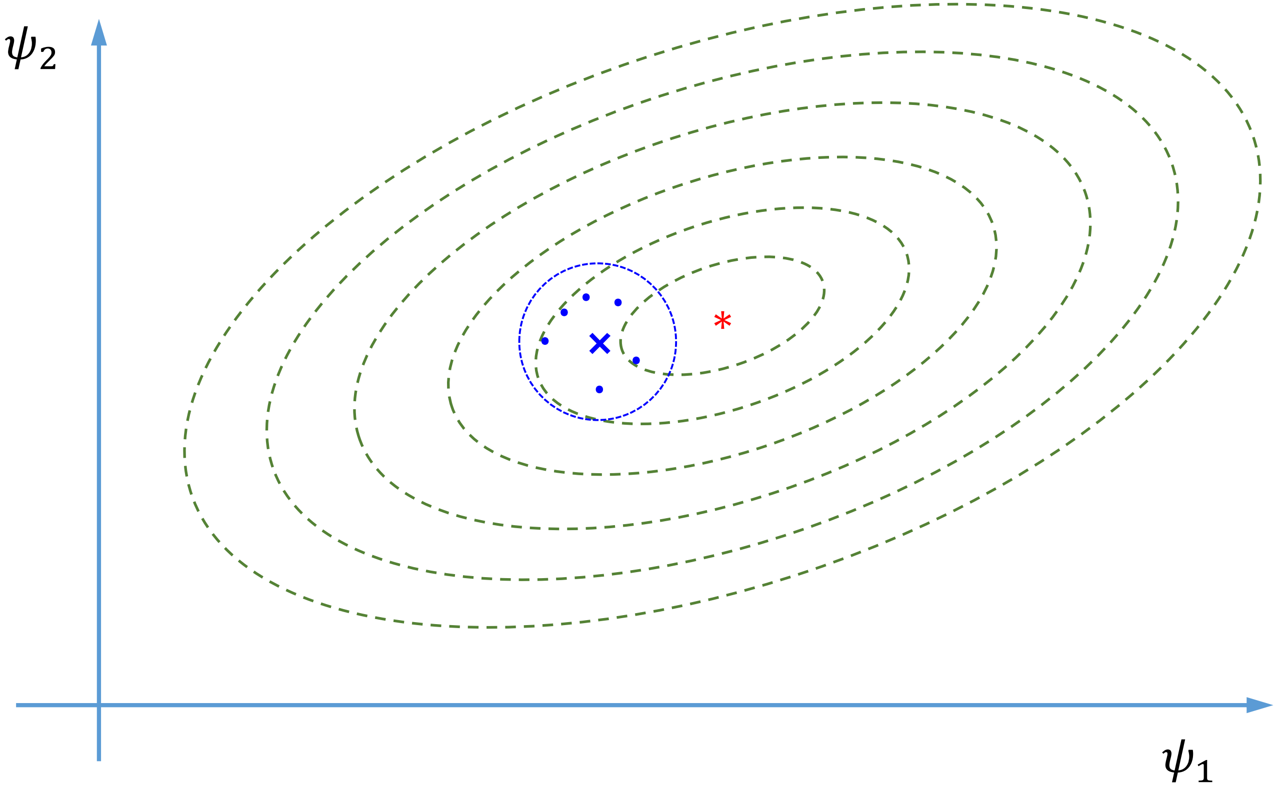


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of ψ, number M of x for surrogate training, number K of x for ψ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter ψ
2: **while** ψ has not converged **do**
3: $\boxed{\text{Sample } \boldsymbol{\psi}'_i \text{ in the region } U^{\boldsymbol{\psi}}_\epsilon,\ i = 1, \ldots, N}$
4: For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$
7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
9: Fix weights of the surrogate model θ
10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \ldots, K$
11: $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
12: $\boldsymbol{\psi} \leftarrow \text{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
13: **end while**



Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

1: Choose initial parameter $\boldsymbol{\psi}$

2: **while** $\boldsymbol{\psi}$ has not converged **do**

3: Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \ldots, N$

4: For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$

5: Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$

6: Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \ldots, N; j = 1, \ldots, M$

7: Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$

8: Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$

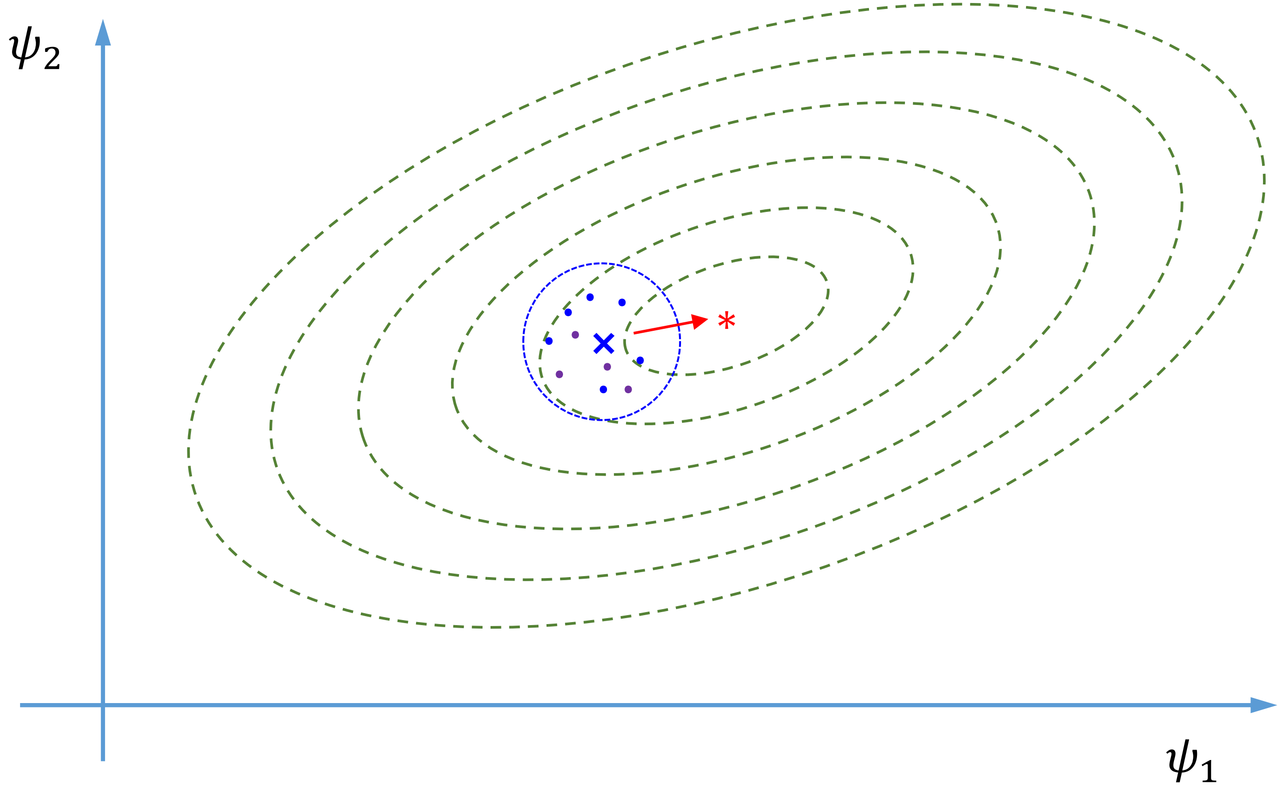
9: Fix weights of the surrogate model θ

10: Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \ldots, K$

11: $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$

12: $\boldsymbol{\psi} \leftarrow \mathrm{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$

13: **end while**

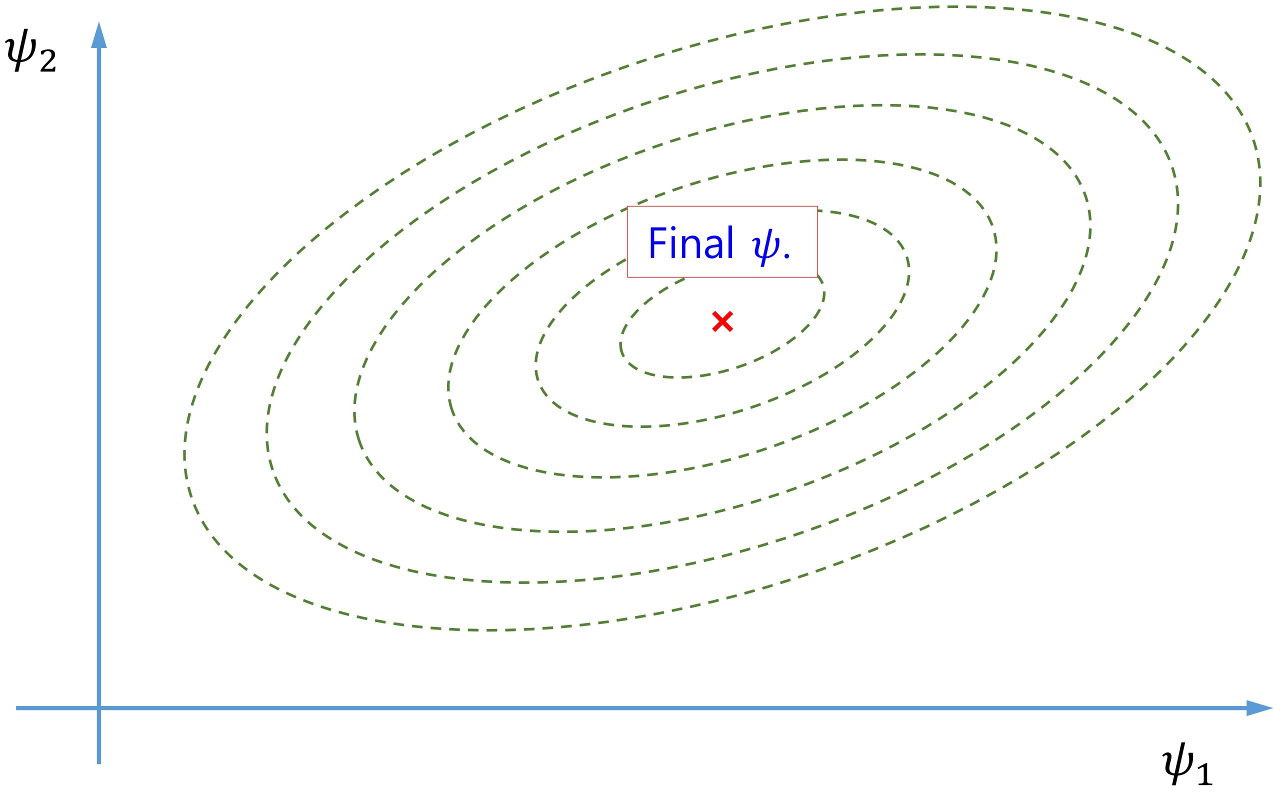


Local Generative Surrogate Optimization

Algorithm 1 Local Generative Surrogate Optimization (L-GSO) procedure

Require: number N of $\boldsymbol{\psi}$, number M of $\boldsymbol{x}$ for surrogate training, number K of $\boldsymbol{x}$ for $\boldsymbol{\psi}$ optimization step, trust region U_ϵ, size of the neighborhood ϵ, Euclidean distance d

Choose initial parameter $\boldsymbol{\psi}$
while $\boldsymbol{\psi}$ has not converged **do**
  Sample $\boldsymbol{\psi}'_i$ in the region $U^{\boldsymbol{\psi}}_\epsilon$, $i = 1, \dots, N$
  For each $\boldsymbol{\psi}'_i$, sample inputs $\{\boldsymbol{x}^i_j\}^M_{j=1} \sim q(\boldsymbol{x})$
  Sample $M \times N$ training examples from simulator $\boldsymbol{y}_{ij} = F(\boldsymbol{x}^i_j; \boldsymbol{\psi}'_i)$
  Store $\boldsymbol{y}_{ij}, \boldsymbol{x}^i_j, \boldsymbol{\psi}'_i$ in history H $i = 1, \dots, N; j = 1, \dots, M$
  Extract all $\boldsymbol{y}_l, \boldsymbol{x}_l, \boldsymbol{\psi}'_l$ from history H, iff $d(\boldsymbol{\psi}, \boldsymbol{\psi}'_l) < \epsilon$
  Train generative surrogate model $S_\theta(\boldsymbol{z}_l, \boldsymbol{x}_l; \boldsymbol{\psi}'_l)$, where $\boldsymbol{z}_l \sim \mathcal{N}(0, 1)$
  Fix weights of the surrogate model θ
  Sample $\bar{\boldsymbol{y}}_k = S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi}), \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$, $\boldsymbol{x}_k \sim q(\boldsymbol{x}),\ k = 1, \dots, K$
  $\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})] \leftarrow \frac{1}{K} \sum_{k=1}^{K} \frac{\partial \mathcal{R}}{\partial \bar{\boldsymbol{y}}_k} \frac{\partial S_\theta(\boldsymbol{z}_k, \boldsymbol{x}_k; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}}$
  $\boldsymbol{\psi} \leftarrow \text{SGD}(\psi, \nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\bar{\boldsymbol{y}})])$
end while



Q & A