Bilevel optimization for machine learning and scientific discovery

Michael Arbel

07 September 2024

Many successes of machine learning

Video generation

W. smith eating spaghetti

RL for controlling fusion reaction

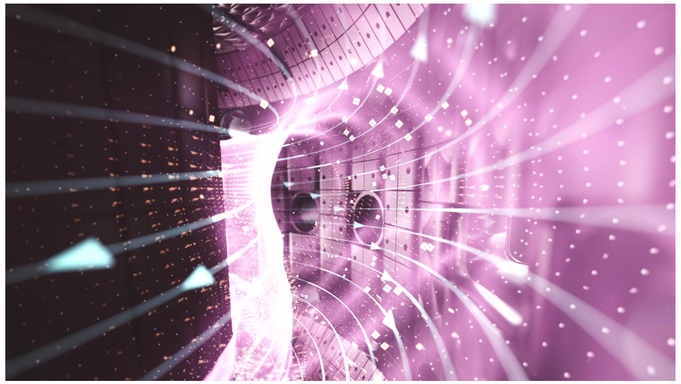
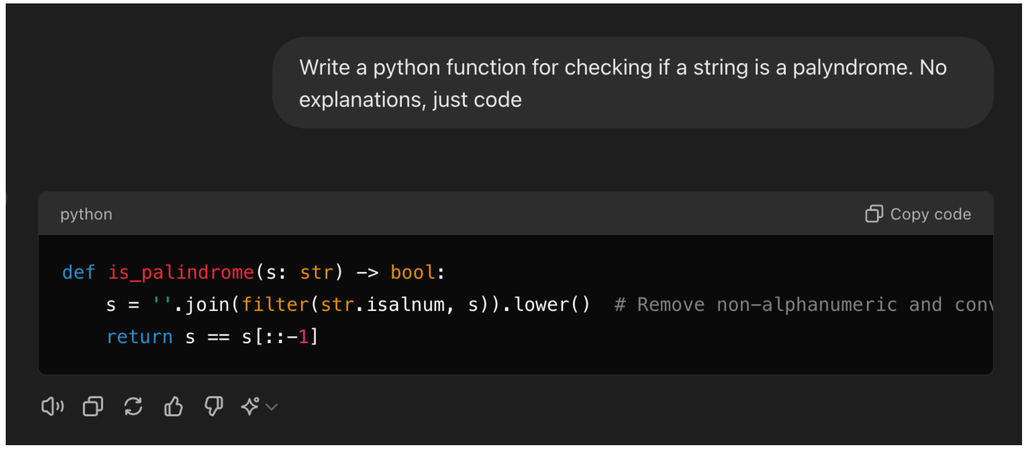


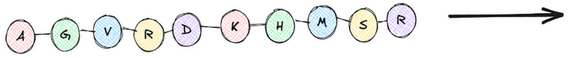
Image generation

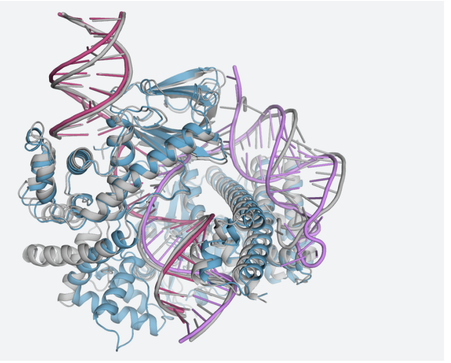
Futuristic Maths & Maroc conference

Code generation with ChatGPT



Protein structure generation with AlphaFold





The workhorse of machine learning

Optimizing huge models using massive data
for prediction tasks by automatic differentiation

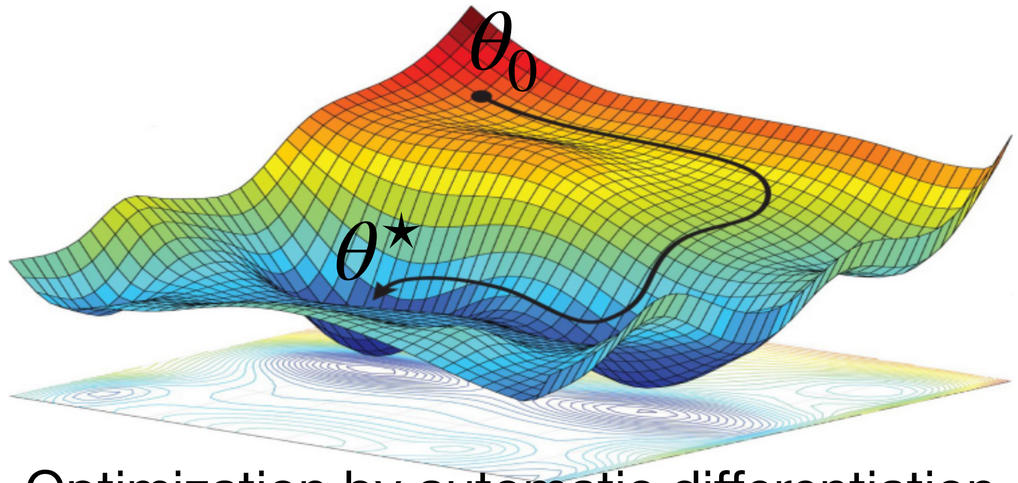
Massive data sets
(essentially all internet)

(x, y)

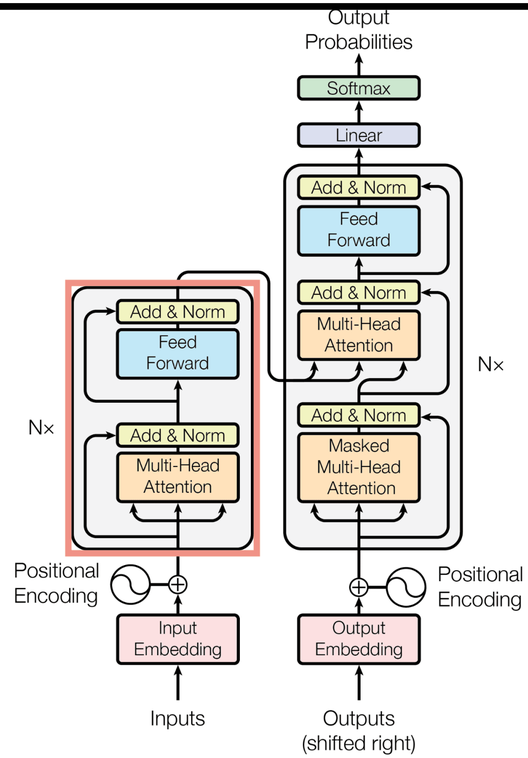
$$\min_{\theta} \mathbb{E}_{(x,y)}[\ell(f_\theta(x), y)]$$

Prediction task

f_θ



Optimization by automatic differentiation



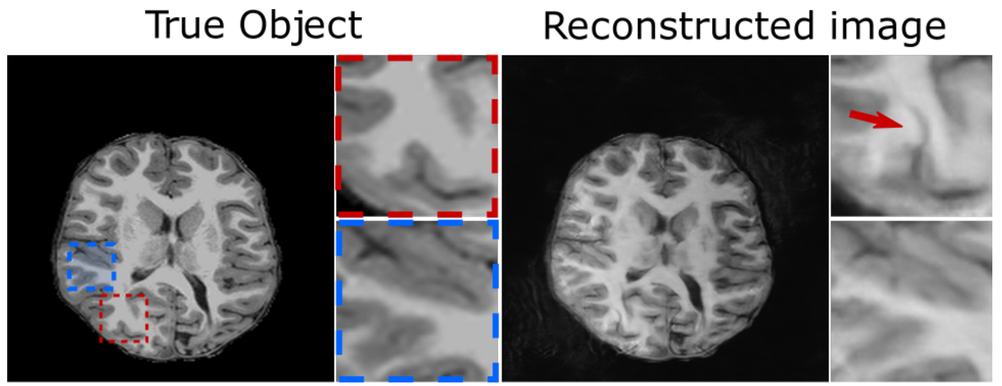
Large flexible models
(Trillion parameters)

Limitations of the current ML paradigm

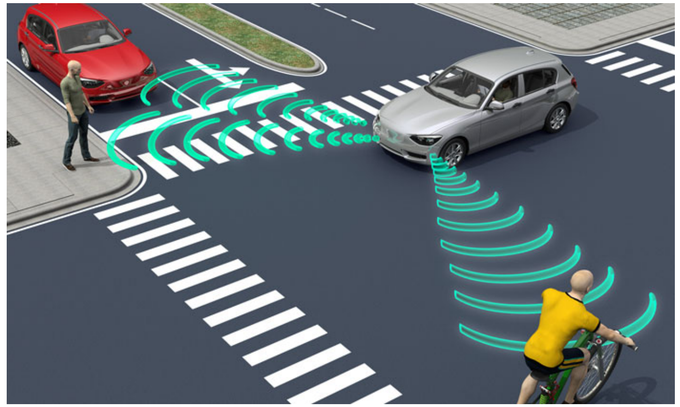
Extremely costly (Several Millions Dollars)

- Large models
- Large data
- Large compute power
- Lots of specialized infrastructure

Unreliable for critical applications



Healthcare



Autonomous driving

Possible fix: leveraging scientific models

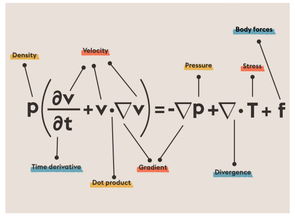
Scientific models

Quantum physics

$$i\hbar\frac{\partial\psi}{\partial t} = -\frac{\hbar^2}{2m}\frac{\|\nabla\psi\|^2}{\psi} + V\psi$$

$$\frac{1}{c^2}(\frac{\partial\psi}{\partial t})^2 - \|\nabla\psi\|^2 + (\frac{mc}{\hbar})^2\psi^2 = 0$$

Navier-Stokes equations



General relativity

$$G_{\mu\nu} = 8\pi G T_{\mu\nu}$$

Einstein's original equation

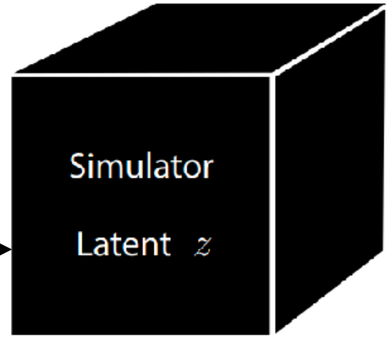
$$G_{\mu\nu} + \Lambda g_{\mu\nu} = 8\pi G T_{\mu\nu}$$

$$G_{\mu\nu} = 8\pi G(T_{\mu\nu} - \bar{\rho}_{DE} g_{\mu\nu})$$

Simulators encode scientific models as algorithms

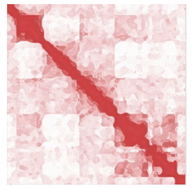
e.g.: Chromatin conformation

Parameter u



Simulation x

e.g.: Contact map



Moderate cost for simulations

Reliable and interpretable predictions

Possible fix: leveraging scientific models

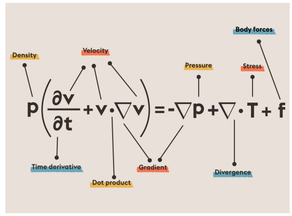
Scientific models

Quantum physics

$$i\hbar\frac{\partial\psi}{\partial t} = -\frac{\hbar^2}{2m}\frac{\|\nabla\psi\|^2}{\psi} + V\psi$$

$$\frac{1}{c^2}(\frac{\partial\psi}{\partial t})^2 - \|\nabla\psi\|^2 + (\frac{mc}{\hbar})^2\psi^2 = 0$$

Navier-Stokes equations



General relativity

$G_{\mu\nu} = 8\pi G T_{\mu\nu}$ Einstein's original equation

Law of an expanding universe — All matter and energy in the universe

$G_{\mu\nu} + \Lambda g_{\mu\nu} = 8\pi G T_{\mu\nu}$

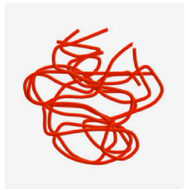
Law of an expanding universe — Cosmological constant — All matter and energy in the universe

$G_{\mu\nu} = 8\pi G(T_{\mu\nu} - \bar{\rho}_{DE} g_{\mu\nu})$

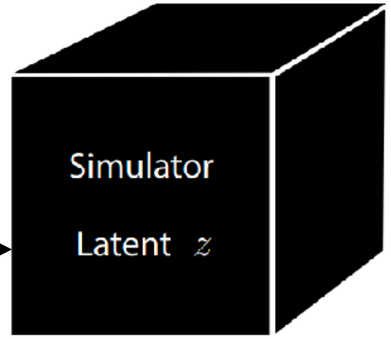
Law of an expanding universe — All matter and energy in the universe

Simulators encode scientific models as algorithms

e.g.: Chromatin conformation

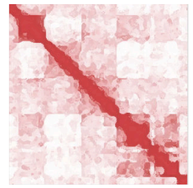


Parameter u



Simulation x

e.g.: Contact map



$u^\star$ ← Simulation-based inference ← Observation x_{obs}

Inference: Gaining knowledge about the world by combining theory (simulator) and experiments (data)

A place for scientific models in ML?

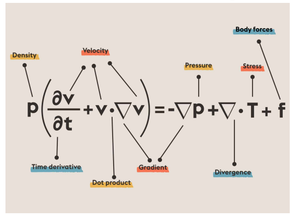
Scientific models

Quantum physics

$$i\hbar\frac{\partial\psi}{\partial t} = -\frac{\hbar^2}{2m}\frac{\|\nabla\psi\|^2}{\psi} + V\psi$$

$$\frac{1}{c^2}(\frac{\partial\psi}{\partial t})^2 - \|\nabla\psi\|^2 + (\frac{mc}{\hbar})^2\psi^2 = 0$$

Navier-Stokes equations



General relativity

$$G_{\mu\nu} = 8\pi G T_{\mu\nu}$$ Einstein's original equation

$$G_{\mu\nu} + \Lambda g_{\mu\nu} = 8\pi G T_{\mu\nu}$$

$$G_{\mu\nu} = 8\pi G(T_{\mu\nu} - \bar{\rho}_{DE} g_{\mu\nu})$$

?

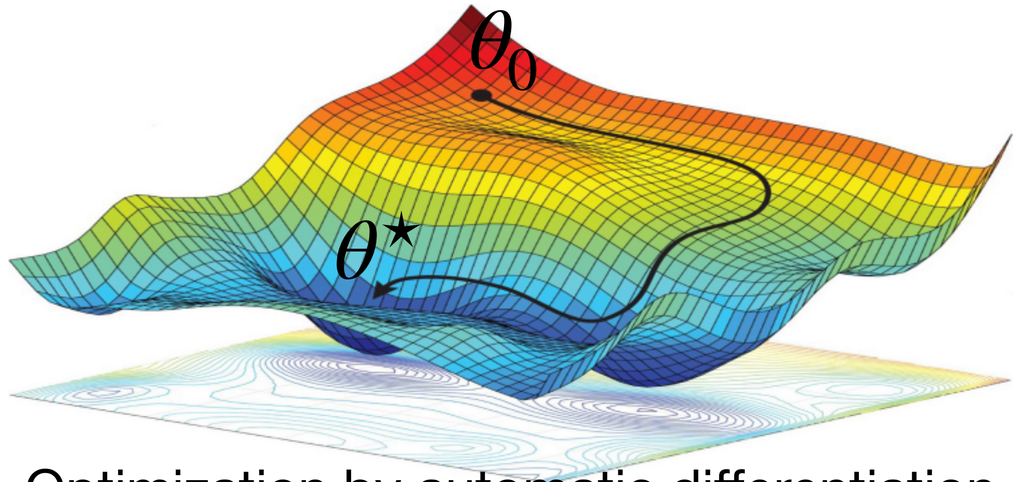
Massive data sets
(essentially all internet)

(x, y)

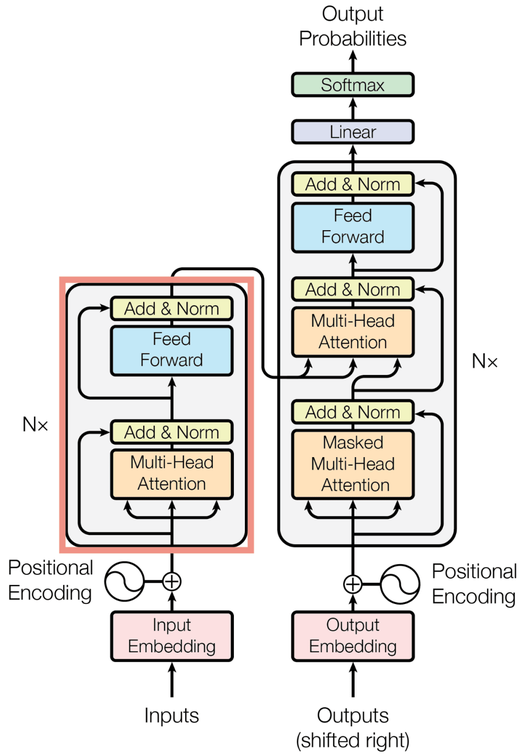
$$\min_\theta \mathbb{E}_{(x,y)}[\ell(f_\theta(x), y)]$$

Prediction task

f_θ



Optimization by automatic differentiation



Large flexible models
(Trillion parameters)

A place for scientific models in ML?

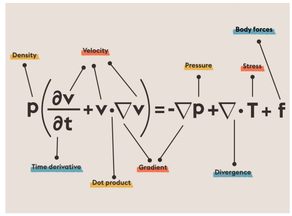
Scientific models

Quantum physics

$$i\hbar\frac{\partial\psi}{\partial t} = -\frac{\hbar^2}{2m}\frac{\|\nabla\psi\|^2}{\psi} + V\psi$$

$$\frac{1}{c^2}(\frac{\partial\psi}{\partial t})^2 - \|\nabla\psi\|^2 + (\frac{mc}{\hbar})^2\psi^2 = 0$$

Navier-Stokes equations



General relativity

$G_{\mu\nu} = 8\pi G T_{\mu\nu}$ Einstein's original equation

$G_{\mu\nu} + \Lambda g_{\mu\nu} = 8\pi G T_{\mu\nu}$

$G_{\mu\nu} = 8\pi G(T_{\mu\nu} - \bar{\rho}_{DE} g_{\mu\nu})$

? Beyond single-level optimization ?

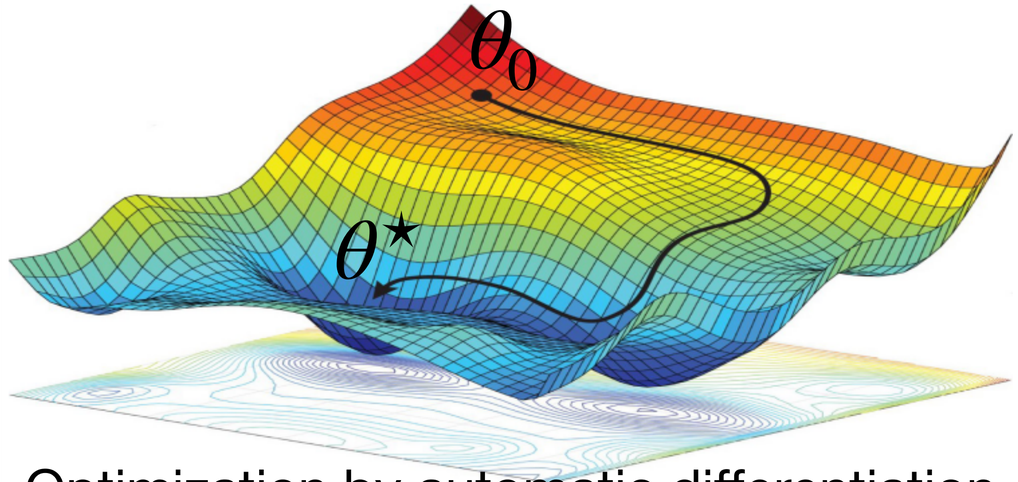
Massive data sets
(essentially all internet)

(x, y)

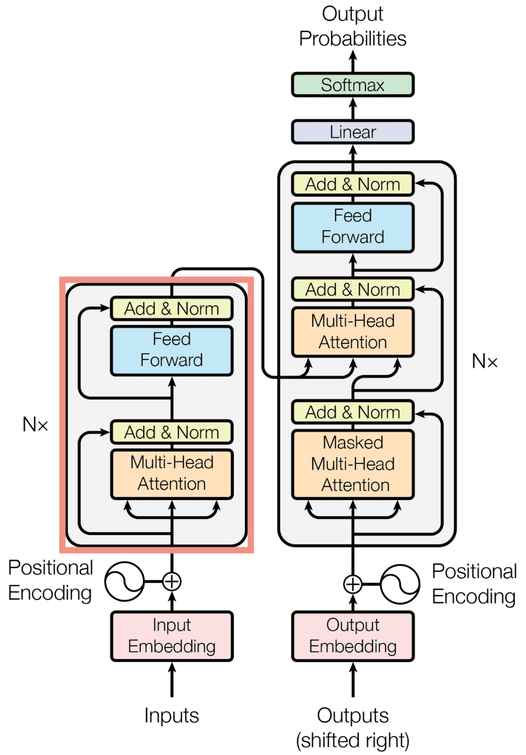
$$\min_\theta \mathbb{E}_{(x,y)}[\ell(f_\theta(x), y)]$$

Prediction task

f_θ



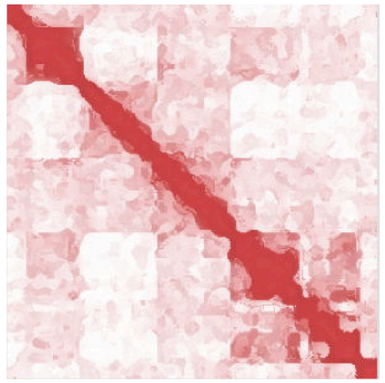
Optimization by automatic differentiation



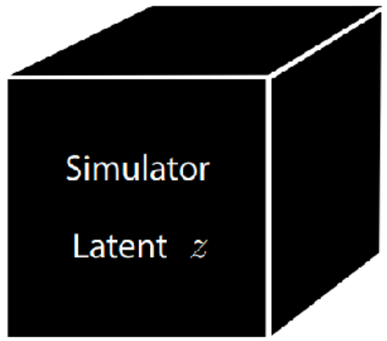
Large flexible models
(Trillion parameters)

BONSAI

Bilevel OptimizatioN for Simulation-bAsed Inference



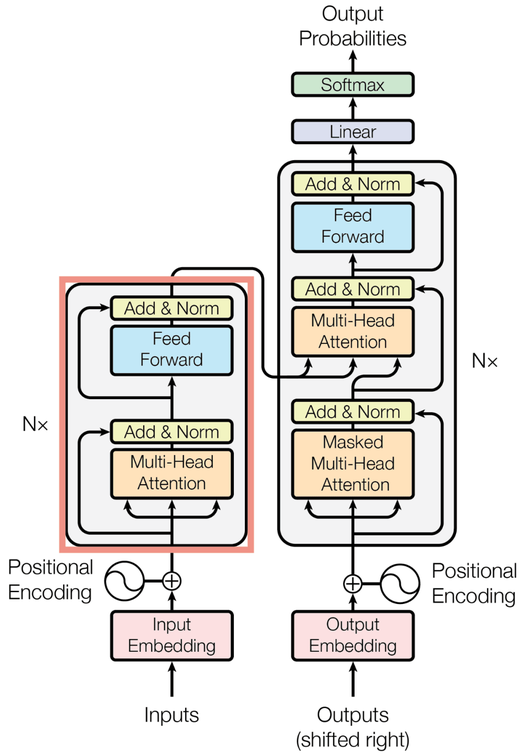
Experimental Data



Simulations

Bilevel optimization

Inference: Gaining knowledge about the world



Large flexible models (Trillion parameters)

BONSAI

Bilevel OptimizatioN for Simulation-bAsed Inference

WP 1: Methods for Bilevel Optimization

Julien Mairal
Inria

Pierre Gaillard
Inria

Diane Larlus
Naver Labs

Juliette Marrie
Inria (PhD student)

Samuel Vaiter
UCA

Edouard Pauwels
TSE

Fares El Khoury
Inria (PhD student)

Ieva Petrulyonite
Inria (PhD student)

WP 2: Surrogate modeling for SBI (via BO)

Florence
Inria

Pedro
Inria

PL. Ruhlmann
Inria (PhD student)

Nelle Varoquaux
TIMC/UGA

Bruno Raffin
Inria

Eloise Touron
Inria (PhD student)

BONSAI

Bilevel OptimizatioN for Simulation-bAsed Inference

WP 1: Methods for Bilevel Optimization

WP 2: Surrogate modeling for SBI (via BO)

Julien Mairal
Inria

Pierre Gaillard
Inria

Diane Larlus
Naver Labs

Juliette Marrie
Inria (PhD student)

Samuel Vaiter
UCA

Edouard Pauwels
TSE

Fares El Khoury
Inria (PhD student)

Ieva Petrulyonite
Inria (PhD student)

Florence
Inria

Pedro
Inria

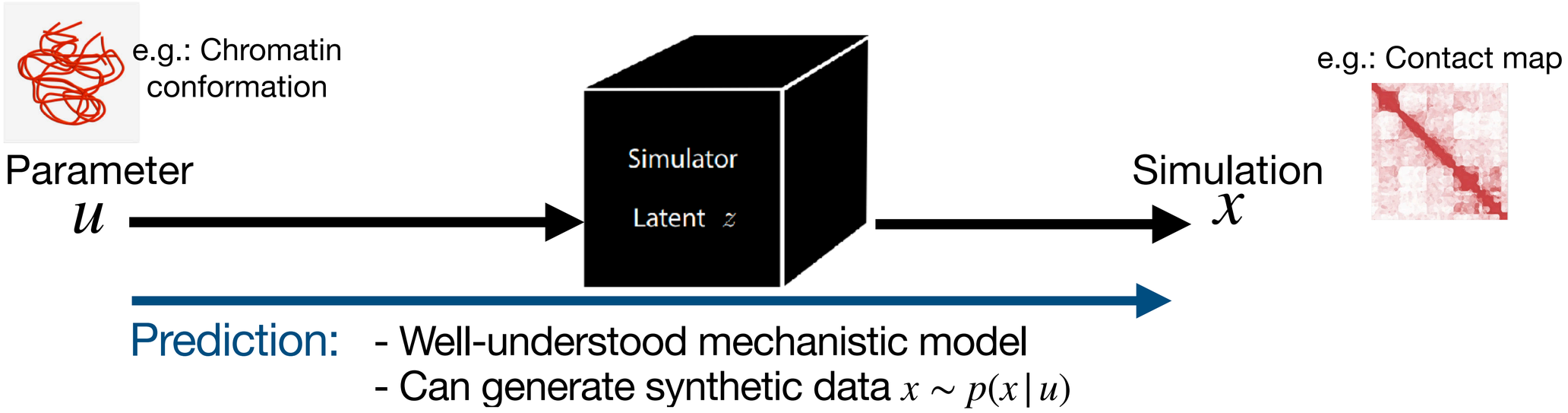
PL. Ruhlmann
Inria (PhD student)

Nelle Varoquaux
TIMC/UGA

Bruno Raffin
Inria

Eloise Touron
Inria (PhD student)

Surrogate modeling for SBI via BO



Prediction: - Well-understood mechanistic model
- Can generate synthetic data $x \sim p(x \mid u)$

Surrogate modeling for SBI via BO

e.g.: Chromatin conformation

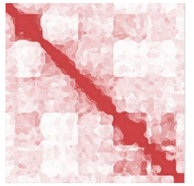
Parameter u

Simulator

Latent z

Simulation x

e.g.: Contact map

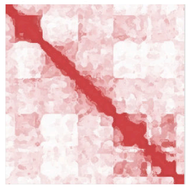


Prediction:
- Well-understood mechanistic model
- Can generate synthetic data $x \sim p(x \mid u)$

$\pi^\star(u)$

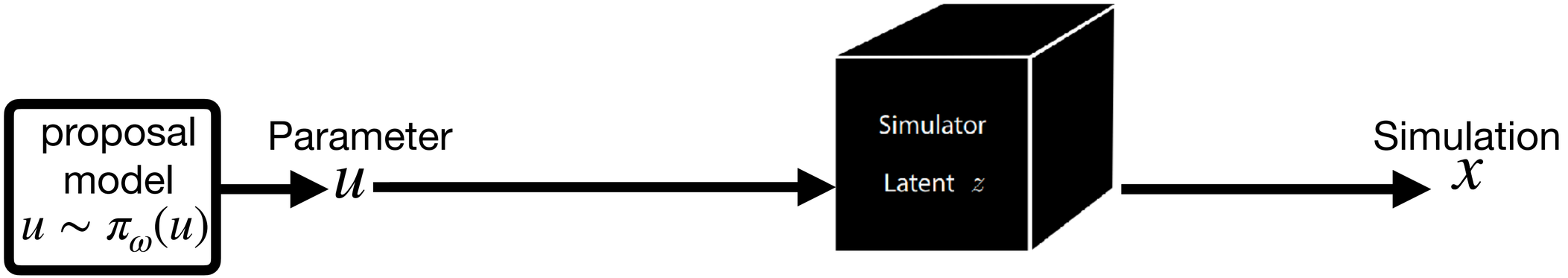
Observation x_{obs}

e.g.: Contact map (sequencing)

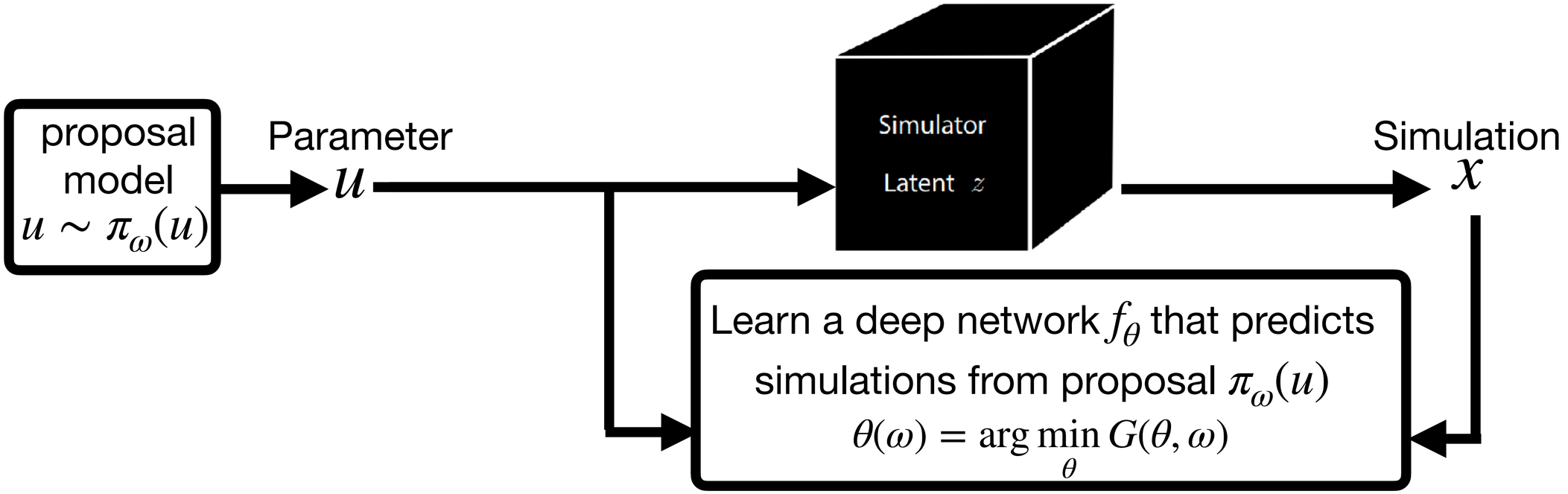


Inference:
- Bayesian inference: $\pi^\star(u) \propto p(x_{obs} \mid u)p(u)$
- Likelihood $p(x_{obs} \mid u)$ is intractable

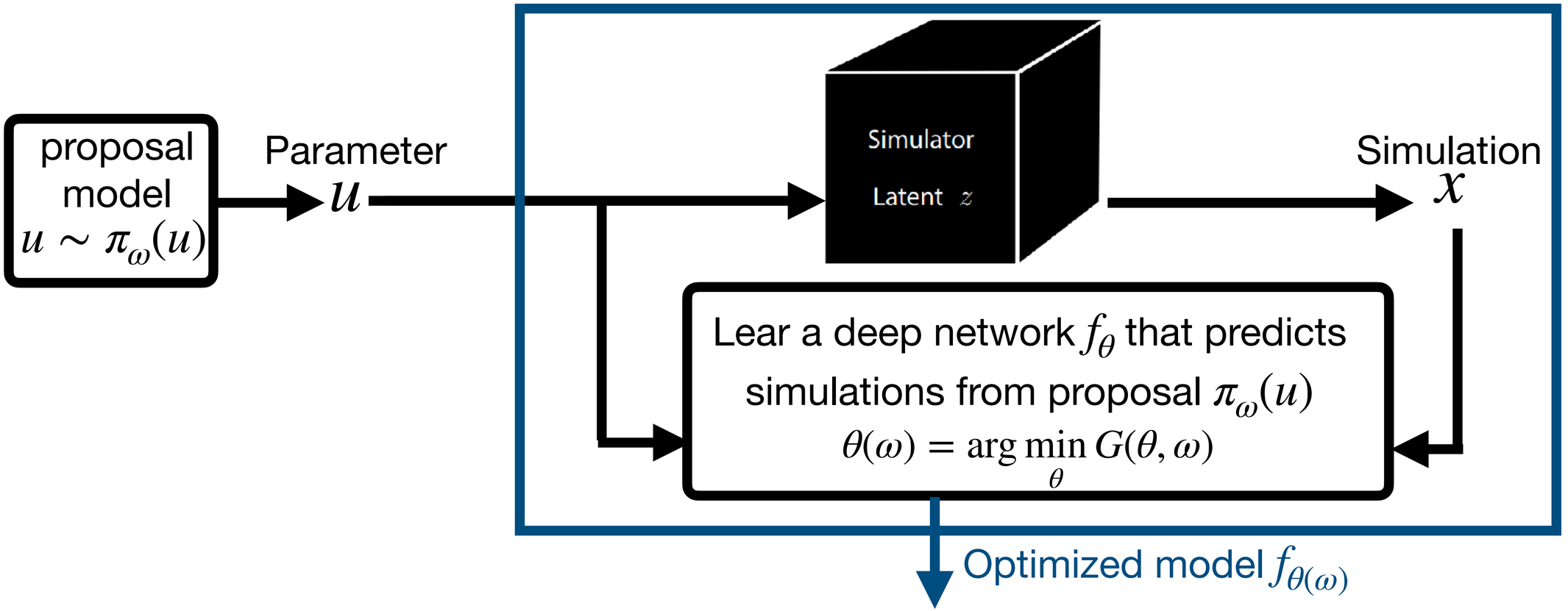
Surrogate modeling for SBI via BO



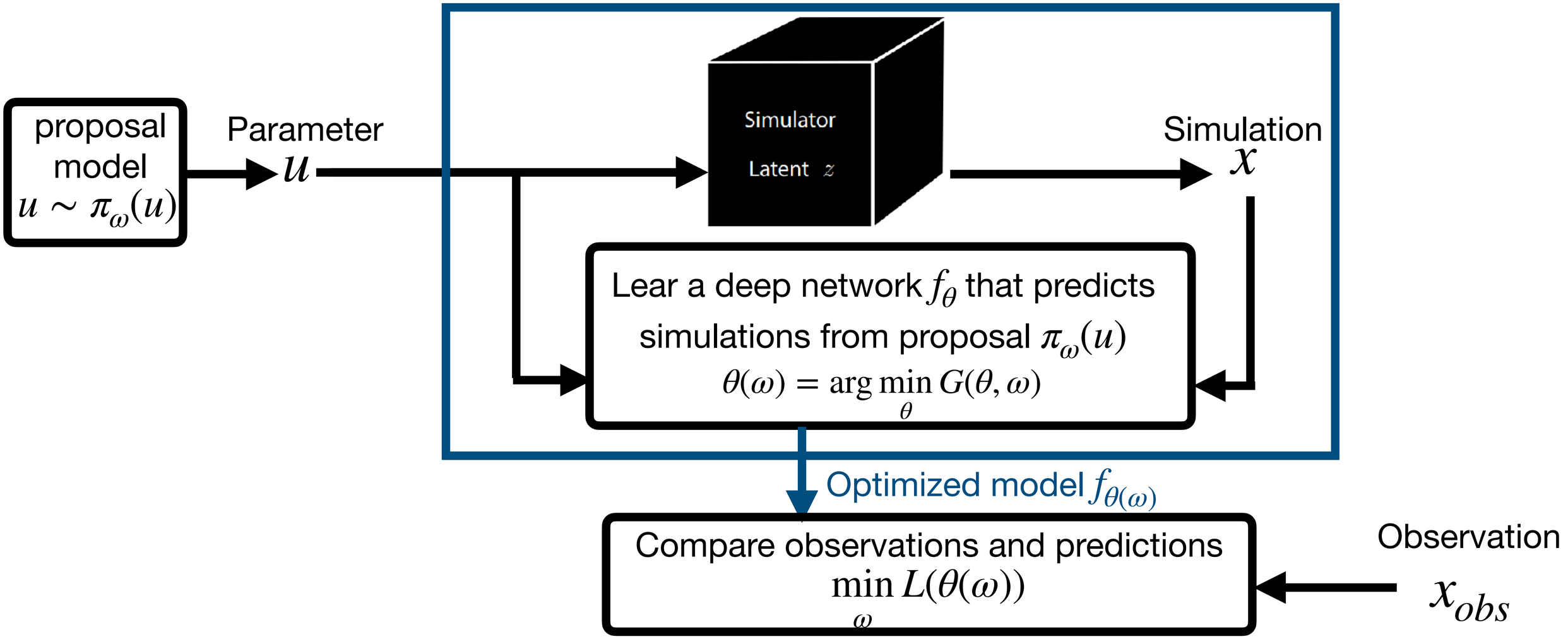
Surrogate modeling for SBI via BO



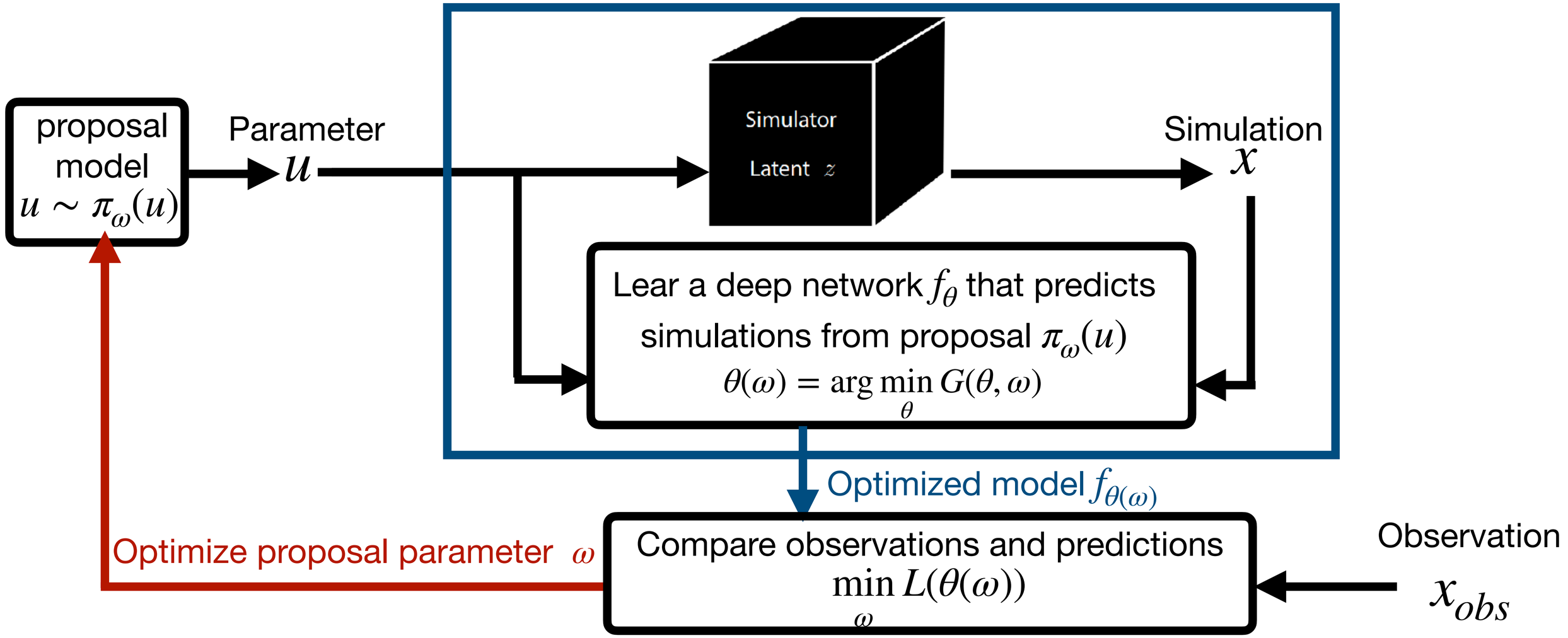
Surrogate modeling for SBI via BO



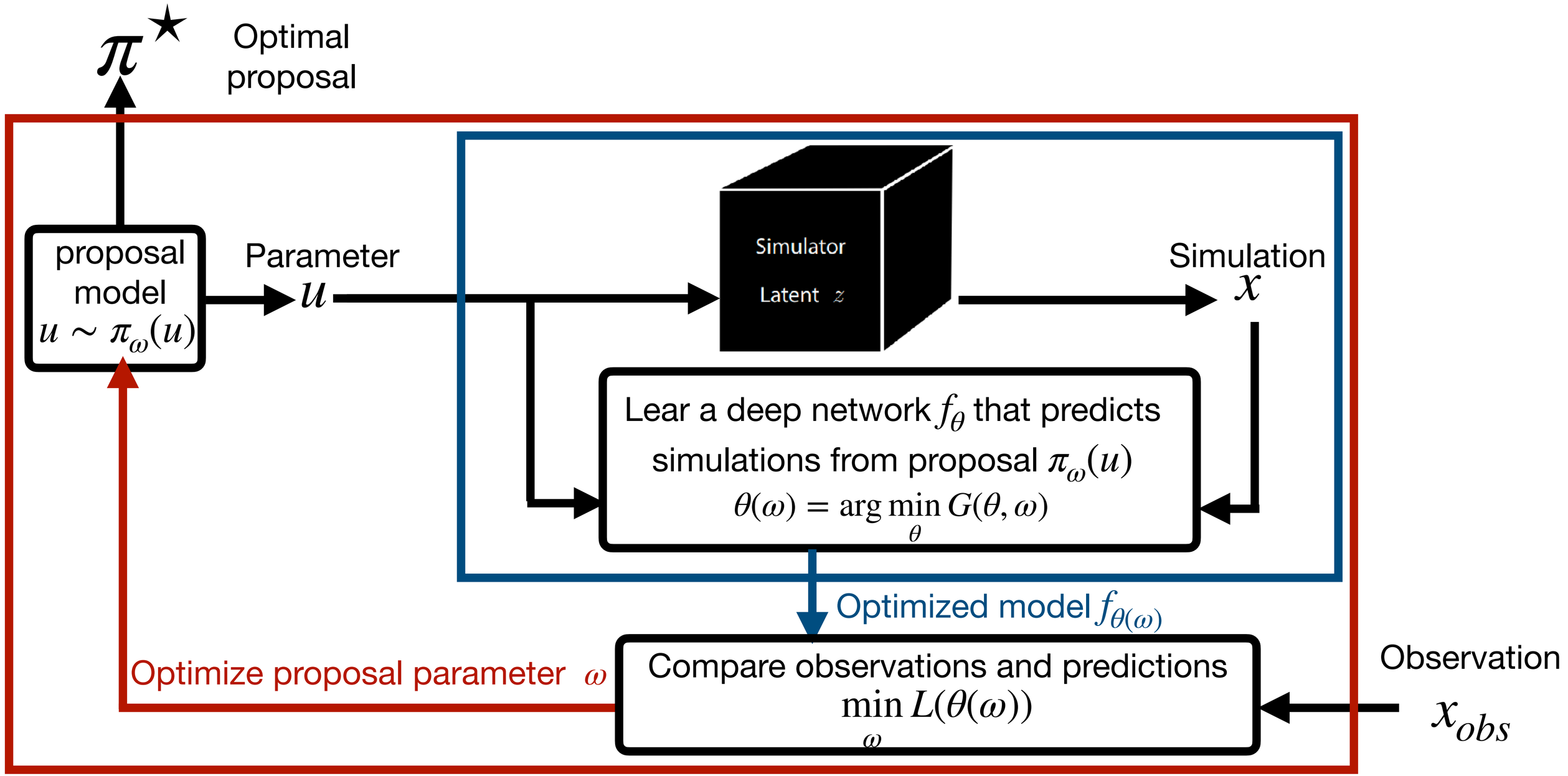
Surrogate modeling for SBI via BO



Surrogate modeling for SBI via BO

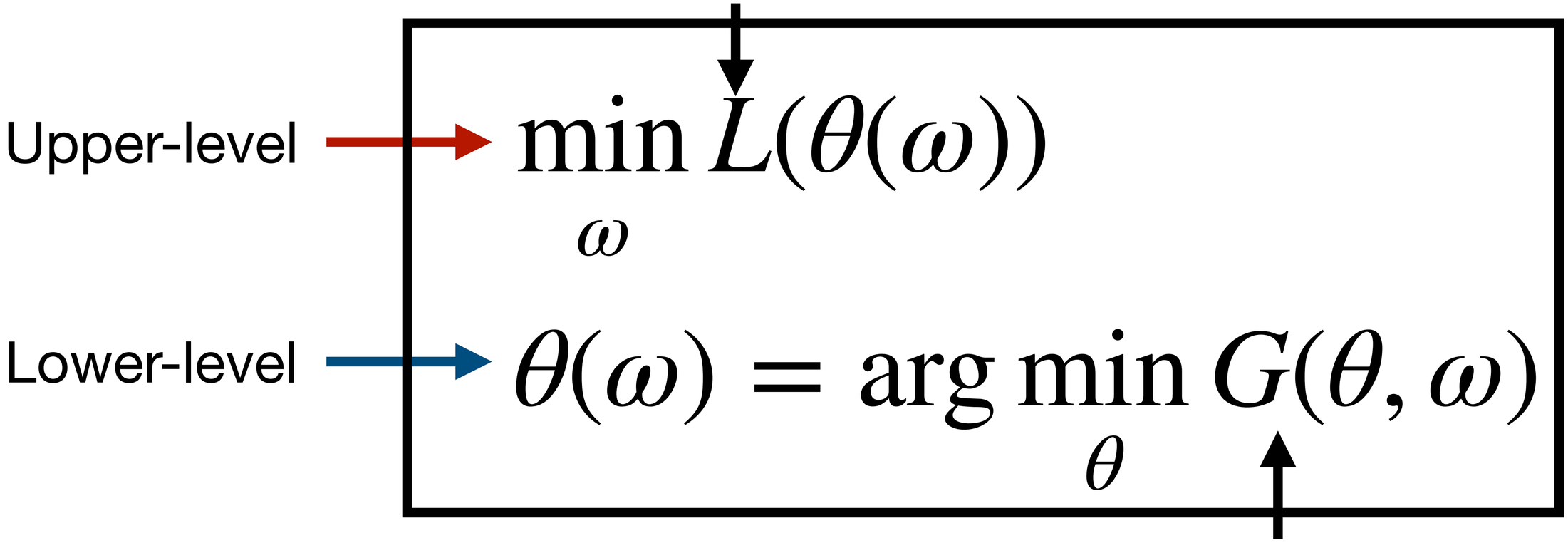


Surrogate modeling for SBI via BO



Surrogate modeling for SBI via BO

Bilevel optimization problem



BONSAI

Bilevel OptimizatioN for Simulation-bAsed Inference

WP 1: Methods for Bilevel Optimization


Julien Mairal
Inria


Pierre Gaillard
Inria


Diane Larlus
Naver Labs


Juliette Marrie
Inria (PhD student)


Samuel Vaiter
UCA


Edouard Pauwels
TSE


Fares El Khoury
Inria (PhD student)


Ieva Petrulyonite
Inria (PhD student)

WP 2: Surrogate modeling for SBI (via BO)


Florence
Inria


Pedro
Inria

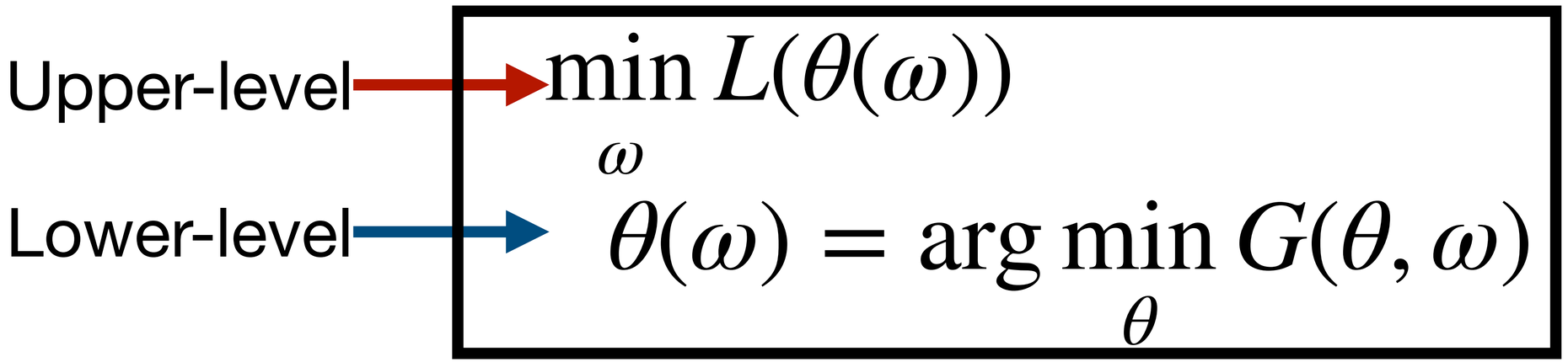

PL. Ruhlmann
Inria (PhD student)


Nelle Varoquaux
TIMC/UGA


Bruno Raffin
Inria


Eloise Touron
Inria (PhD student)

Background on Bilevel Optimization



Introduced in game theory by von Stackelberg (1934).

A very natural formulation for model selection in machine learning:

- θ represents the model parameters, and ω the hyper-parameters.
- $G(\theta, \omega)$ is a regularized empirical risk on training data,
- $L(\theta(\omega))$ measure the fit of the optimal model $\theta(\omega)$ on validation data.

Early occurrences in machine learning

Upper-level → $\min_{\omega} L(\theta(\omega))$

Lower-level → $\theta(\omega) = \arg\min_{\theta} G(\theta, \omega)$

Introduced in machine learning by Bennett et al. (2006):

Model Selection via Bilevel Optimization

Kristin P. Bennett, Jing Hu, Xiaoyun Ji, Gautam Kunapuli, and Jong-Shi Pang

***Abstract*— A key step in many statistical learning methods used in machine learning involves solving a convex optimization problem containing one or more hyper-parameters that must be selected by the users. While cross validation is a commonly employed and widely accepted method for selecting these parameters, its implementation by a grid-search procedure in the parameter space effectively limits the desirable number**

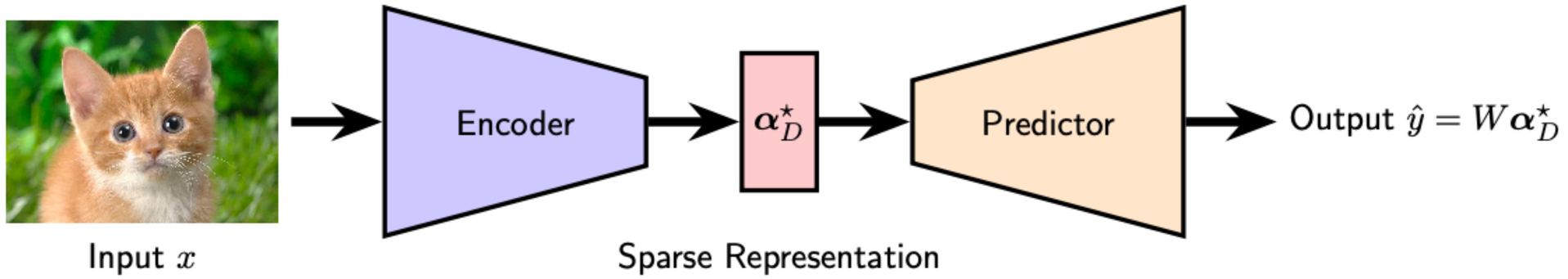
are pervasive in data analysis, e.g., they arise frequently in feature selection [16], [2], kernel construction [19], [22], and multitask learning [4], [10]. For such high-dimensional problems, greedy strategies such as stepwise regression, backward elimination, filter methods, or genetic algorithms are used. Yet, these heuristic methods, including grid search,

Early occurrences in machine learning

Task-driven dictionary learning: Mairal et al. 2010

$$\min_{W,D} \mathbb{E}[\ell(y, W\alpha_D^\star(x))]$$

$$\alpha_D^\star(x) = \arg\min_{\alpha} \frac{1}{2}\|x - D\alpha\|^2 + \lambda\|\alpha\|_1 + \frac{\gamma}{2}\|\alpha\|^2$$



Recent occurrences in machine learning

Generative adversarial networks [Goodfellow et al. 2014]

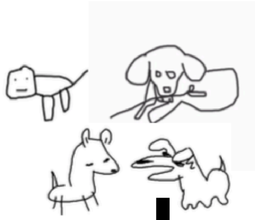
- Generator (student)

- Sample noise

$$Z \sim p(Z)$$

- Pass it to a generator

$$f_\omega(Z)$$



Fake images

Learn features ϕ_θ that best discriminates real from fake images

$$\theta(\omega) = \arg\max_\theta G(\theta, \omega)$$

- Critic (teacher)

ϕ_θ

Real images

Recent occurrences in machine learning

Generative adversarial networks [Goodfellow et al. 2014]

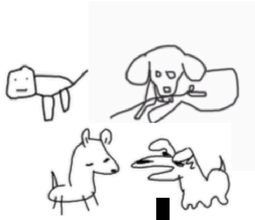
- Generator (student)

- Sample noise

$$Z \sim p(Z)$$

- Pass it to a generator

$$f_\omega(Z)$$



Fake images

Optimize ω so that $\phi_{\theta(\omega)}$ can't easily discriminate real from fake images

Learn features ϕ_θ that best discriminates real from fake images

$$\theta(\omega) = \arg\max_\theta G(\theta, \omega)$$

- Critic (teacher)

ϕ_θ

$\phi_{\theta(\omega)}$

Real images

Recent occurrences in machine learning

Meta learning [Bertinetto et al. 2019]

Setup: T prediction tasks with training and validation sets containing pairs (x, y)

Goal: Predict y from x using a model of the form:

$$\hat{y} = \theta^t h_\omega(x)$$

Shared parameters

Task-dependent parameter

Recent occurrences in machine learning

Meta learning [Bertinetto et al. 2019]

Setup: T prediction tasks with training and validation sets containing pairs (x, y)

Goal: Predict y from x using a model of the form:

$$\hat{y} = \theta^t h_\omega(x)$$

Shared parameters

Task-dependent parameter

Method:

1. Learn optimal weight $\theta^t(\omega)$ for each task t using fixed representation $h_\omega(x)$
2. Minimize prediction error $y - \theta^t(\omega)h_\omega(x)$ w.r.t ω over validation sets

Many other ML applications

- Model-based reinforcement learning [Hong et al. 2023, Nikishin et al. 2022]
- Inverse problems [Holler et al. 2018]
- Invariant risk minimization [Arjovsky et al. 2019, Ahuja et al. 2020]
- Causal inference/Instrumental variable regression [Petrulionyte et al. 2024]

Basic theory from the well-defined (strongly convex) world

Main tool: Implicit differentiation

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) = \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

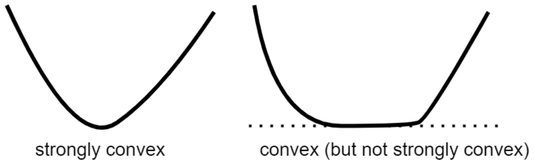
- G is twice differentiable and strongly convex w.r.t. θ
- L is differentiable

Main tool: Implicit differentiation

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) = \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

- G is twice differentiable and strongly convex w.r.t. θ
- L is differentiable



Main tool: Implicit differentiation

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) = \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

- G is twice differentiable and strongly convex w.r.t. θ
- L is differentiable

Computing the derivative of $\mathscr{L}$:

$$\nabla \mathscr{L}(\omega) = \partial_\omega \theta(\omega)^\top \partial_\theta L(\theta(\omega)) \longleftarrow \text{Chain rule}$$

where:

$$\partial_\theta G(\omega, \theta(\omega)) = 0 \longleftarrow \text{Optimality condition}$$

Main tool: Implicit differentiation

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) = \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

- G is twice differentiable and strongly convex w.r.t. θ
- L is differentiable

Computing the derivative of $\mathscr{L}$:

$$\nabla \mathscr{L}(\omega) = \partial_\omega \theta(\omega)^\top \partial_\theta L(\theta(\omega))$$

Implicit differentiation

where:

$$\partial_{\omega,\theta} G(\omega, \theta(\omega)) + \partial_\omega \theta(\omega)^\top \partial^2_\theta G(\omega, \theta(\omega)) = 0$$

Main tool: Implicit differentiation

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) = \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

- G is twice differentiable and strongly convex w.r.t. θ
- L is differentiable

Computing the derivative of $\mathscr{L}$:

$$\nabla \mathscr{L}(\omega) = \partial_{\omega,\theta} G(\omega, \theta(\omega)) a_\omega$$

Adjoint vector

where:

$$a_\omega = -\partial_\theta^2 G(\omega, \theta(\omega))^{-1} \partial_\theta L(\theta(\omega))$$

Questions/Topics

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) = \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Efficiently approximating the gradient $\nabla \mathscr{L}(\omega)$

- Controlling the approximation error, designing approximations: [Ablin et al. 20233, Blondel et al. 2022]
- Optimal convergence rates: [Ghadimi and Wang 2018, Yang et al. 2021, Arbel and Mairal, 2022]
- Variance reduction: [Dagréou et al. 2022]

Non-smooth implicit differentiation: [Blote et al., 2021]

Limitations of the strongly convex setting

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) \in \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

Unrealistic when θ are parameters of a neural networks

- G is twice differentiable and strongly convex w.r.t. θ
- L is differentiable

Dealing with non-convex lower problems

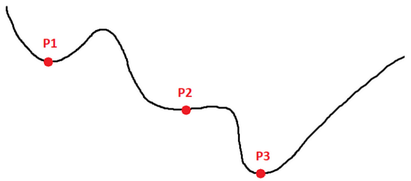
Dealing with non-convex lower problems

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := L(\theta(\omega)) \qquad \theta(\omega) \in \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \omega)$$

Assumption:

More realistic when θ are parameters of a neural networks

- G is twice differentiable and non-convex w.r.t. θ
- L is differentiable



Challenges:

- No uniqueness guarantees for the solution $\theta(\omega)$
- No more implicit function theorem
- Whole problem is ambiguously defined

Methods for non-convex Bilevel Optimization

NeurIPS 2022

Non-Convex Bilevel Games with Critical Point Selection Maps

Michael Arbel and Julien Mairal
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
firstname.lastname@inria.fr

Under review

FUNCTIONAL BILEVEL OPTIMIZATION FOR MACHINE LEARNING

Ieva Petrulionyte, Julien Mairal, Michael Arbel
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
firstname.lastname@inria.fr

Methods for non-convex Bilevel Optimization

NeurIPS 2022

Non-Convex Bilevel Games with Critical Point Selection Maps

Michael Arbel and Julien Mairal
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
firstname.lastname@inria.fr

Under review

FUNCTIONAL BILEVEL OPTIMIZATION FOR MACHINE LEARNING

Ieva Petrulionyte, Julien Mairal, Michael Arbel
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
firstname.lastname@inria.fr

Typical bilevel optimization in ML

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := \mathbb{E}[\ell(h_{\theta(\omega)}(x), y)]$$

$$\theta(\omega) \in \min_{\theta \in \mathbb{R}^p} \mathbb{E}[g(\omega, h_\theta(x), y)]$$

- Lower problem requires optimizing a neural network $h_\theta(x)$ with parameters θ
- g is a strongly convex function in its second argument
- Upper problem only needs to evaluate $h_\theta(x)$

Typical bilevel optimization in ML

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := \mathbb{E}[\ell(h_{\theta(\omega)}(x), y)]$$

$$\theta(\omega) \in \min_{\theta \in \mathbb{R}^p} \mathbb{E}[g(\omega, h_\theta(x), y)]$$

Functional point of view: This is only an approximation to a more general problem

- Lower problem requires optimizing a neural network $h_\theta(x)$ with parameters θ
- g is a strongly convex function in its second argument
- Upper problem only needs to evaluate $h_\theta(x)$

Functional bilevel optimization

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := \mathbb{E}[\ell(h(\omega)(x), y)]$$

$$h(\omega) = \arg\min_{h \in \mathscr{H}} \mathbb{E}[g(\omega, h(x), y)]$$

Functional point of view: Lower optimization happens in a Hilbert space $\mathscr{H}$

Functional bilevel optimization

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := \mathbb{E}[\ell(h(\omega)(x), y)]$$

$$h(\omega) = \arg\min_{h \in \mathscr{H}} \mathbb{E}[g(\omega, h(x), y)]$$

Functional point of view: Lower optimization happens in a Hilbert space $\mathscr{H}$

- Strong convexity w.r.t. to h is a mild assumption, ex:

$$h(\omega) = \arg\min_{h \in \mathscr{H}} \mathbb{E}[\|y - h(x)\|^2] + \omega \|h\|^2_{\mathscr{H}}$$

- No more ambiguity to define $h(\omega)$

- Compatible with deep neural networks used for function approximation

Functional bilevel optimization

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := \mathbb{E}[\ell(h(\omega)(x), y)]$$

$$h(\omega) = \arg\min_{h \in \mathscr{H}} \mathbb{E}[g(\omega, h(x), y)]$$

Functional point of view: Lower optimization happens in a Hilbert space $\mathscr{H}$

Challenges:

- Need to develop a theory and algorithms for Functional Bilevel Optimization
- Implicit differentiation in infinite dimensions is tricky!

Functional bilevel optimization

$$\min_{\omega \in \mathbb{R}^d} \mathscr{L}(\omega) := \mathbb{E}[\ell(h(\omega)(x), y)]$$

$$h(\omega) = \arg\min_{h \in \mathscr{H}} \mathbb{E}[g(\omega, h(x), y)]$$

Functional implicit differentiation in L_2 spaces: [Petrulionyte et al. 2024]

Adjoint function

$$\nabla \mathscr{L}(\omega) = \mathbb{E}[\partial_{1,2} g(\omega, h(\omega)(x), y) a_\omega(x)]$$

Strongly convex in a

where:

$$a_\omega = \arg\min_{a \in \mathscr{H}} \mathbb{E}[\ell_{adj}(\omega, a(x), y)]$$

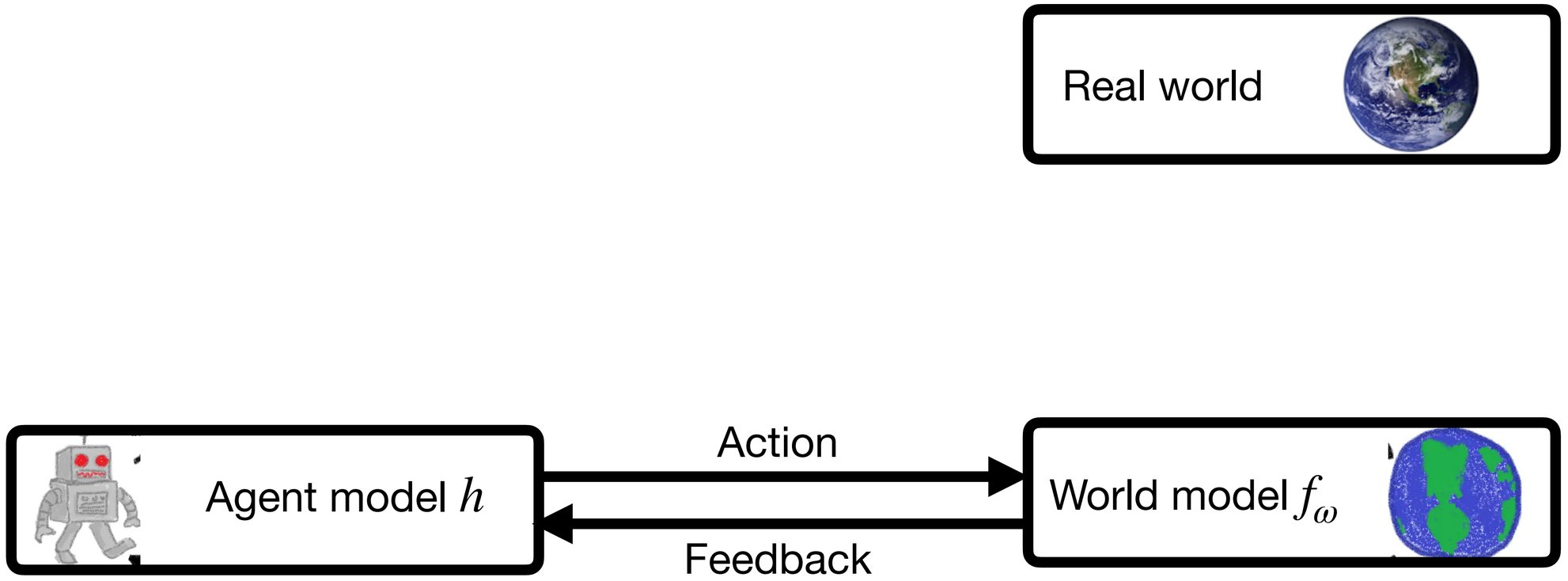
Can use neural networks to approximate both h and a

Functional bilevel optimization: The algorithm

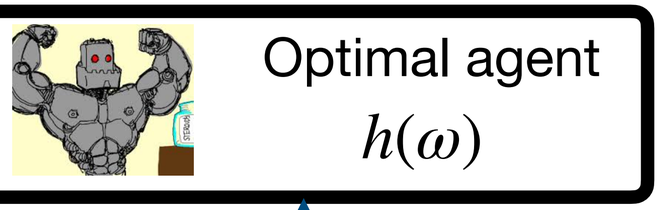
Algorithm 1 ***FuncID***

Input: initial outer, inner, and adjoint parameter ω_0, θ_0, ξ_0; warm-start option `WS`.
for $n = 0, \ldots, N-1$ **do**
 # Optional warm-start
if WS=True **then** $(\theta_0, \xi_0) \leftarrow (\theta_n, \xi_n)$ **end if**
 # Inner-level optimization
$\hat{h}_{\omega_n}, \theta_{n+1} \leftarrow \texttt{InnerOpt}(\omega_n, \theta_0, \mathcal{D}_{in})$
 # Adjoint optimization
$\hat{a}_{\omega_n}, \xi_{n+1} \leftarrow \texttt{AdjointOpt}(\omega_n, \xi_0, \hat{h}_{\omega_n}, \mathcal{D})$
 # Outer gradient estimation
Sample a mini-batch $\mathcal{B} = (\mathcal{B}_{out}, \mathcal{B}_{in})$ from $\mathcal{D} = (\mathcal{D}_{out}, \mathcal{D}_{in})$
$g_{out} \leftarrow \texttt{TotalGrad}(\omega_n, \hat{h}_{\omega_n}, \hat{a}_{\omega_n}, \mathcal{B})$
$\omega_{n+1} \leftarrow$ update ω_n using g_{out};
end for

Application I : Model-based RL



Application I : Model-based RL

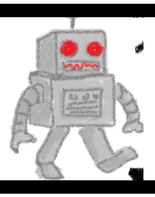


Optimal agent

$h(\omega)$

Real world

Optimal agent $h(\omega)$

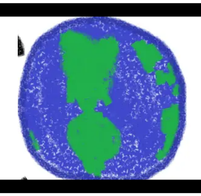


Agent model h

Action

Feedback

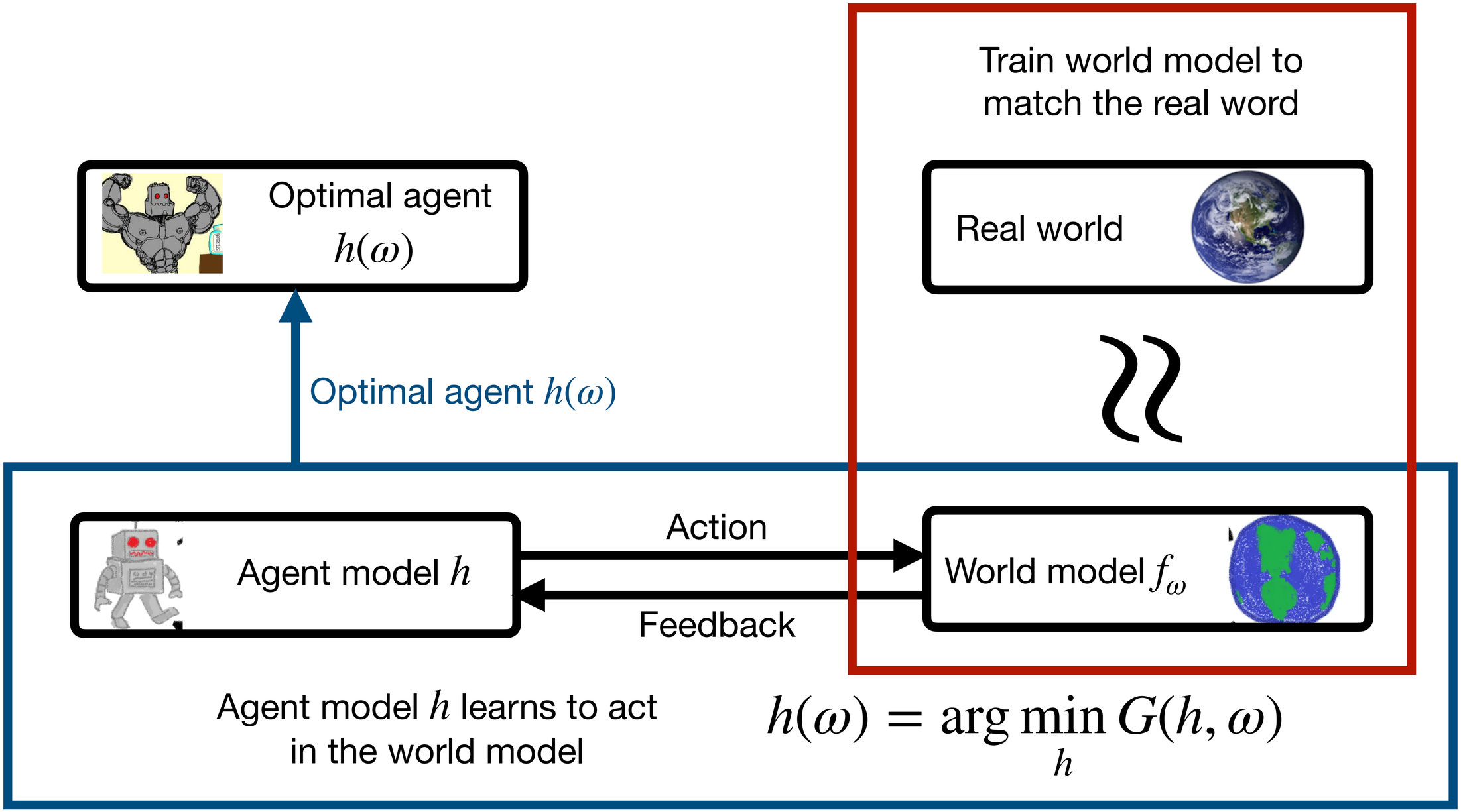
World model f_ω



Agent model h learns to act
in the world model

$$h(\omega) = \arg\min_h G(h, \omega)$$

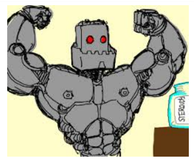
Application I : Model-based RL without BO



Application I : Model-based RL without BO

Requires accurate world models!

Train world model to match the real word

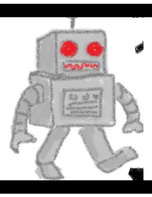


Optimal agent $h(\omega)$

Real world

≈

Optimal agent $h(\omega)$

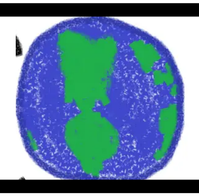


Agent model h

Action

Feedback

World model f_ω



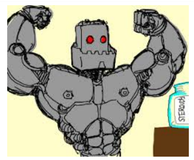
Agent model h learns to act in the world model

$$h(\omega) = \arg\min_h G(h, \omega)$$

Application I : Model-based RL with BO

World model trained so that optimal agent acts well in the real world

$$\arg\min_{\omega} L(h(\omega))$$



Optimal agent $h(\omega)$

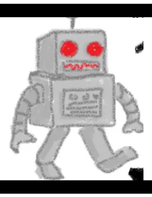
Action

Feedback

Real world

Optimal agent $h(\omega)$

Optimize ω

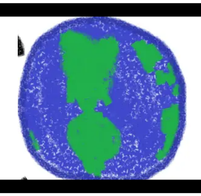


Agent model h

Action

Feedback

World model f_ω

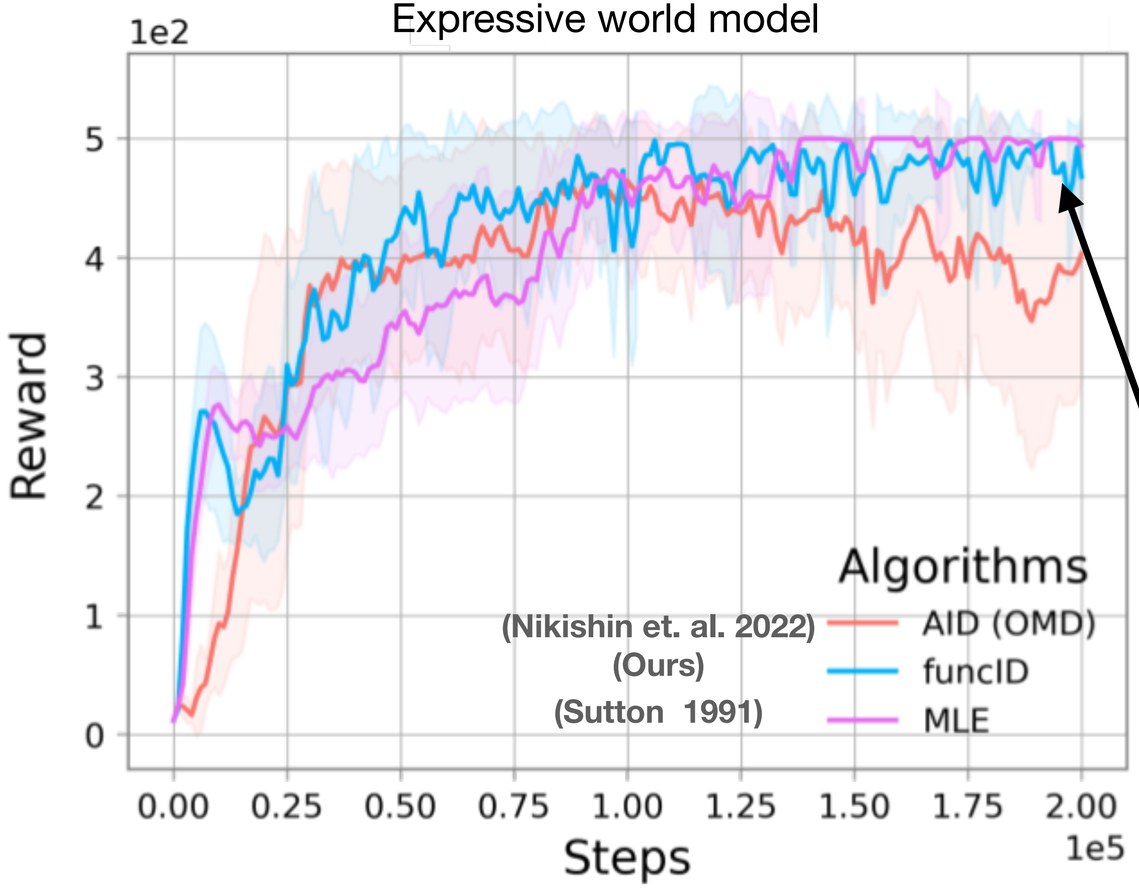


Agent model h learns to act in the world model

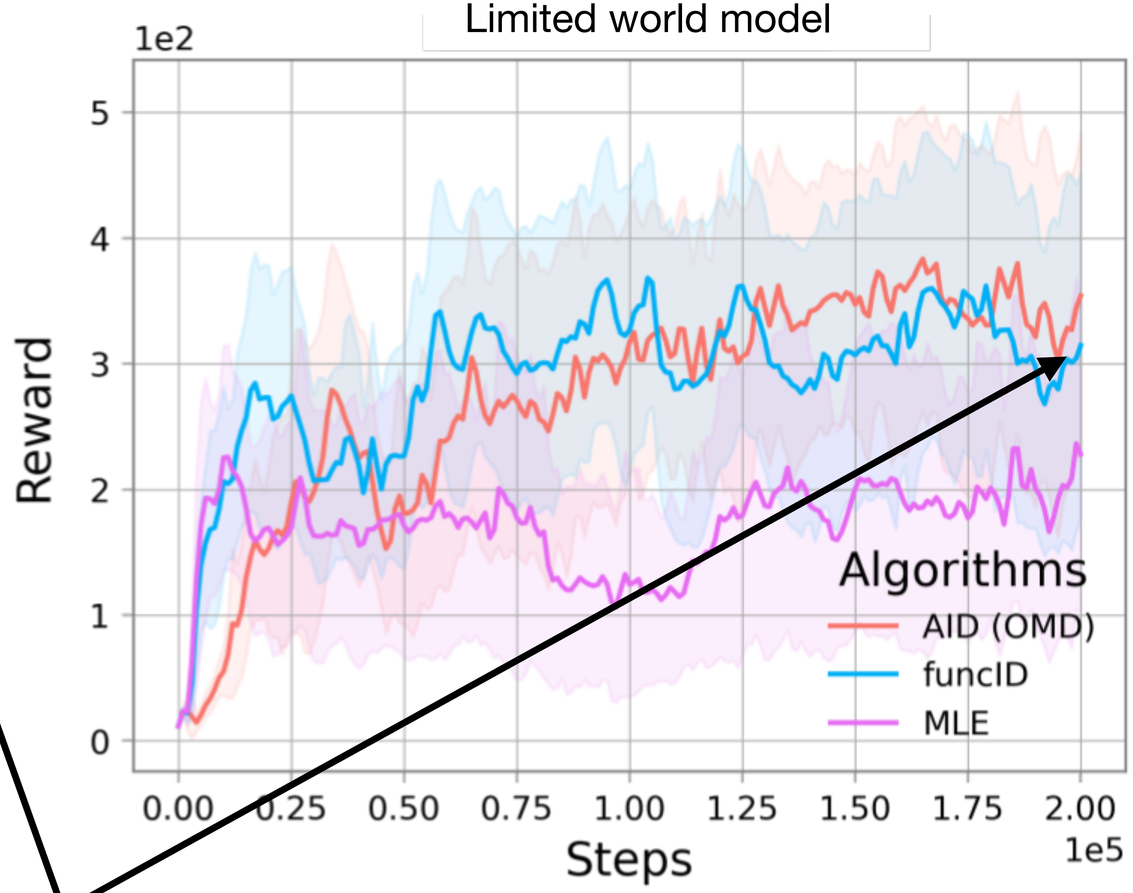
$$h(\omega) = \arg\min_{h} G(h, \omega)$$

Application I : Model-based RL with BO

Expressive world model

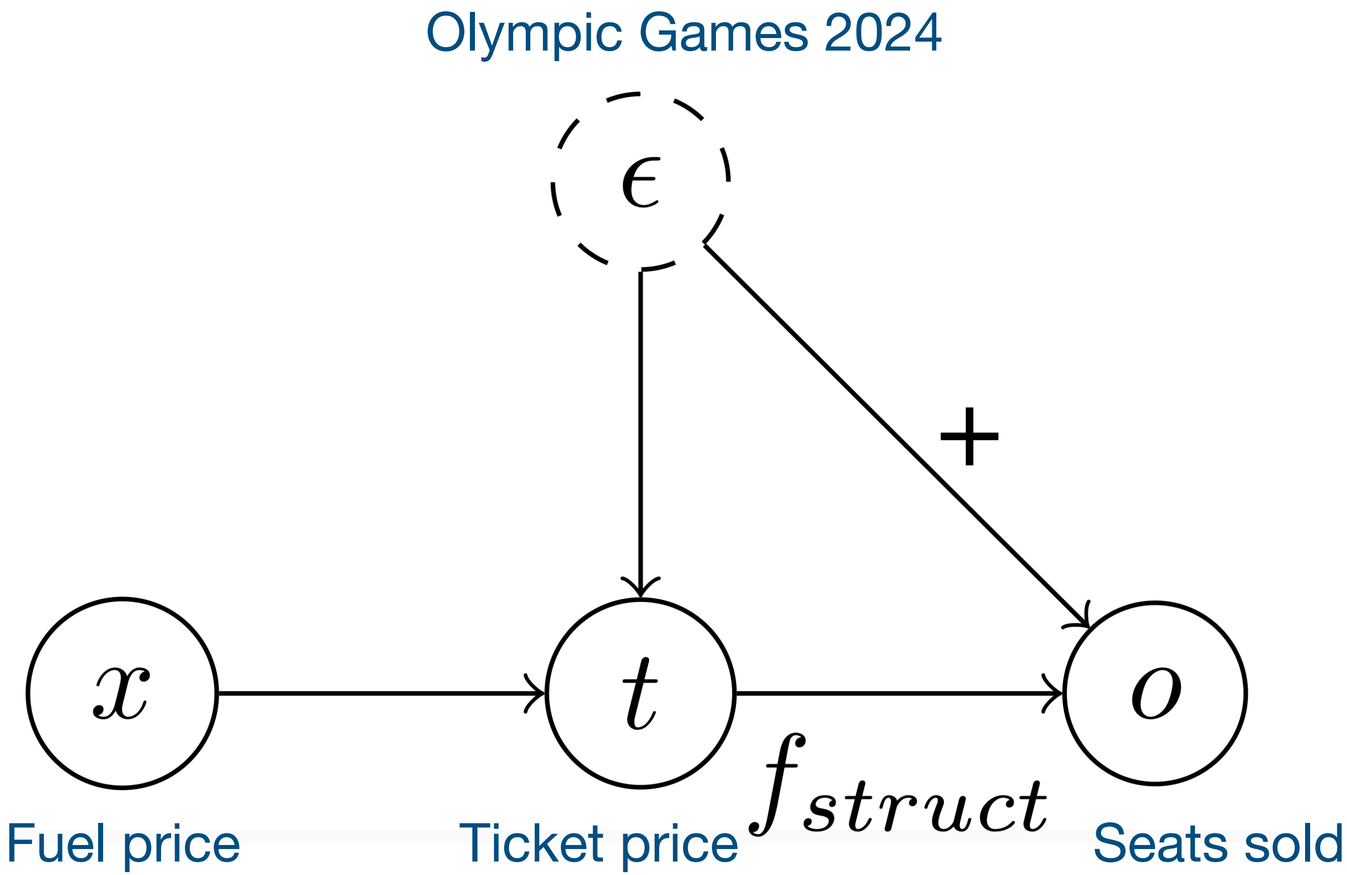


Limited world model



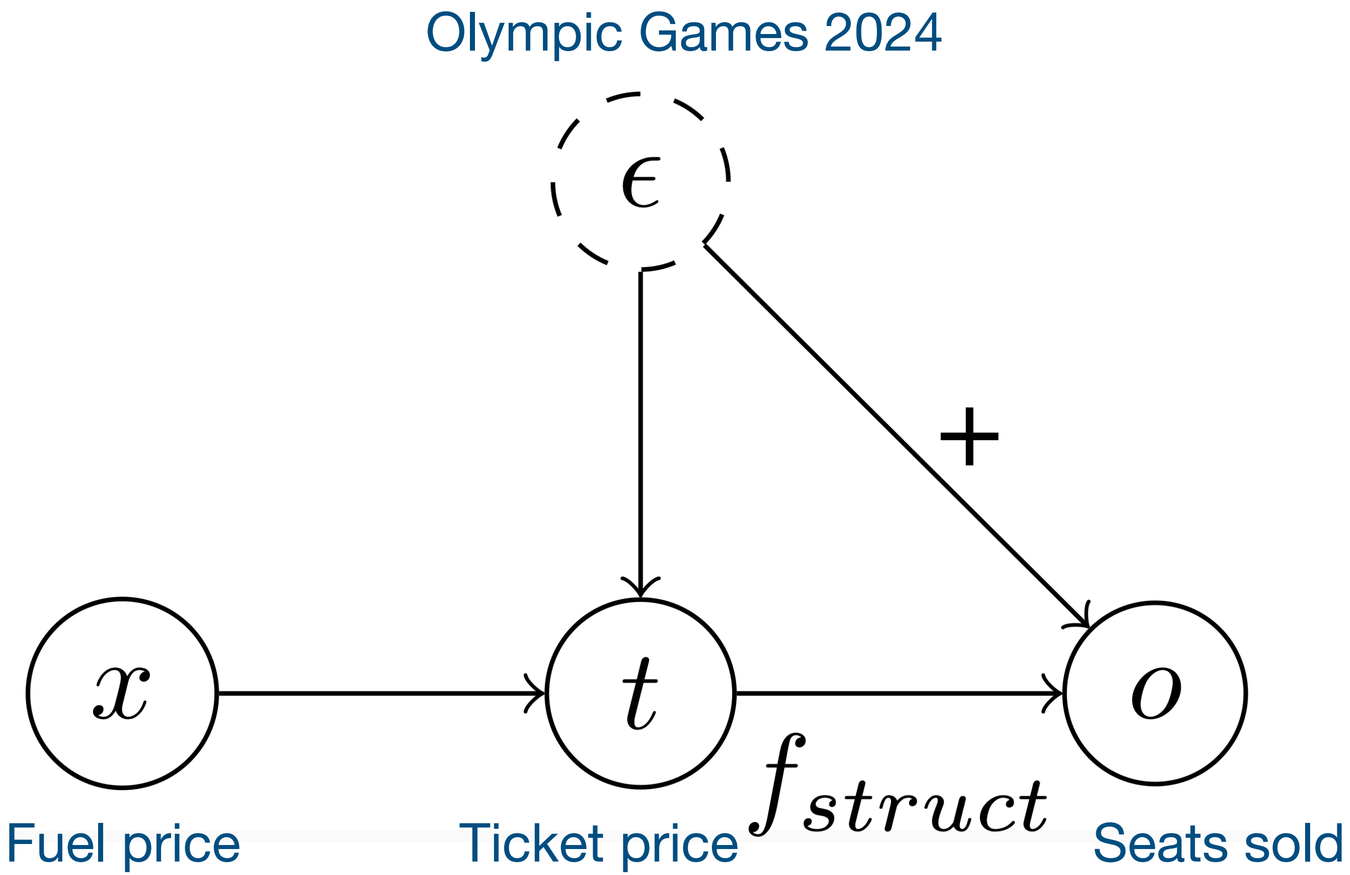
Performs well in both cases

Application II : Instrumental variable regression



$$o = f_{struct}(t) + \epsilon$$

Application II : Instrumental variable regression



$$o = f_{struct}(t) + \epsilon$$

Goal: Estimate f_{struct} from samples (x, t, o)

- Direct regression doesn't help

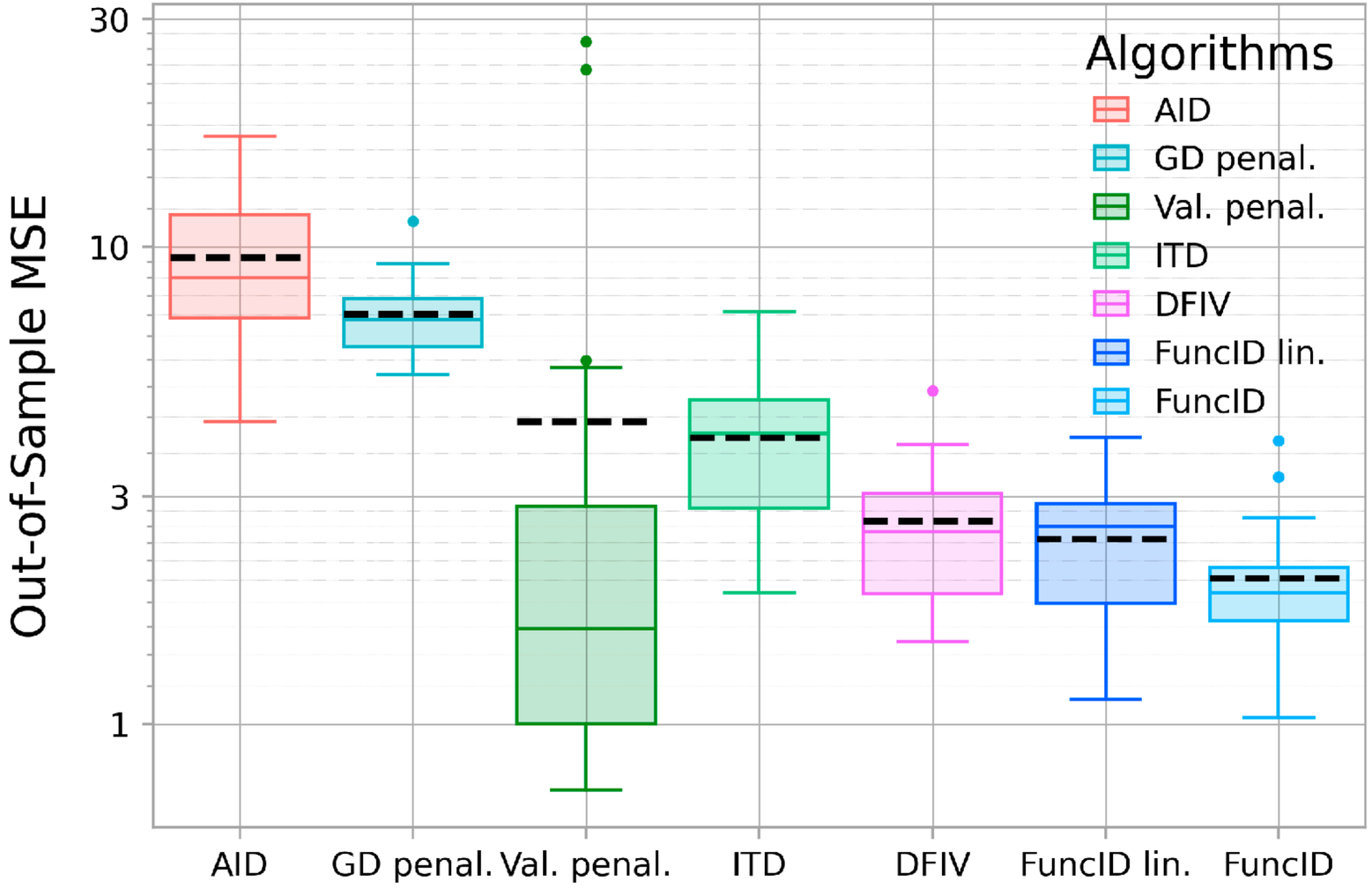
$$f_{struct} \neq \arg\min_f \mathbb{E}[\|o - f(t)\|^2]$$

- Can use the instrument x

$$f_{struct} = \arg\min_f \mathbb{E}[\|o - h_f^\star(x)\|^2]$$

$$h_f^\star = \arg\min_h \mathbb{E}[\|f(t) - h(x)\|^2]$$

Application II : Instrumental variable regression



Conclusion

Summary:

- Opportunity to obtain cost efficient and reliable ML using scientific modeling in the form of simulators
- Bilevel optimization: a promising framework to combine ML and simulations
- Challenges: Non-convexity arising from using deep networks introduces several technical challenges when performing bilevel optimization
- The functional point of view offers a general framework for addressing these challenges

Open questions:

- Extending the range of applications: Self-supervised learning, causal inference
- Extending the functional bilevel optimization framework: beyond L_2 spaces
- Convergence and generalization theory