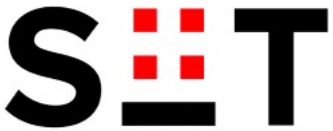
Gradient-Based Methods for Black-Box Tuning and Optimization

Andrey Ustyuzhanin[1,2] on behalf of the team

[1] HSE University

[2] Schaffhausen Institute of Technology



August 2021, XIVth Quark confinement

Optimization project examples

Example 1: Detector Design Optimization (SHiP)

$$\text{background}(\psi) = \mathop{\mathbb{E}}_{\text{event}} \mathbb{I}\left[\text{muons} > 0 \mid \text{event}, \psi\right] \to \min$$

- SHiP searches for Dark Matter, thus we want to minimize amount of background (muons)
- How can we optimize shield design (ψ) with respect to **smallest number of muons** and **budget limits**?
  - Noisy (stochastic) target
  - Computationally expensive
  - Non-differentiable simulator

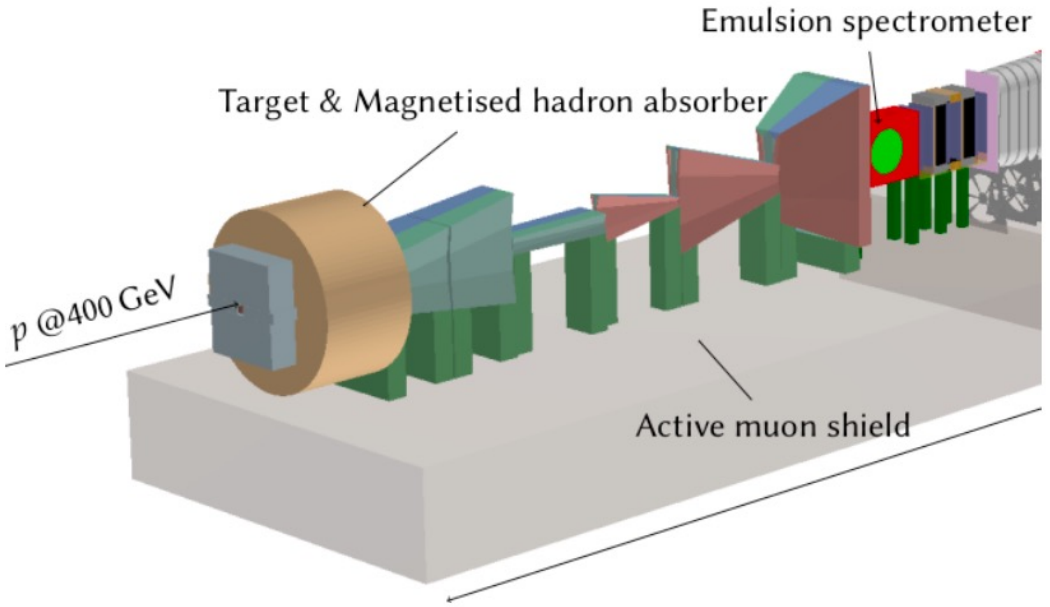
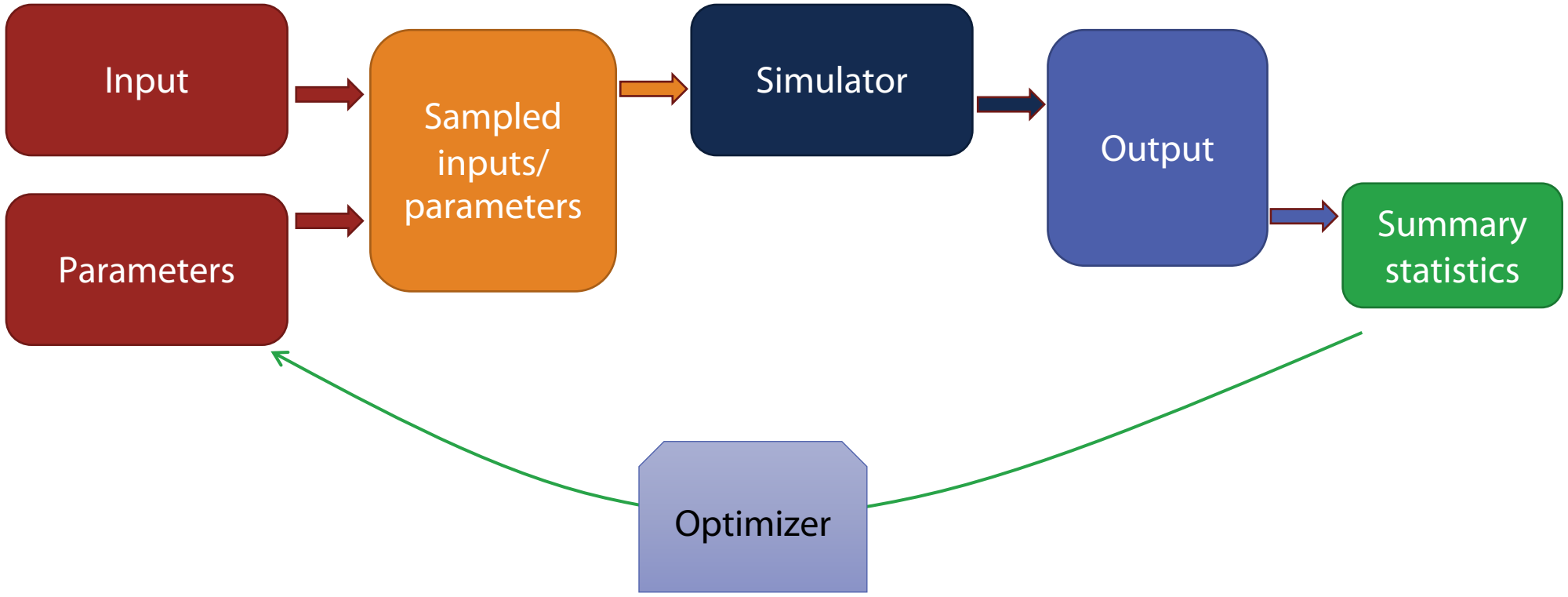


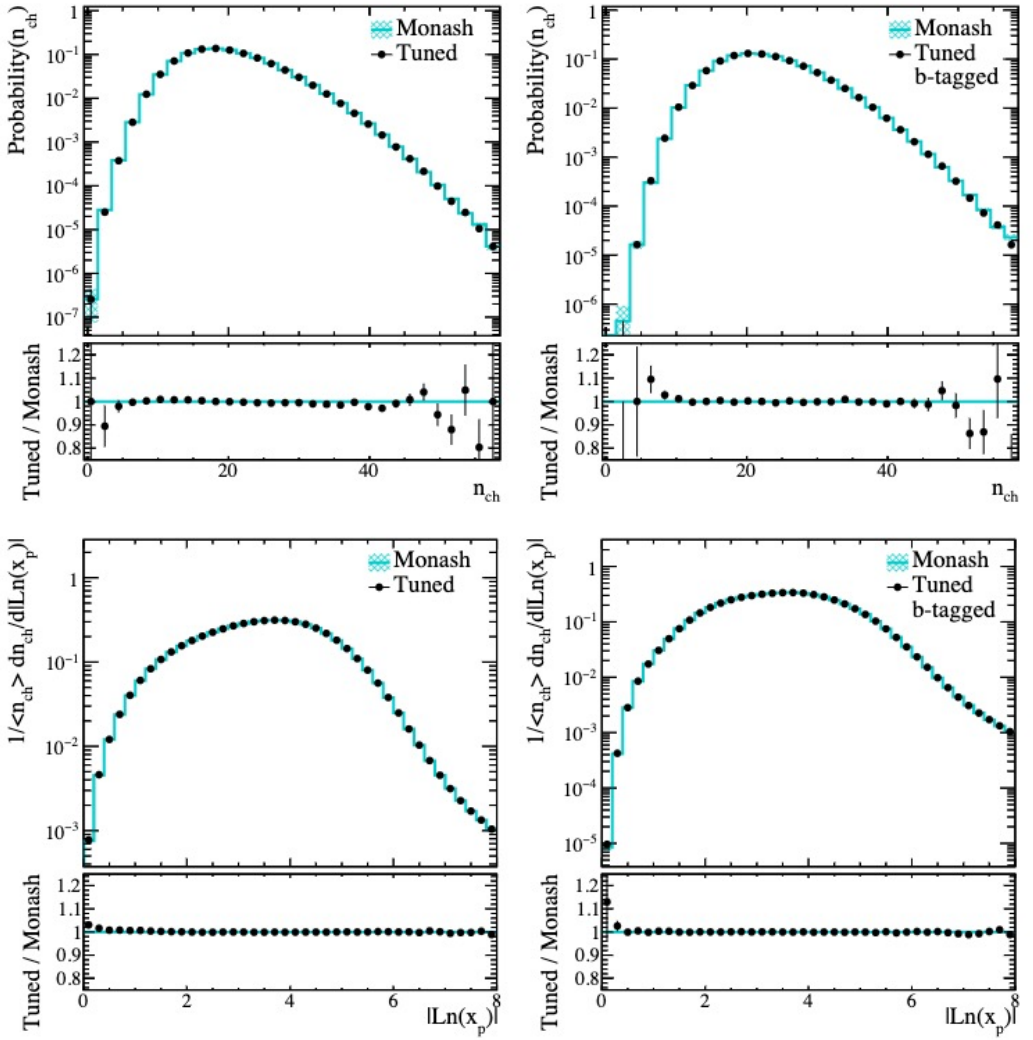
Image source: Oliver Lantwin, Bayesian optimisation of the SHiP muon shield.

Optimization layout



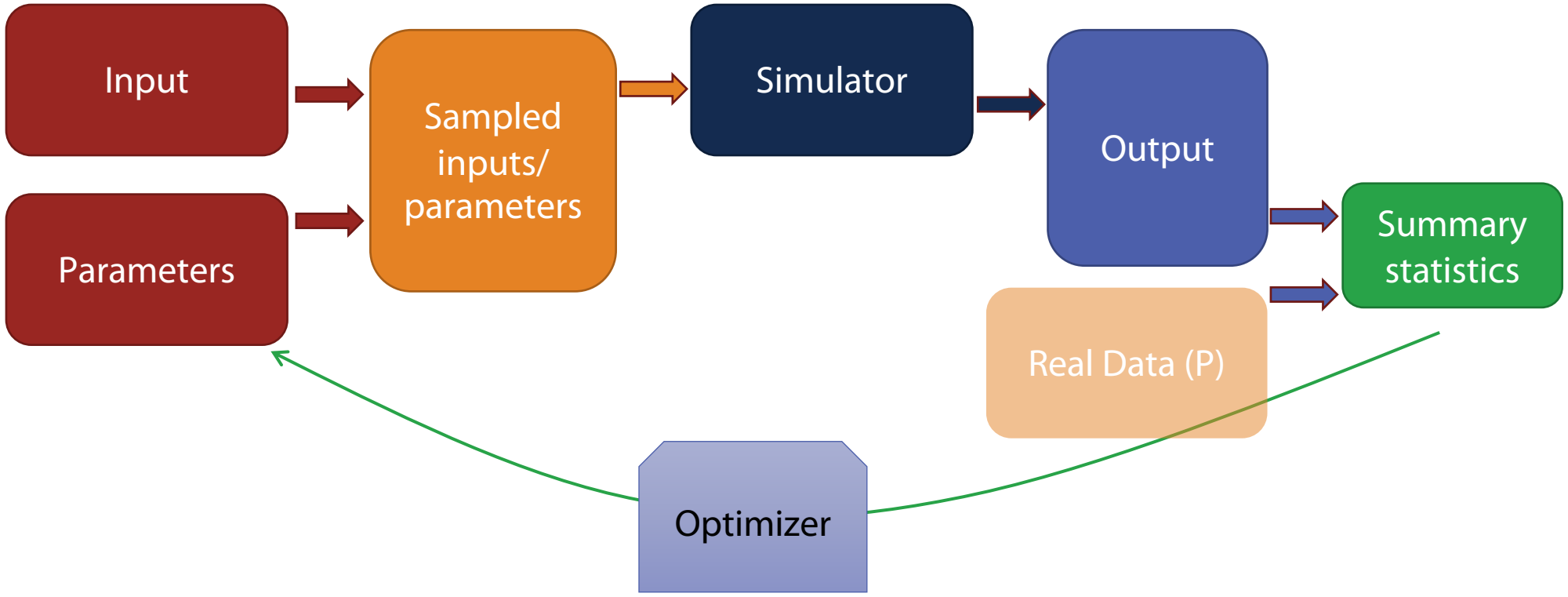
Example 2. Simulation fine-tuning

- How can we fine-tune a simulator given real-data sample P?
- Simulated sample $\boldsymbol{Q}$, parametrized by ψ, such that $Q_\psi \longrightarrow P$ everywhere
  - Noisy (stochastic) target
  - Computationally expensive
  - Non-differentiable simulator



https://bit.ly/3eLtyxl

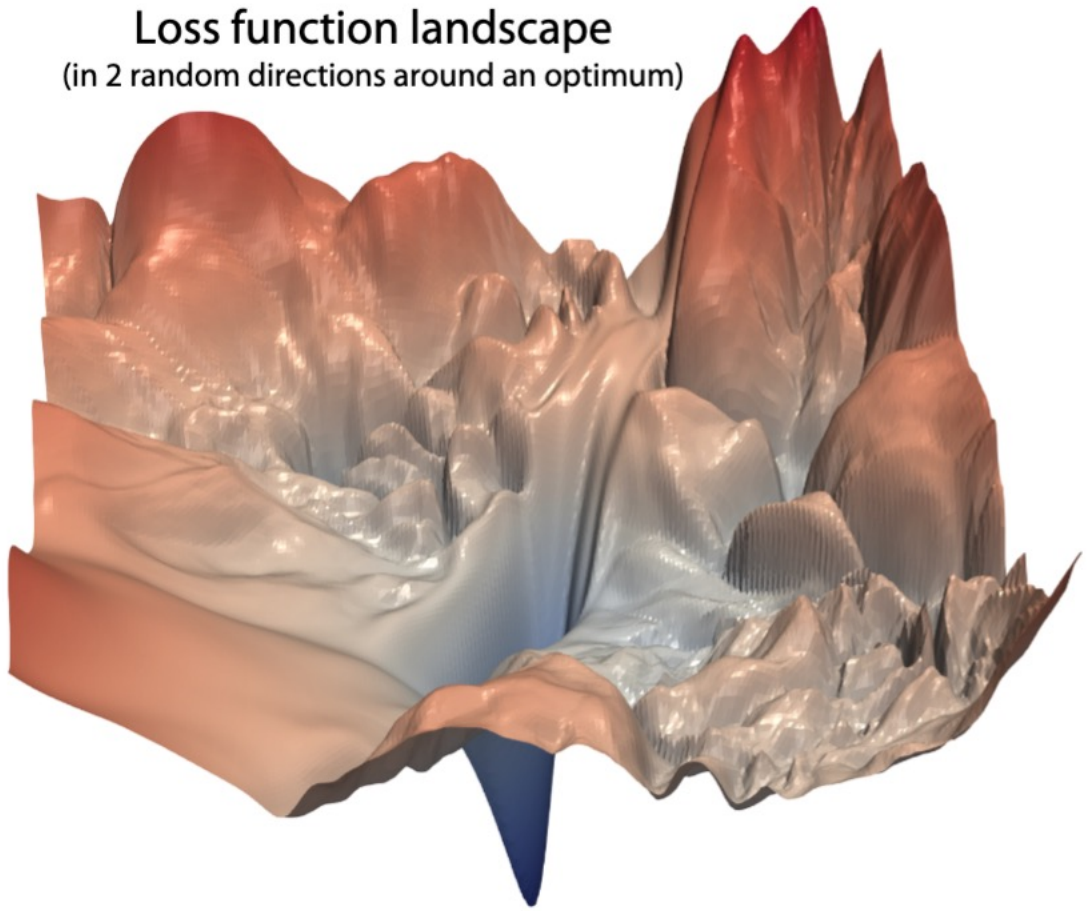
Optimization layout



Common optimization approaches

Optimization method families

- Gradient-free
  - Random search,
  - Simulated annealing,
  - Evolution strategies, …
- Gradient
  - Stochastic GD, ADAM, RMSProp, Langevin SGD, …
- Surrogate + gradient
  - Bayesian,
  - **Variational optimization,**
  - Guided Evolution Strategies,
  - “Learning to Simulate”,
  - “Backprop through the void”,
  - Adversarial Variational Optimization,
  - **NN-based (L-GSO)**,
  - **Adaptive Divergence**, …



https://papers.nips.cc/paper/7875-visualizing-the-loss-landscape-of-neural-nets

Variational bound

Variational Optimization replaces problem:

$$\mathsf{f}(\theta) \to \min_{\theta};$$

with:

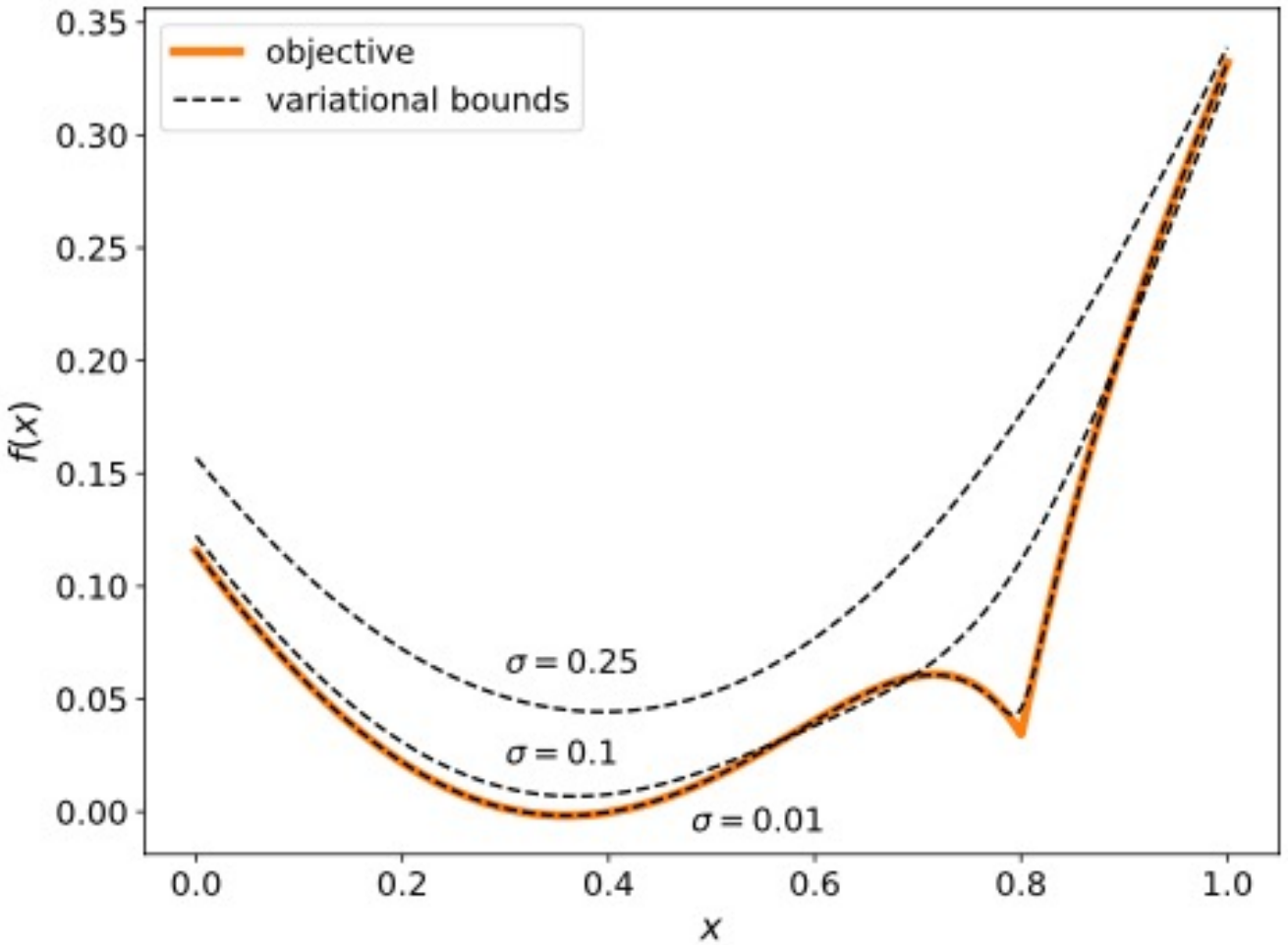
$$\mathsf{J}(\psi) = \mathop{\mathbb{E}}_{\theta \sim \mathsf{P}(\cdot|\psi)} \mathsf{f}(\theta) \to \min_{\psi}$$

where:

- $\mathsf{J}(\psi)$ - variational bound;
- $\mathsf{P}(\cdot \mid \psi)$ - search distribution.

This variational bound is not the only one, nevertheless, it is the most common in Varitional Optimization.

Variational bound: example



Gradient of the variational bound

$$\frac{\partial}{\partial \psi} \mathsf{J}(\psi) = \frac{\partial}{\partial \psi} \mathop{\mathbb{E}}_{\theta \sim \mathsf{P}(\cdot|\psi)} \mathsf{f}(\theta) =$$

$$\frac{\partial}{\partial \psi} \int_{\theta} \mathrm{d}\theta \, \mathsf{f}(\theta) \mathsf{P}(\theta \mid \psi) =$$

$$\int_{\theta} \mathrm{d}\theta \, \mathsf{f}(\theta) \, \frac{\partial}{\partial \psi} \mathsf{P}(\theta \mid \psi) =$$

$$\int_{\theta} \mathrm{d}\theta \, \mathsf{f}(\theta) \, \mathsf{P}(\theta \mid \psi) \, \frac{\partial}{\partial \psi} \log \mathsf{P}(\theta \mid \psi) =$$

$$\mathop{\mathbb{E}}_{\theta \sim \mathsf{P}(\cdot|\psi)} \mathsf{f}(\theta) \frac{\partial}{\partial \psi} \log \mathsf{P}(\theta \mid \psi)$$

Gradient of the variational bound

$$\nabla_\psi \mathsf{J}(\psi) = \mathop{\mathbb{E}}_{\theta \sim \mathsf{P}(\cdot|\psi)} \mathsf{f}(\theta)\, \nabla_\psi \log \mathsf{P}(\theta \mid \psi)$$

$\nabla_\psi \mathsf{J}(\psi)$ does not depend on $\nabla_\theta \mathsf{f}(\theta)$

REINFORCE-style gradient estimator [Williams, 1992]

Gradient of the variational bound

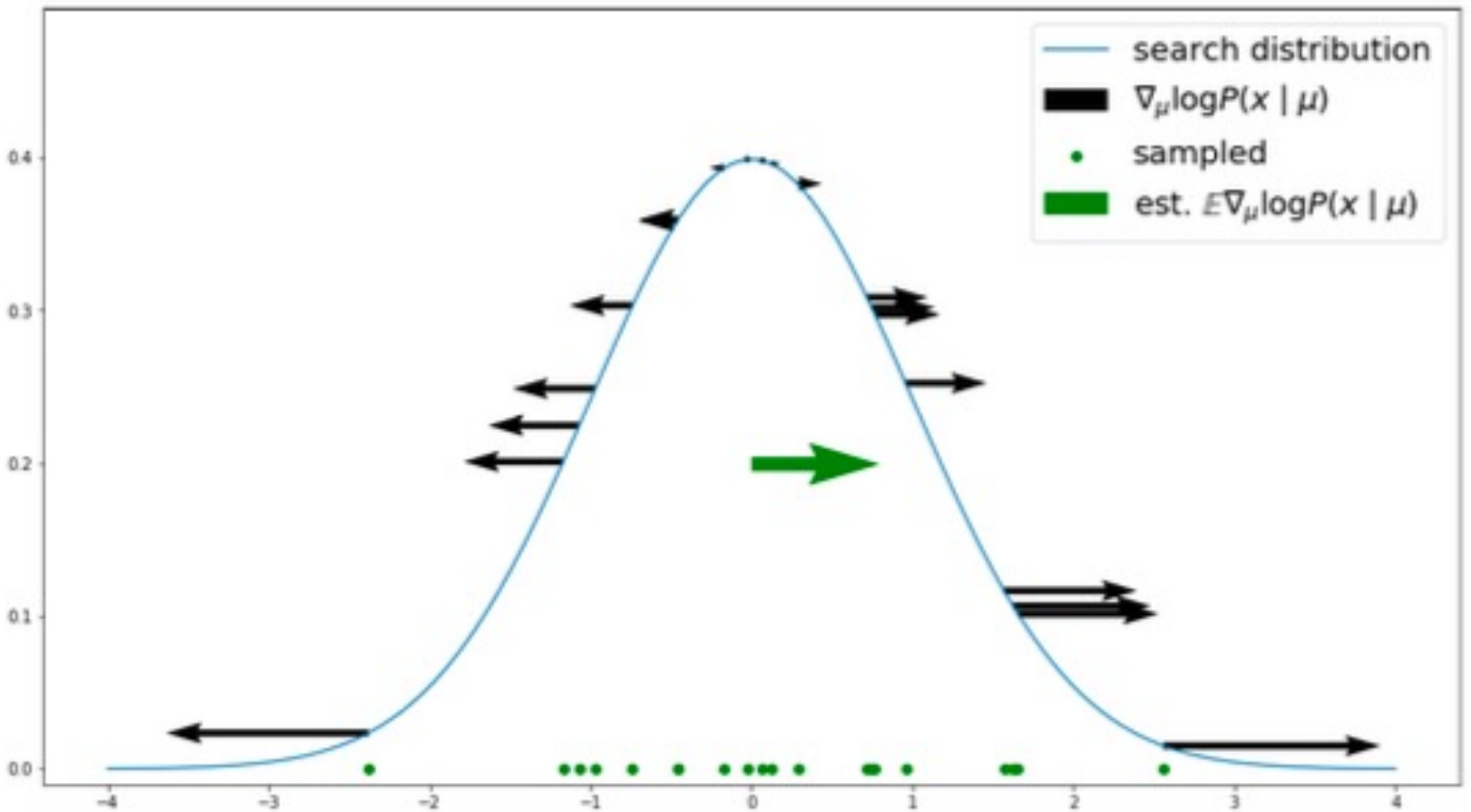


Figure 1: Gradient of $\mathcal{N}(x; \mu, \sigma)$ w.r.t. μ, i.e. $\nabla_\mu \log \mathcal{N}(x; \mu, \sigma)$

Gradient of the variational bound

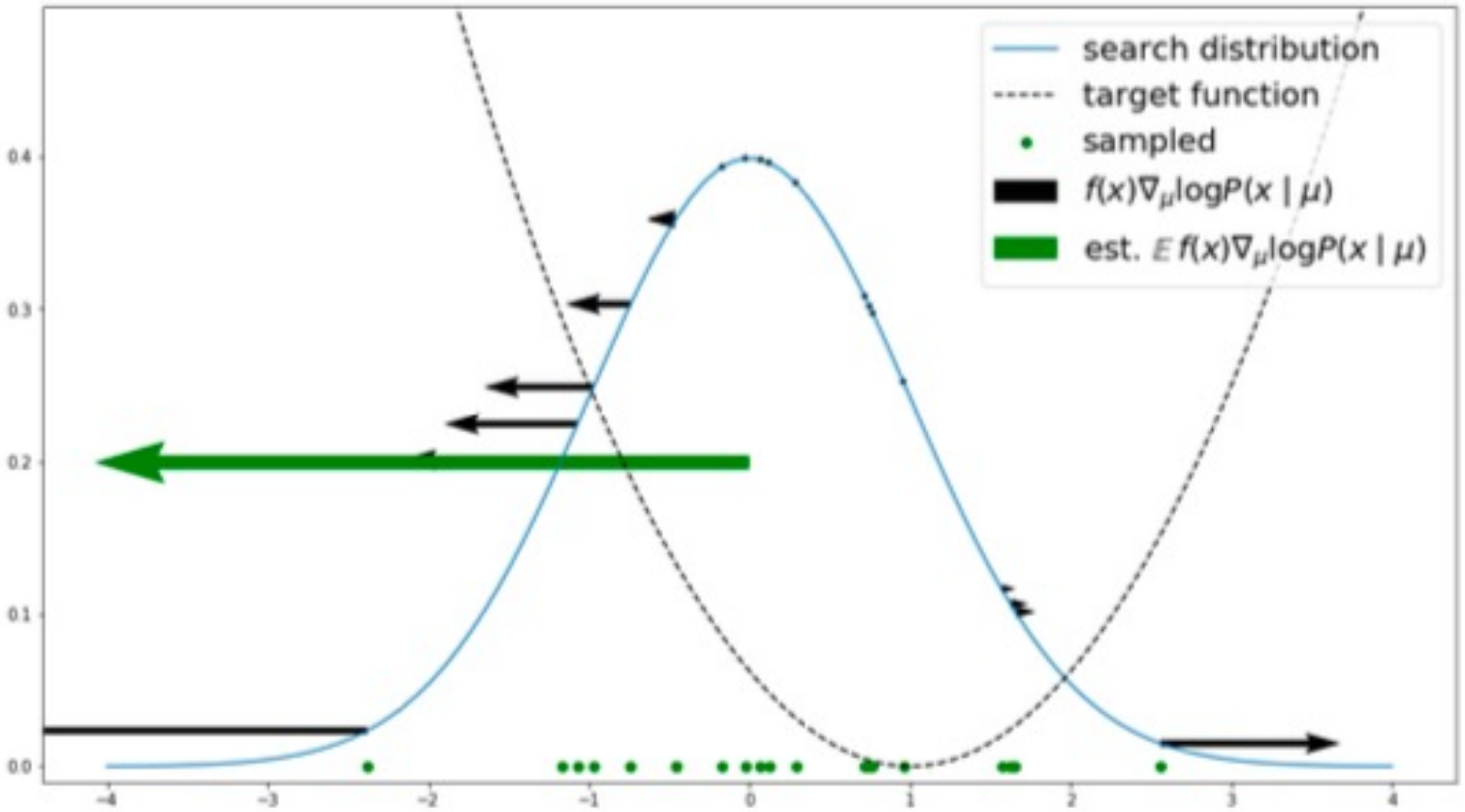


Figure 2: Gradient of variational bound with $\mathcal{N}(\mathrm{x}; \mu, \sigma)$ as search distribution w.r.t. μ, i.e. $\mathrm{f}(\mathrm{x})\nabla_\mu \log \mathcal{N}(\mathrm{x}; \mu, \sigma)$

Stochastic GD Variational Optimization

Algorithm 1 SGD-VO

1: initialize $P(\cdot \mid \psi)$
2: **while** not converged **do**
3: sample θ from $P(\cdot \mid \psi)$;
4: $\nabla_\psi J(\psi) \leftarrow f(\theta)\nabla_\psi \log P(\theta \mid \psi)$;
5: $\psi \leftarrow \psi - \gamma \nabla_\psi J(\psi)$;
6: **end while**

- allows usage of stochastic gradient methods for black-box problems:
  - VO is much slower in contrast to using analytical gradient;
- search distribution is chosen to be simple: – e.g. normal distribution;
- dimensionality of the problem can be retained:
  - at least 1 additional parameter to allow the search distribution to collapse (σ);
  - $O(n^2)$ for a full covariance matrix;
  - O(n) parameters for a normal distribution with diagonal covariance.

Black-Box Optimization with Local Generative Surrogates (L-GSO)

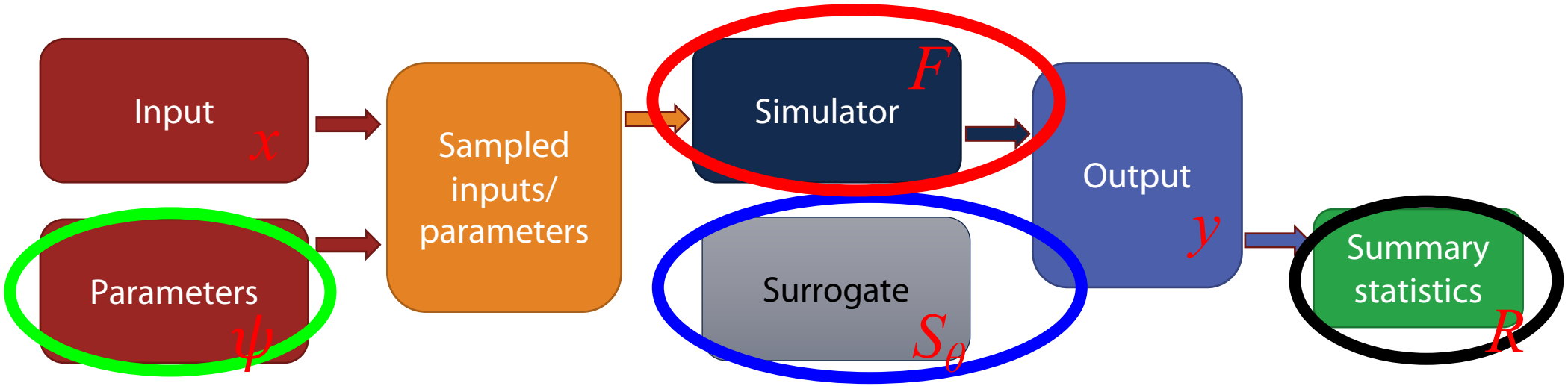
TL;DR:

Let's approximate a stochastic black-box with a local generative surrogate.

This allows computing gradients of the **objective** w.r.t. parameters of the black-box.

$$\mathbb{E}[\mathcal{R}(\boldsymbol{y})] = \int \mathcal{R}(\boldsymbol{y}) p(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\psi}) q(\boldsymbol{x}) d\boldsymbol{x} d\boldsymbol{y} \approx \frac{1}{N} \sum_{i=1}^{N} \mathcal{R}\left(F(\boldsymbol{x}_i;\boldsymbol{\psi})\right)$$

$$\boldsymbol{y}_i = F(\boldsymbol{x}_i;\boldsymbol{\psi}) \sim p(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\psi}),$$
$$\boldsymbol{x}_i \sim q(\boldsymbol{x})$$



$$\nabla_{\boldsymbol{\psi}} \mathbb{E}[\mathcal{R}(\boldsymbol{y})] \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\psi}} \mathcal{R}\left(S_\theta(\boldsymbol{z}_i, \boldsymbol{x}_i;\boldsymbol{\psi})\right)$$

$$\nabla_{\psi}\,\mathbb{E}[\mathcal{R}(\boldsymbol{y})] = \int \nabla_{\psi}\mathcal{R}(\boldsymbol{y})\, p(\boldsymbol{y}|\boldsymbol{x},\psi)\, p(\boldsymbol{x})\, d\boldsymbol{x}\, d\boldsymbol{y} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_{\psi}\mathcal{R}(F(\boldsymbol{x}_i;\psi))$$

From intractable gradient estimation of the black-box.

$$\nabla_{\psi}\,\mathbb{E}[\mathcal{R}(\boldsymbol{y})] \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_{\psi}\mathcal{R}(S_{\theta}(\boldsymbol{z}_i, \boldsymbol{x}_i, \psi))$$

To gradient estimation with learnable generative surrogate(GAN, NF, etc).

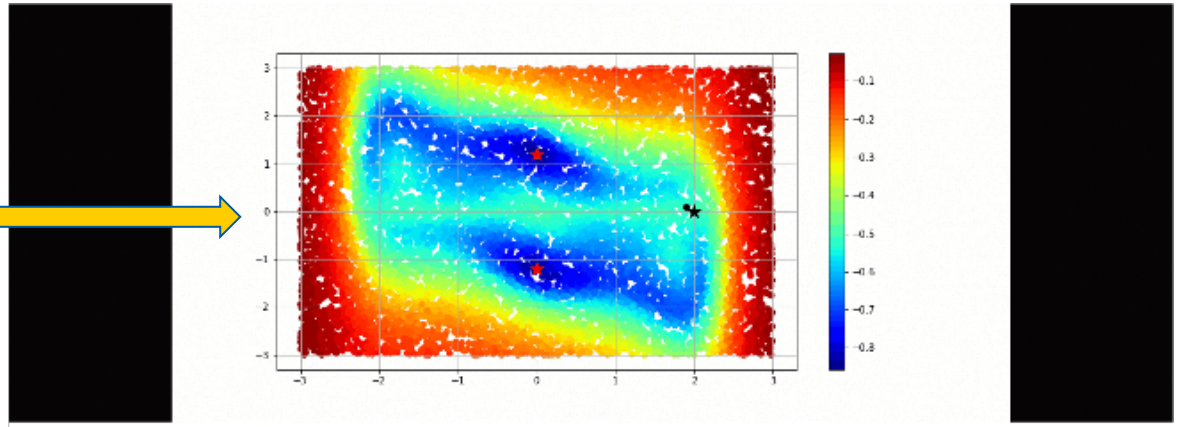
$$\psi = \psi - \mu\frac{1}{N}\sum_{i=1}^{N} \nabla_{\psi}\mathcal{R}(S_{\theta}(\boldsymbol{z}_i, \boldsymbol{x}_i, \psi))$$

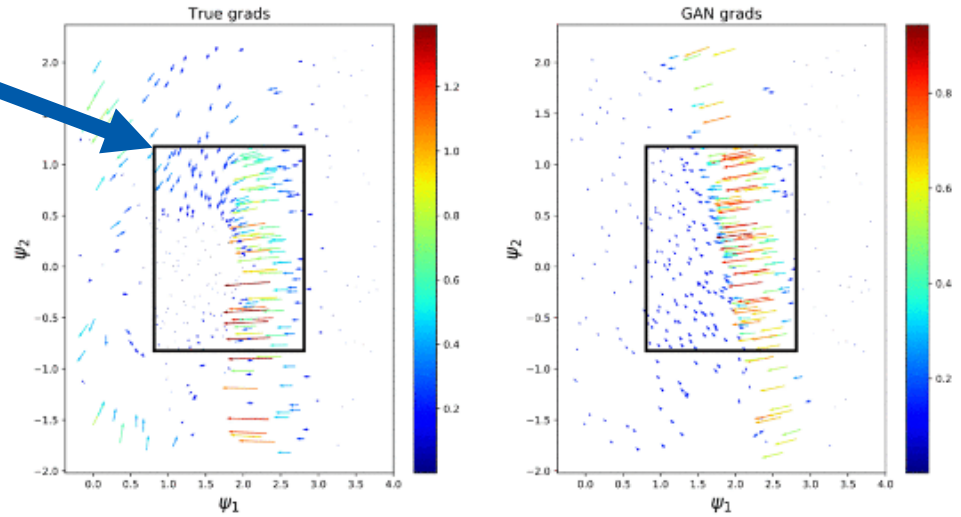
And successive gradient based optimization of the parameters.

Key point: training **local** generative surrogate

Optimization path

Area inside which the local surrogate was trained



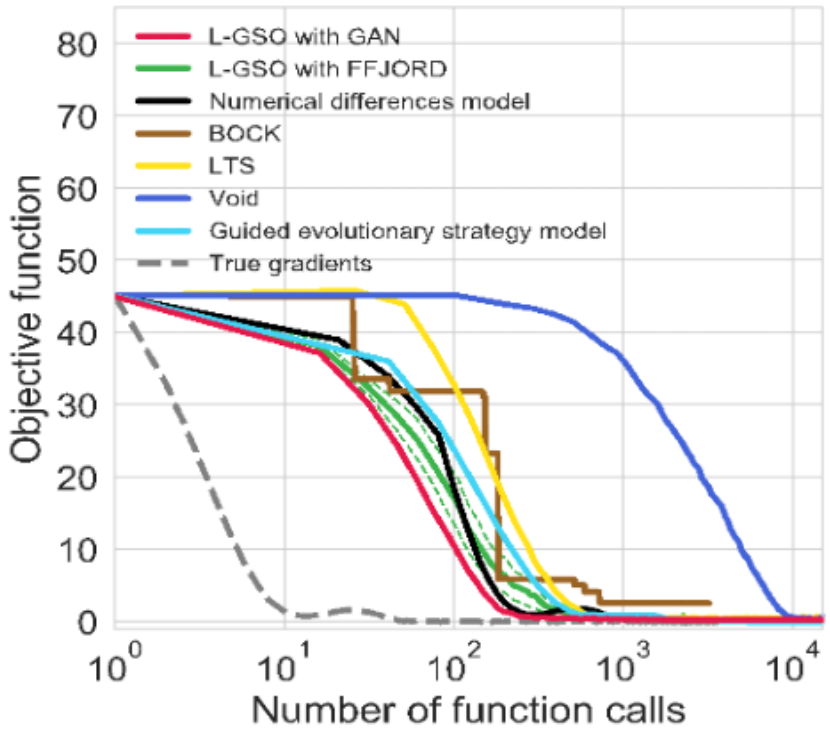


True gradients

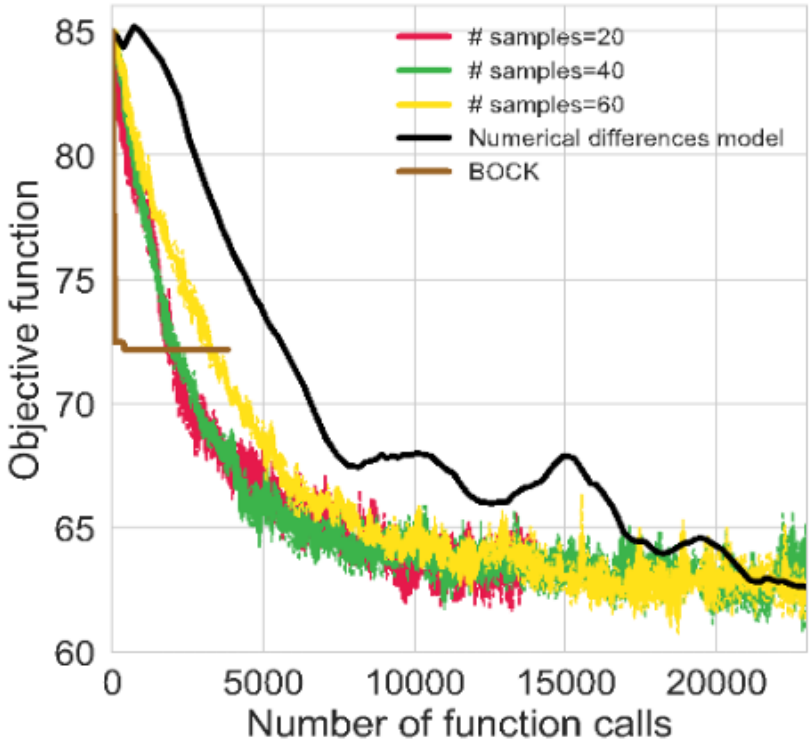
Surrogate gradients

- ✓ gradients of the non-linear surface are well estimated inside the local area.
- ✓ L-GSO outperforms **all** algorithms in a high-dimensional setting when parameters lie on a lower dimension manifold.

Results on high-dimensional problems with low-dimensional manifold



Nonlinear Three Hump problem, 40dim



Neural network weights optimization, 91dim

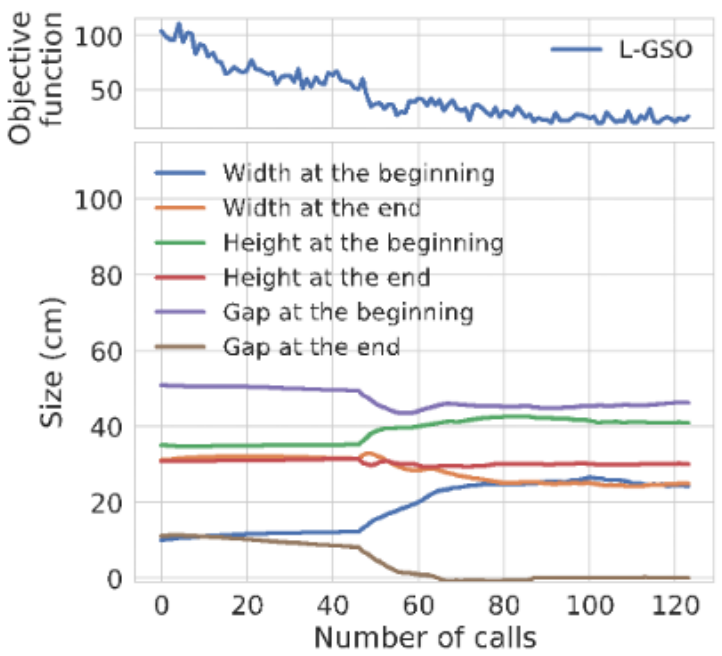
L-GSO is free from explicit variational distribution model

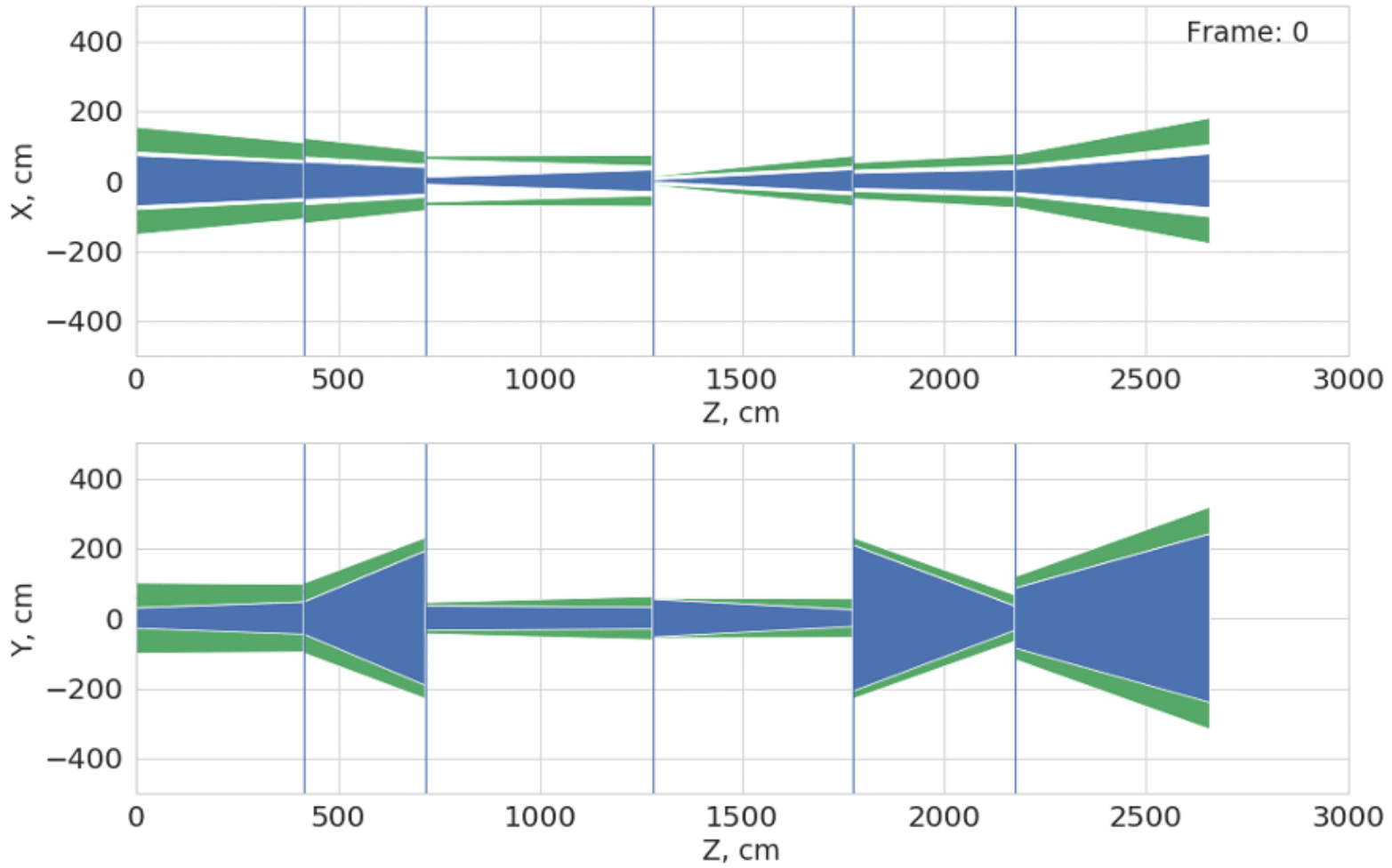
L-GSO outperforms **all** algorithms in a high-dimensional setting with lower dimension manifold.

1. Liu, Shuang, and Kamalika Chaudhuri. "The inductive bias of restricted f-gans." *arXiv preprint arXiv:1809.04542* (2018).

2. Uppal, Ananya, Shashank Singh, and Barnabás Póczos. "Nonparametric density estimation & convergence rates for gans under besov ipm losses." *Advances in Neural Information Processing Systems*. 2019.

Design optimisation in 42 dimensional space of physics simulator





- L-GSO improves previous results obtained with BO with the same computational budget.

- New design is 25% more efficient.

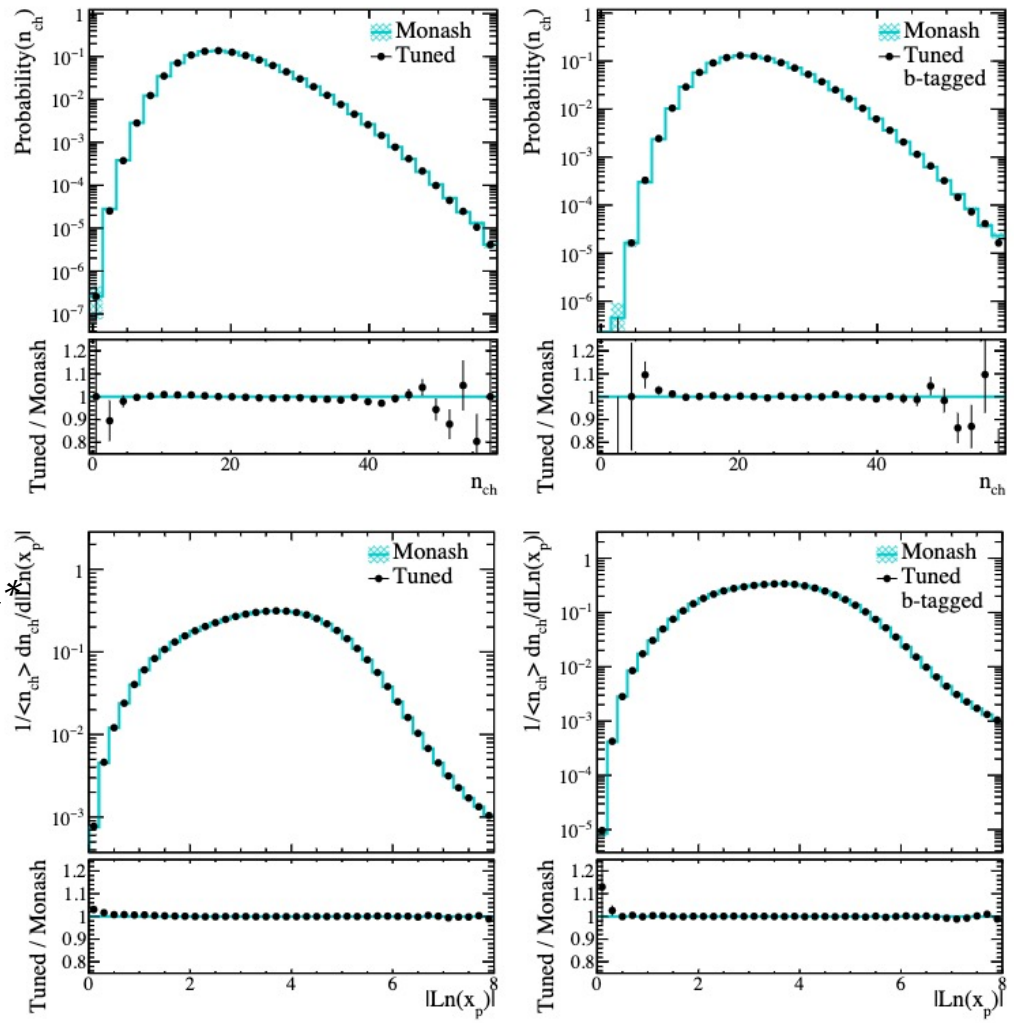
Shirobokov S., Belavin V., Kagan M, AU, Baydin A., NeurIPS'20 paper
https://arxiv.org/abs/2002.04632

Fine-tuning of computer simulations: Adaptive Divergence

Generic Problem statement

- Experiment X produces observations P.
- Computer simulations (Q):
  - computationally demanding;
  - non-differentiable;
- Fine-tuning: find simulation parameters ψ^* such that:

$$Q_{\psi^*} = P \text{ (almost everywhere);}$$



https://bit.ly/3eLtyxl

Existing approaches

- Approximate Bayesian Computations
  - Relies on definition of a summary statistics;
  - Curse of dimensionality;
- Adversarial [Goodfellow et al, 2014], [Louppe, Hermans & Cranmer, 2017]
  - Relies on the underlying classifier model;
  - Requires many samples;

$$\mathrm{JSD}(P, Q_\psi) \to \min_\psi;$$

$$\mathrm{JSD}(P, Q) = \log 2 + \frac{1}{2} \max_{f \in \mathcal{F}} \left[\mathop{\mathbb{E}}_{x \sim P} \log f(x) + \mathop{\mathbb{E}}_{x \sim Q} \log(1 - f(x)) \right]$$

Pseudo-divergence

Definition (pseudo-divergence)

A function $D : \Pi(\mathcal{X}) \times \Pi(\mathcal{X}) \to \mathbb{R}$ is a pseudo-divergence, if:

(P1) $\forall P, Q \in \Pi(\mathcal{X}) : D(P, Q) \geq 0;$

(P2) $\forall P, Q \in \Pi(\mathcal{X}) : (P = Q) \Rightarrow D(P, Q) = 0;$

where $\Pi(\mathcal{X})$ — set of all probability distributions on space $\mathcal{X}$.

$$\text{pJSD}_M(P, Q) = \log 2 + \frac{1}{2} \max_{f \in M} \left[\mathop{\mathbb{E}}_{x \sim P} \log f(x) + \mathop{\mathbb{E}}_{x \sim Q} \log(1 - f(x)) \right]$$

- If M is high-capacity (large), it gives good approximation of JSD but
  - takes large number of samples for estimation;
- If M is low-capacity:
  - small number of samples for estimation but $\exists P \neq Q : \text{pJSD}_M(P, Q) = 0;$

Adaptive divergence (AD)

- Given P and Q:
  - Use low-capacity pseudo-divergence first:

$$\mathrm{pJSD}(P, Q) > 0 \implies \mathrm{JSD}(P, Q) > 0;$$

  - Increase model capacity until it is needed
- Adaptive divergence:

  If a family of pseudo-divergences $D = \{D_\alpha \mid \alpha \in [0, 1]\}$ with respect to Jensen-Shannon divergence, then adaptive divergence AD_D produced by D is defined as:

$$\mathrm{AD}_{\mathcal{D}}(P, Q) = \inf \{D_\alpha(P, Q) \mid \mathrm{D}_\alpha(P, Q) \geq (1 - \alpha) \log 2\}.$$

Boosted AD family

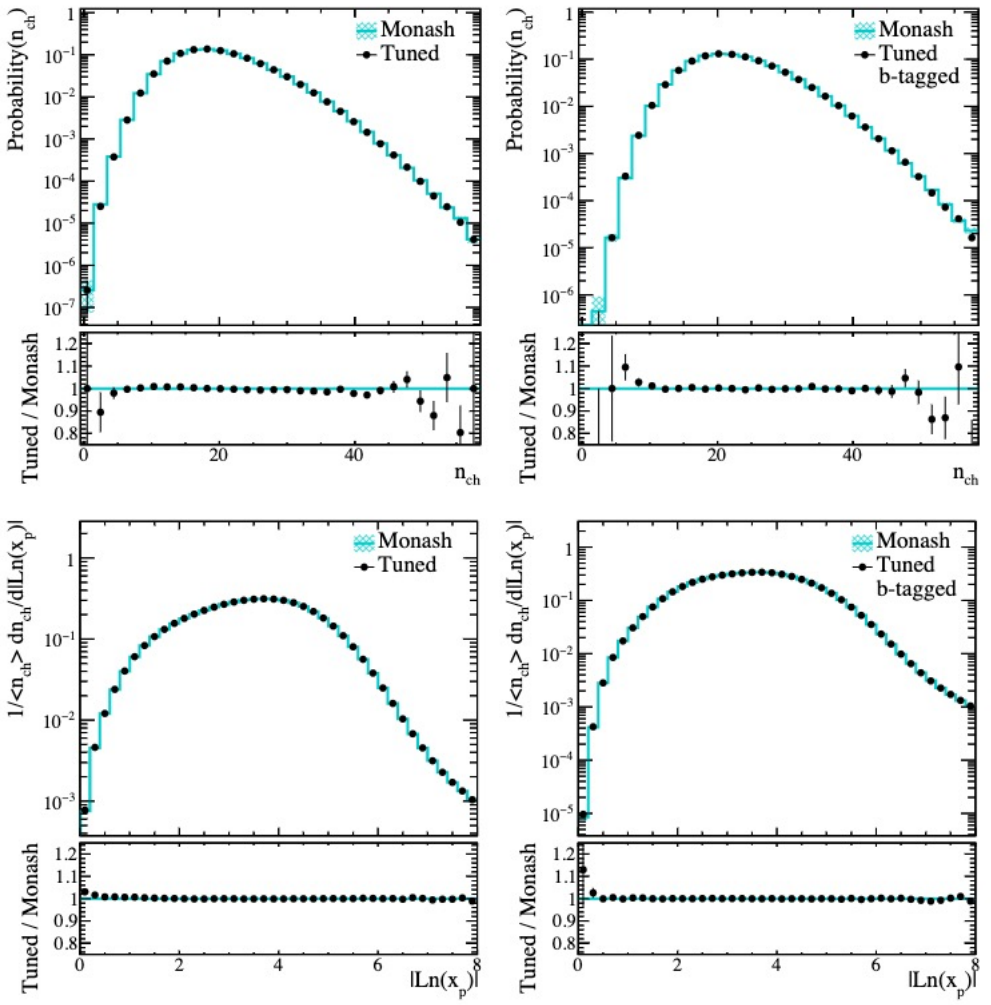
- A boosting-based method is applicable for a discrete approximation:

$$
\begin{aligned}
D_{c(i)}(P, Q) &= \log 2 - L(F_i, P, Q); \\
F_i &= F_{i-1} + \rho \cdot \arg\min_{f \in B} L(F_{i-1} + f, P, Q); \\
F_0 &\equiv \frac{1}{2};
\end{aligned}
$$

- where:
  - B – family of learners (e.g. decision trees);
  - L – cross-entropy loss function;
  - $\boldsymbol{\rho}$ – learning rate;
  - $\mathrm{c(x)}$ – capacity function (strictly increasing), maps number of learners to $\alpha \in [0, 1]$.

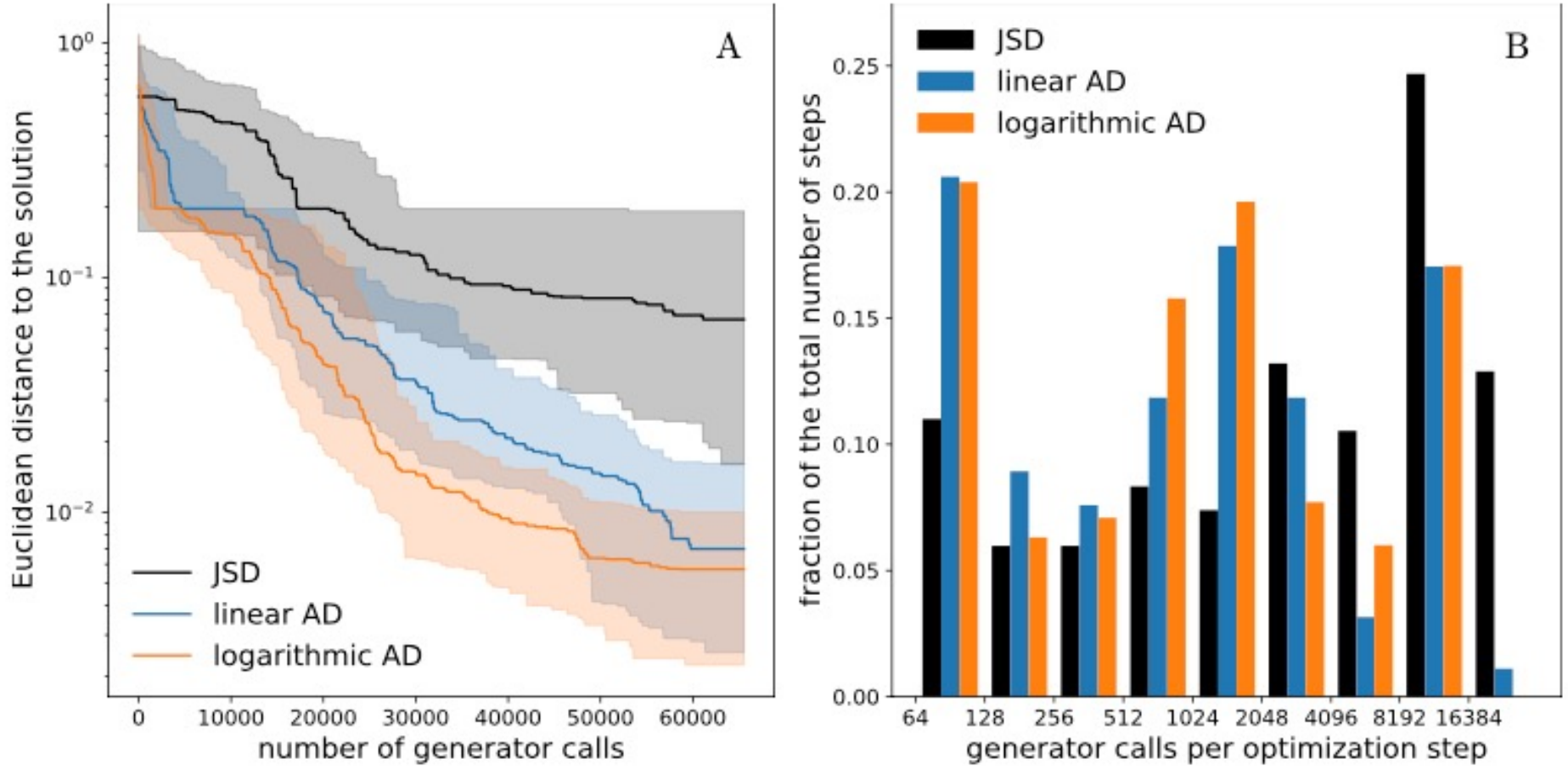
Illustration: Pythia fine-tuning

- electron-positron collisions are simulated at a center-of-mass energy 91.2 GeV.
- A collision event is described by the properties of the final (stable) products. This process is intrinsically stochastic.



https://bit.ly/3eLtyxl

Illustration: Pythia fine-tuning

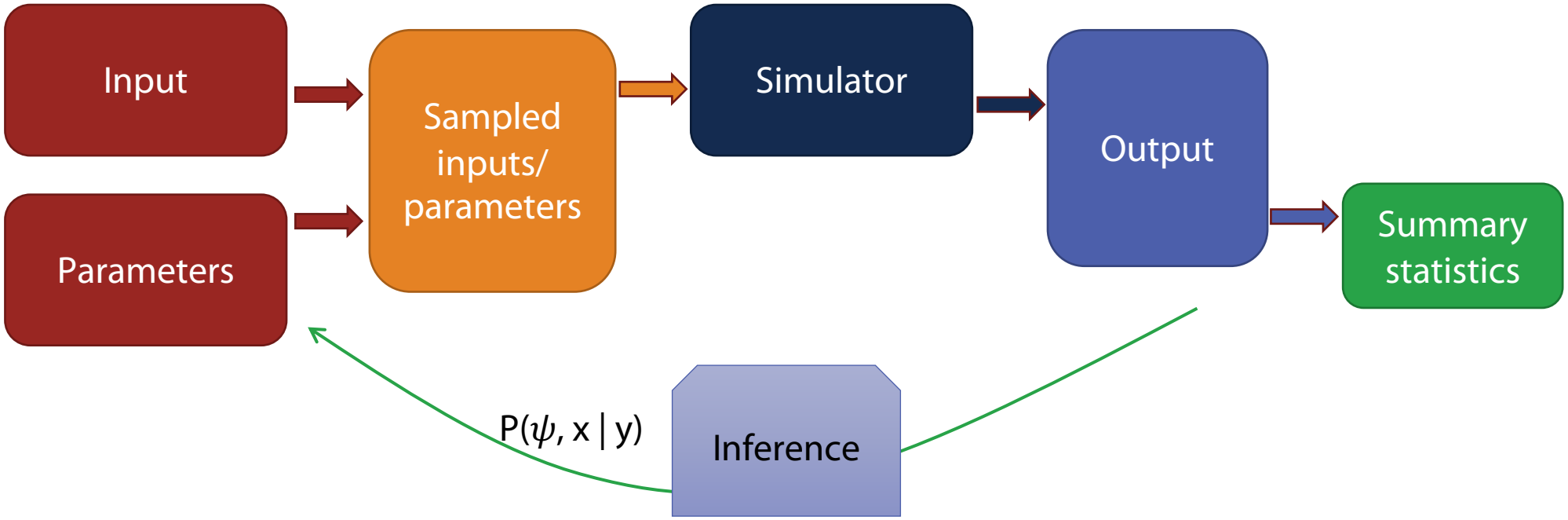


- (A) Convergence of Optimization, (B) Distribution of computational costs per step

Borisyak M, Gaintseva T, UA., Adaptive divergence for rapid adversarial optimization, PeerJ Computer Science 6:e274, 2020, https://doi.org/10.7717/peerj-cs.274

Simulation-based inference for inverse problems

From observations to the laws of nature



How can we get from objective or output to the likelihood of inputs / parameters?

Why?
Simulation defines a computational model of reality we want to learn about.

Challenges:
- reverse implicit non-tractable function defined by simulator
- Simulated/real data matching

Main venues of research:
- Approximate Bayesian Computations
- Active Learning, surrogates
- Probabilistic inference

https://doi.org/10.1073/pnas.1912789117

Fields of future research

- Optimization can be seen as a special case of surrogate-based inference (SBI):
  - (Systematic) uncertainties estimation for surrogate generative models;
  - Increased inference accuracy using simulated and real data mixture;
  - Interpretation of surrogate models.
- Surrogate-based approximation
  - Representation learning;
  - Dynamics of high-level system.

System dynamics learning

- For some fields of research (e.g., molecular dynamics or material design), a target system cannot be represented by static vectors, but by functions instead: i.e., it can exhibit dependence on time or a controllable parameter (temperature, magnetic field, etc.):
  - Not known a priori;
  - Model-free surrogate that can approximate not a mere proposal distribution $p(\theta \mid x, y)$, but a functional dependency between observables and a controllable parameters;
  - Akin to conditional GAN, but for functional dependency.

Conclusion

- [Physical] sciences heavily rely on simulation tools that represent knowledge on the target system in computational form (usually black-boxed);
- Extraction of the patterns from such tools can mean a lot;
- Optimization methods aid such extraction:
  - Surrogate-based methods, L-GSO gives gradient estimation without explicit variational distribution specification;
  - Adaptive exploration helps reducing simulation budget;
- Simulation tools of next generation can and should be integrated with Machine Learning, thus giving plenty of research challenges:
  - Uncertainty, extrapolation, interpretability, functional approximation and so on.

Backup

Boosted AD

Algorithm 2 Boosted adaptive divergence

Require: X_P, X_Q — samples from distributions P and Q, B — base estimator training algorithm, N — maximal size of the ensemble, $c : \mathbb{Z}_+ \to [0, 1]$ — capacity function; ρ — learning rate;

$F_0 \leftarrow 1/2$

$i \leftarrow 0$

$L_0 \leftarrow \log 2$

for $i = 1, \ldots, N$ **do**

  if $L_i > c(i)\log 2$ **then**

    $F_{i+1} \leftarrow F_i + \rho \cdot B(F_i, X_P, X_Q)$

    $L_{i+1} \leftarrow L(F_{i+1}, X_P, X_Q)$

    $i \leftarrow i + 1$

  else

    return $\log 2 - L_i$

  end if

end for

return $\log 2 - L_N$

Neural Network Regularized AD

Algorithm 4 Adaptive divergence estimation by a regularized neural network

Require: X_P, X_Q — samples from distributions P and Q;
$f_\theta : \mathcal{X} \to \mathbb{R}$ — neural network with parameters $\theta \in \Theta$;
$R : \Theta \to \mathbb{R}$ — regularization function; c — capacity function;
ρ — exponential average coefficient;
β — coefficient for R_1 regularization;
γ — learning rate of SGD.

$L_{\text{acc}} \leftarrow \log 2$
while not converged **do**
  $x_P \leftarrow \text{sample}(X_P)$
  $x_Q \leftarrow \text{sample}(X_Q)$
  $\zeta \leftarrow c\left(1 - \frac{L_{\text{acc}}}{\log 2}\right)$
  $g_0 \leftarrow \nabla_\theta \left[L(f_\theta, x_P, x_Q) + \zeta \cdot R(f_\theta)\right]$
  $g_1 \leftarrow \nabla_\theta \|\nabla_\theta f_\theta(x_P)\|^2$
  $L_{\text{acc}} \leftarrow \rho \cdot L_{\text{acc}} + (1-\rho) \cdot L(f_\theta, x_P, x_Q)$
  $\theta \leftarrow \theta - \gamma\left(g_0 + \beta g_1\right)$
end while
return $\log 2 - L(f_\theta, X_P, X_Q)$