Practical Black-box Optimization

Luigi Nardi

WASP-AI Assistant Professor Lund University

Researcher Stanford University

@RSS seminar, November 12, 2019

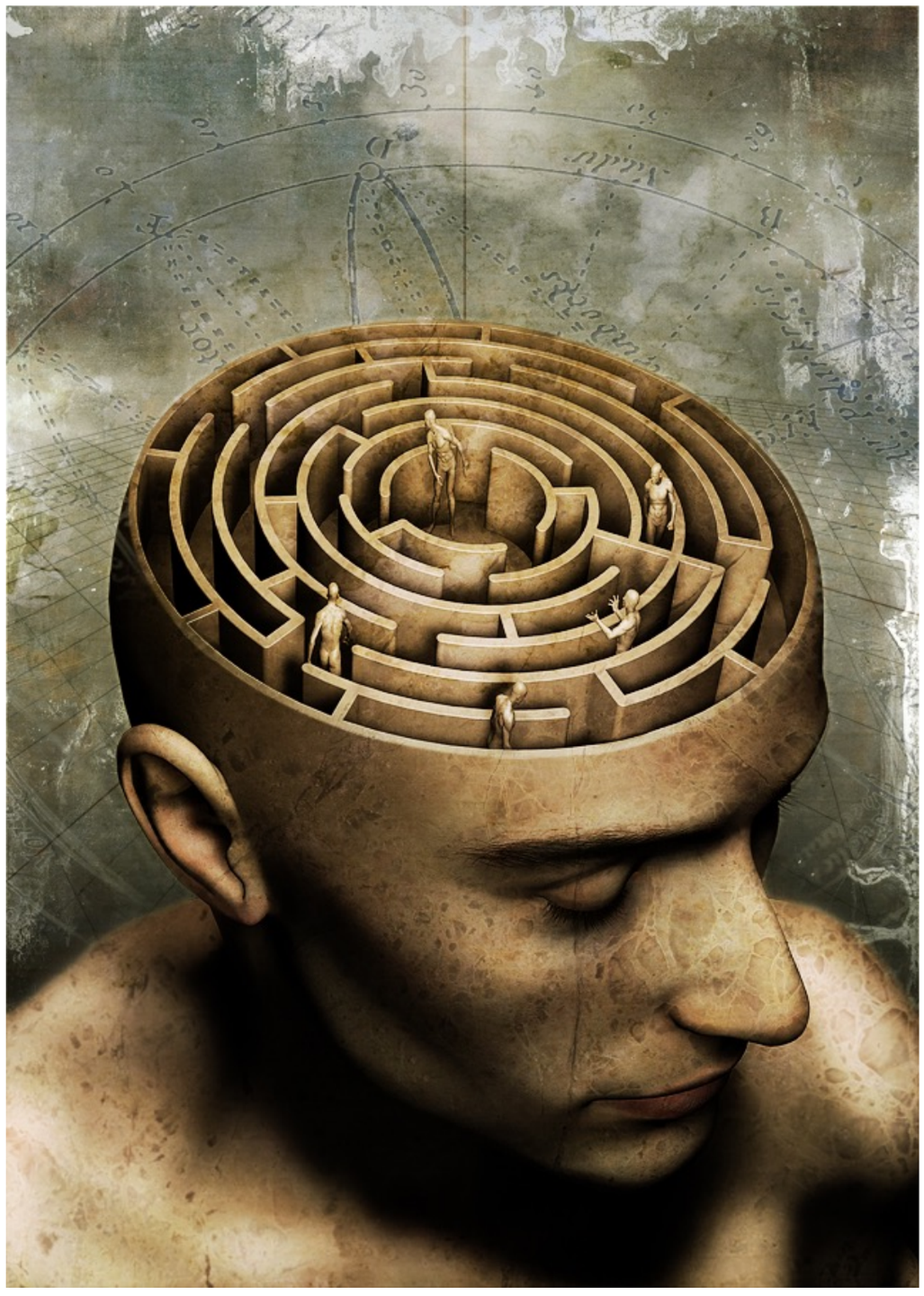
LUND UNIVERSITY

Main Papers Behind This Talk

1. Nardi, et al., "Practical Design Space Exploration", MASCOTS, 2019.
2. Koeplinger, et al., "Spatial: A Language and Compiler for Application Accelerators", PLDI, 2018.
3. Nardi, et al., "Algorithmic performance-accuracy trade-off in 3D vision applications using HyperMapper", iWAPT-IPDPS, 2017.
4. Saeedi, et al., "Application-oriented design space exploration for SLAM algorithms", ICRA, 2017.
5. Bodin, et al., "Integrating algorithmic parameters into benchmarking and design space exploration in 3D scene understanding", PACT, 2016.

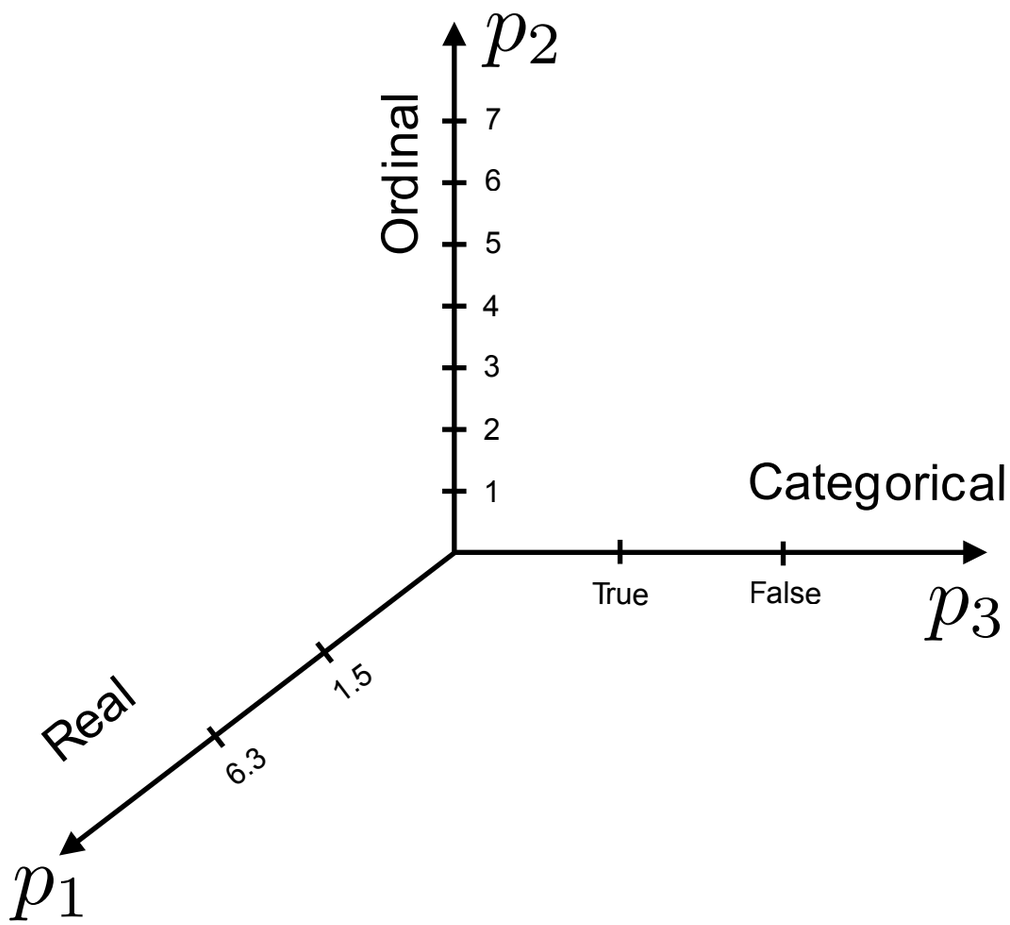
Outline

1. **Black-box Optimization**
   - **Problem Setting**
   - **Motivation Examples**
   - **Taxonomy**
2. The HyperMapper Framework
   - Bayesian Optimization
   - Pareto and Hypervolume Indicator
   - Prior Distribution
3. Experimental Results
   - Hardware Design
   - Computer Vision
   - Databases



Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

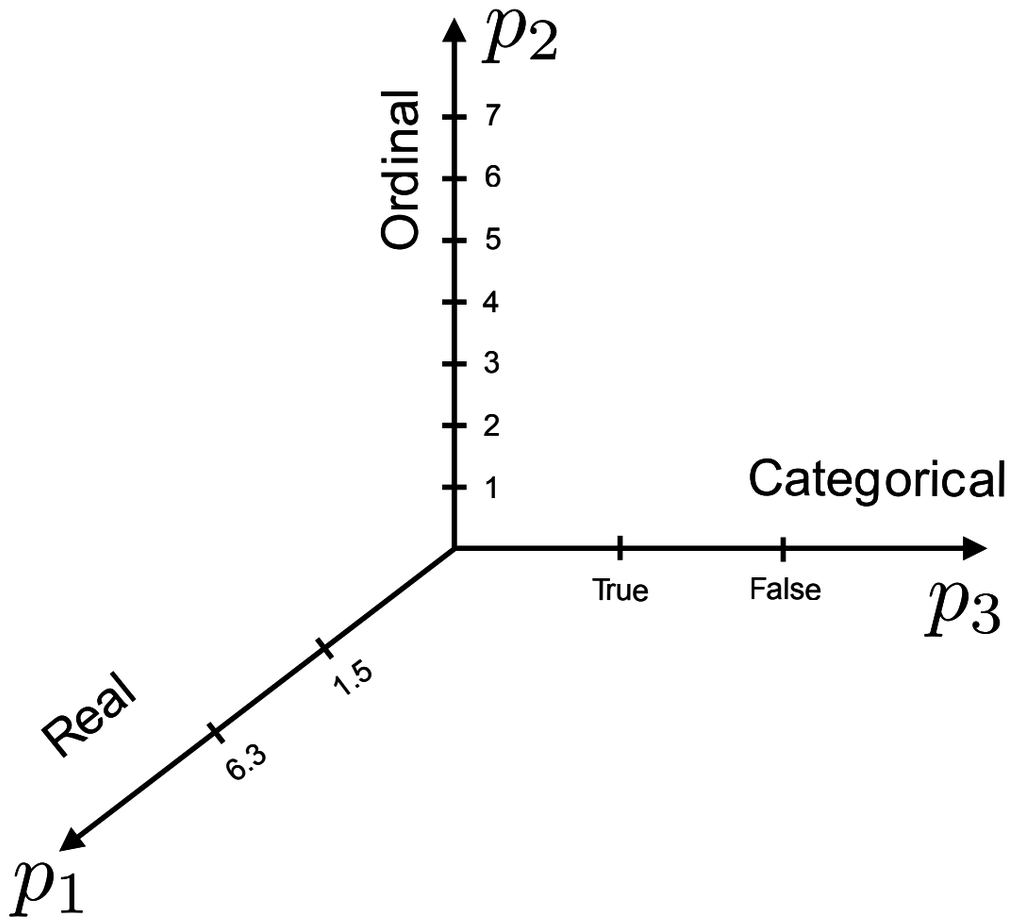
Input space
(a.k.a. search or design space)



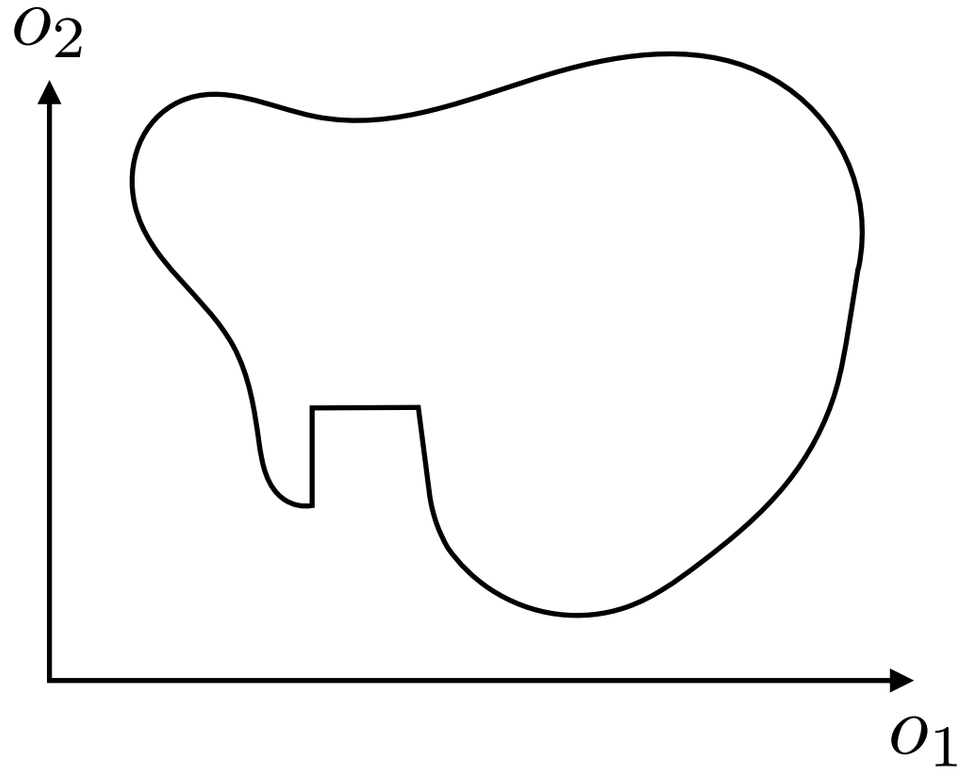
Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)



Output space
(a.k.a. objective space)

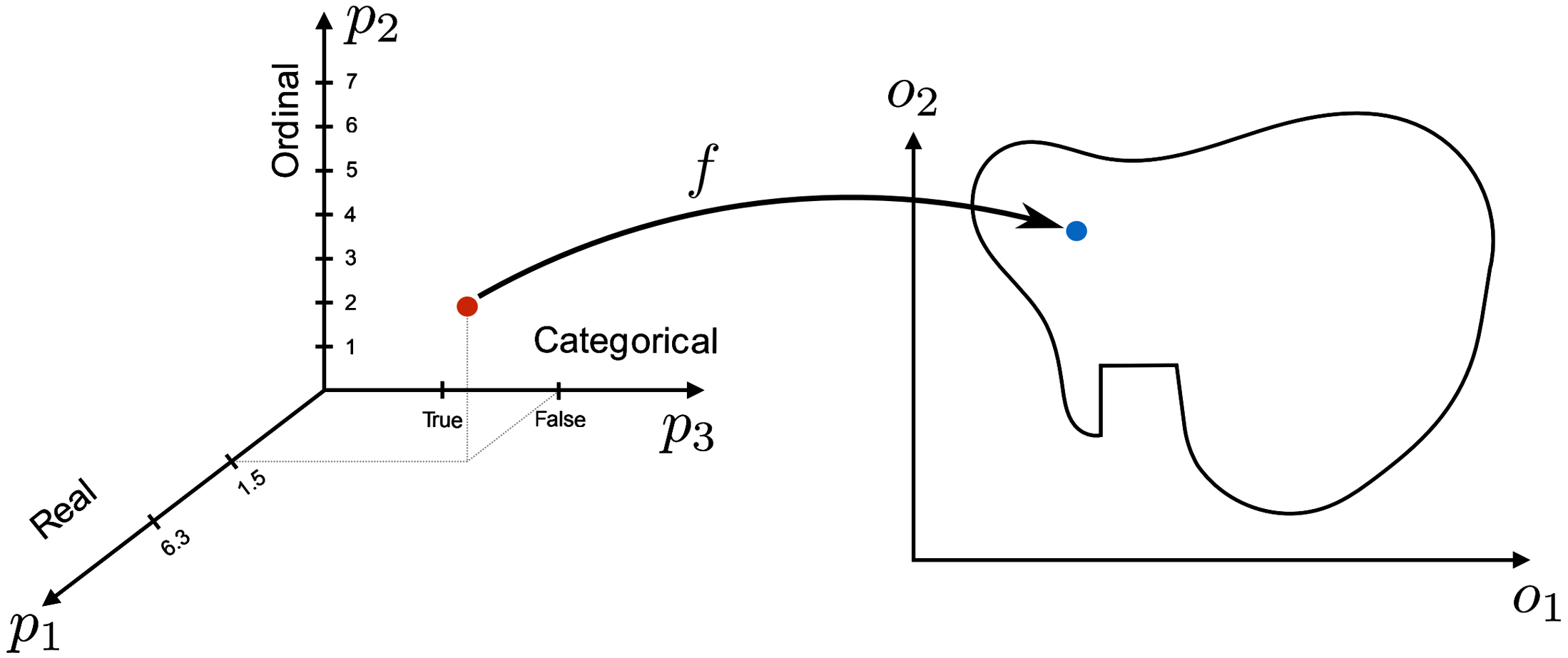


Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)

Output space
(a.k.a. objective space)

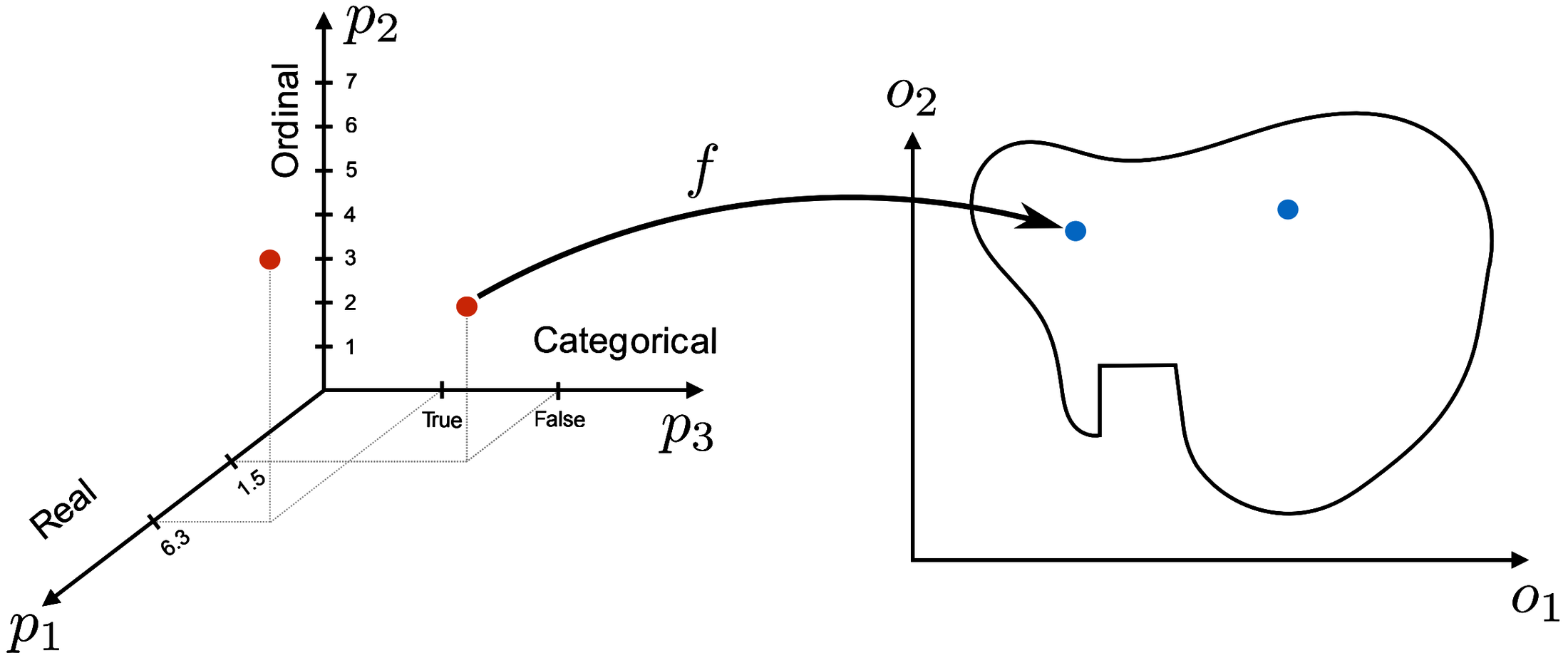


Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)

Output space
(a.k.a. objective space)

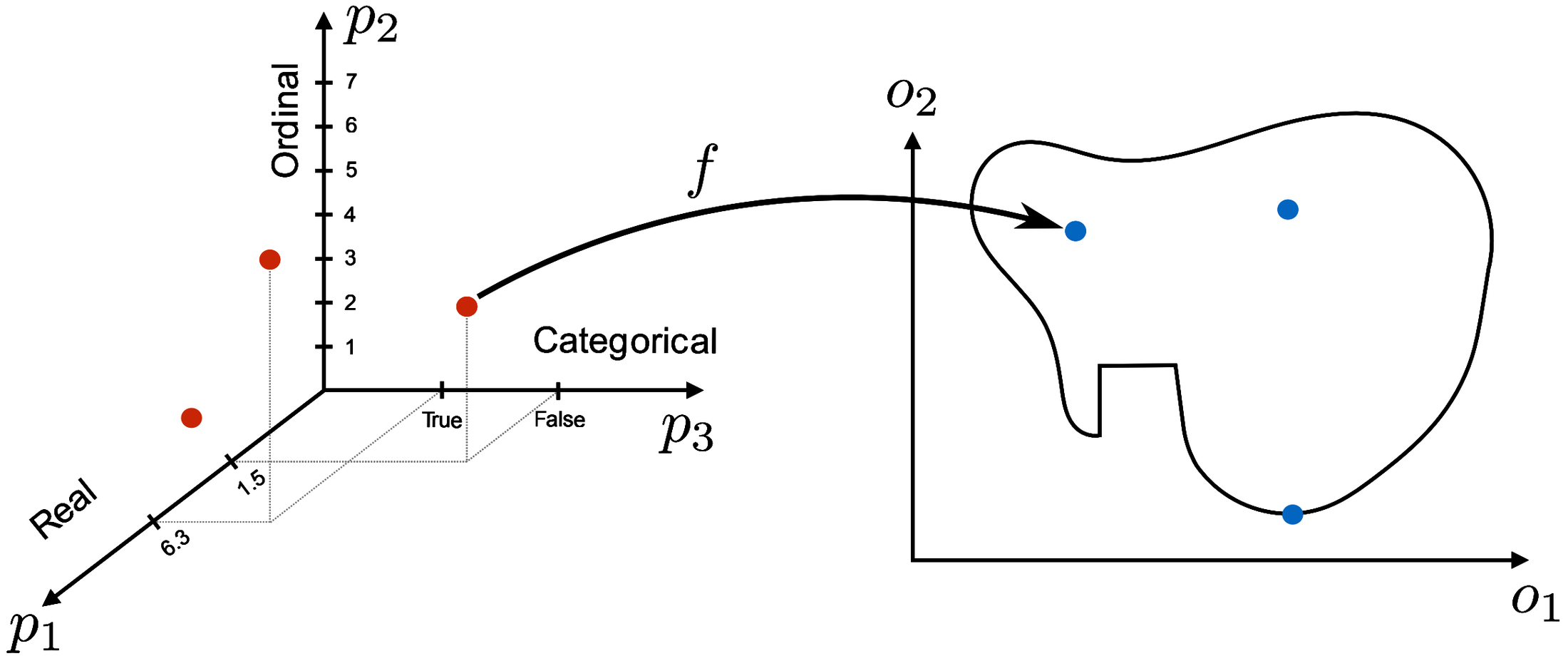


Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)

Output space
(a.k.a. objective space)

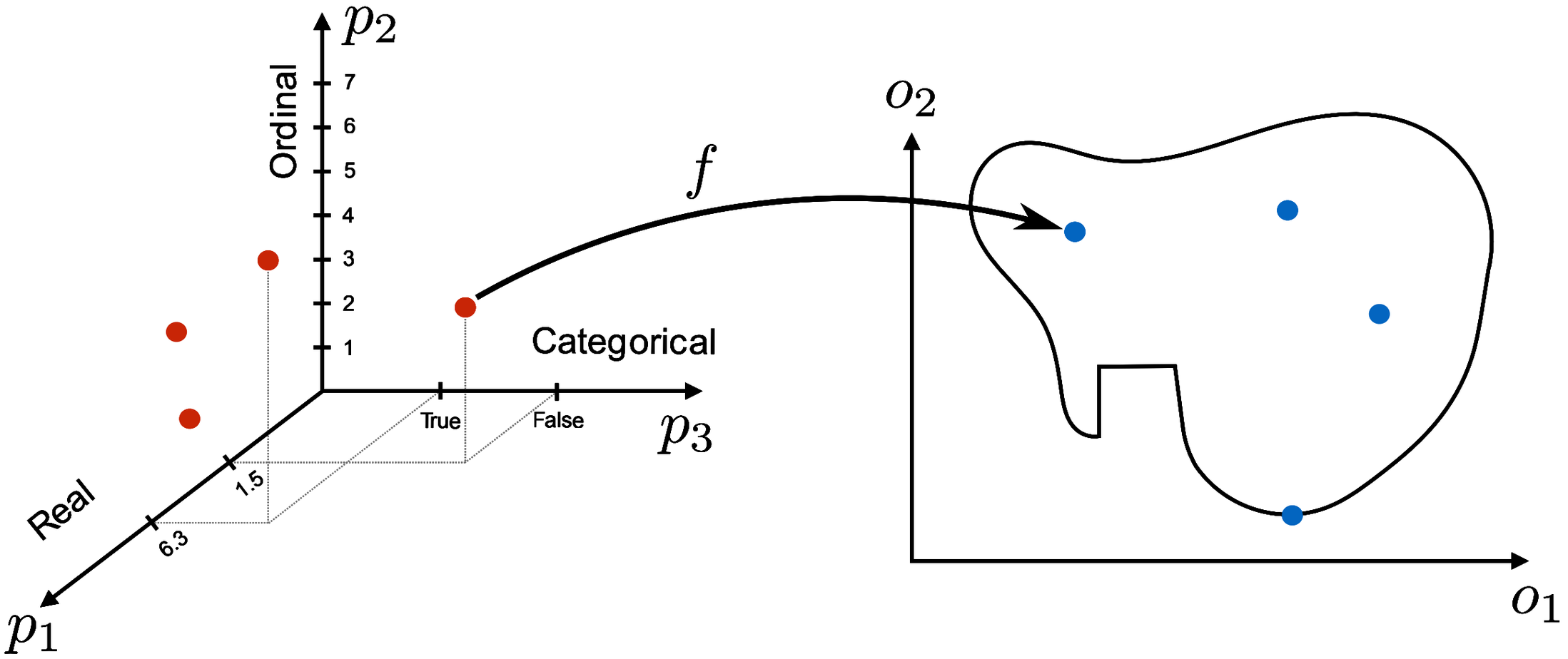


Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)

Output space
(a.k.a. objective space)

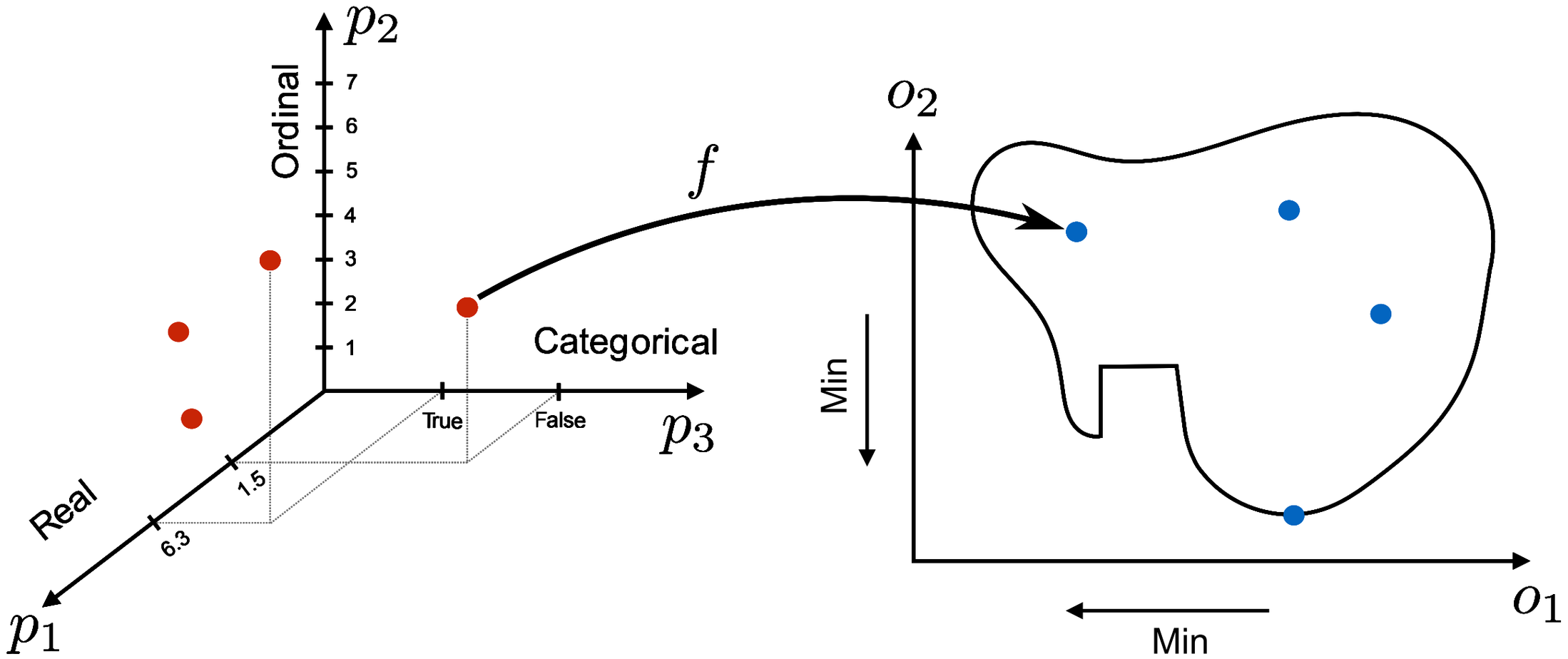


Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)

Output space
(a.k.a. objective space)

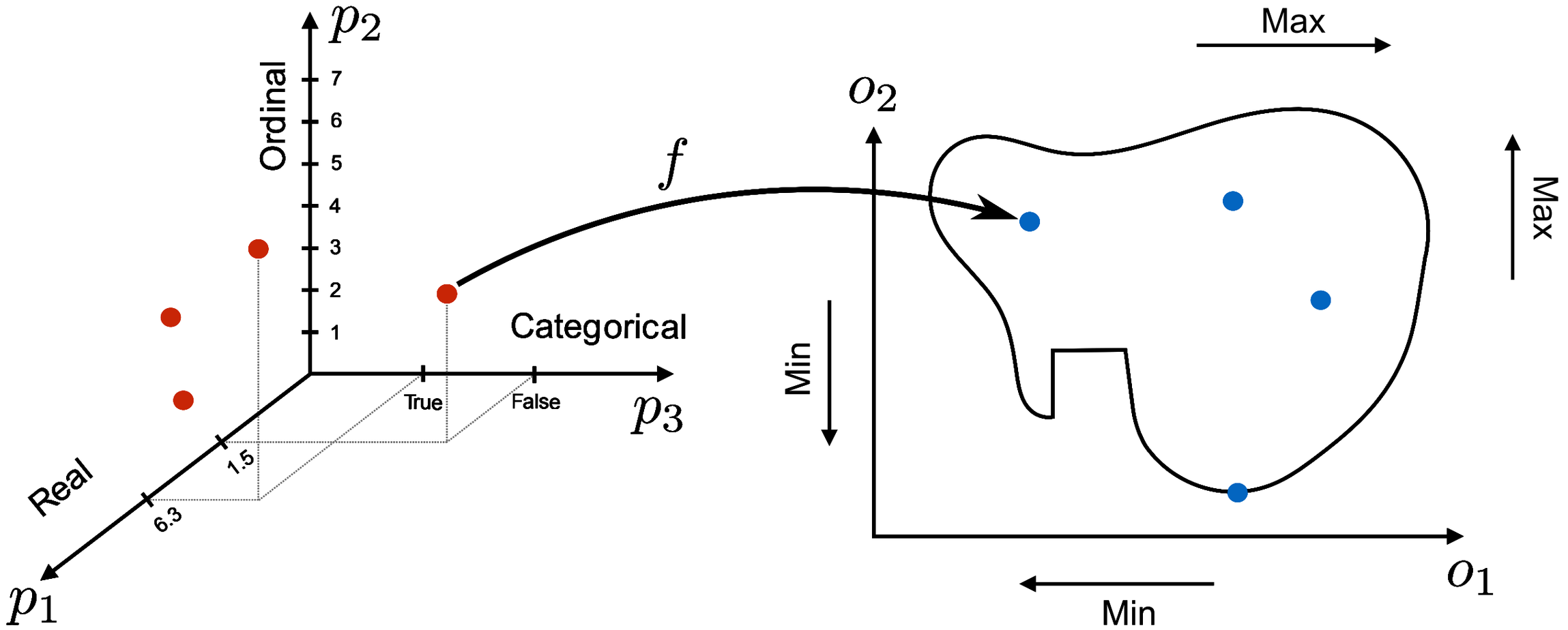


Derivatives are unavailable: e.g., categorical variables

Black-box Optimization (BBO)
3-parameters and 2-objective Pictorial

Input space
(a.k.a. search or design space)

Output space
(a.k.a. objective space)



Derivatives are unavailable: e.g., categorical variables

Black-box Optimization

- Define an objective function. Examples:
  - Accuracy, runtime, throughput, latency, area, energy, etc.

Black-box Optimization

- Define an objective function. Examples:
  - Accuracy, runtime, throughput, latency, area, energy, etc.
- Standard search procedures:
  - Grid search
  - Random search (very simple, works surprisingly well)
  - Heuristics: e.g., genetic algorithms
  - Black magic

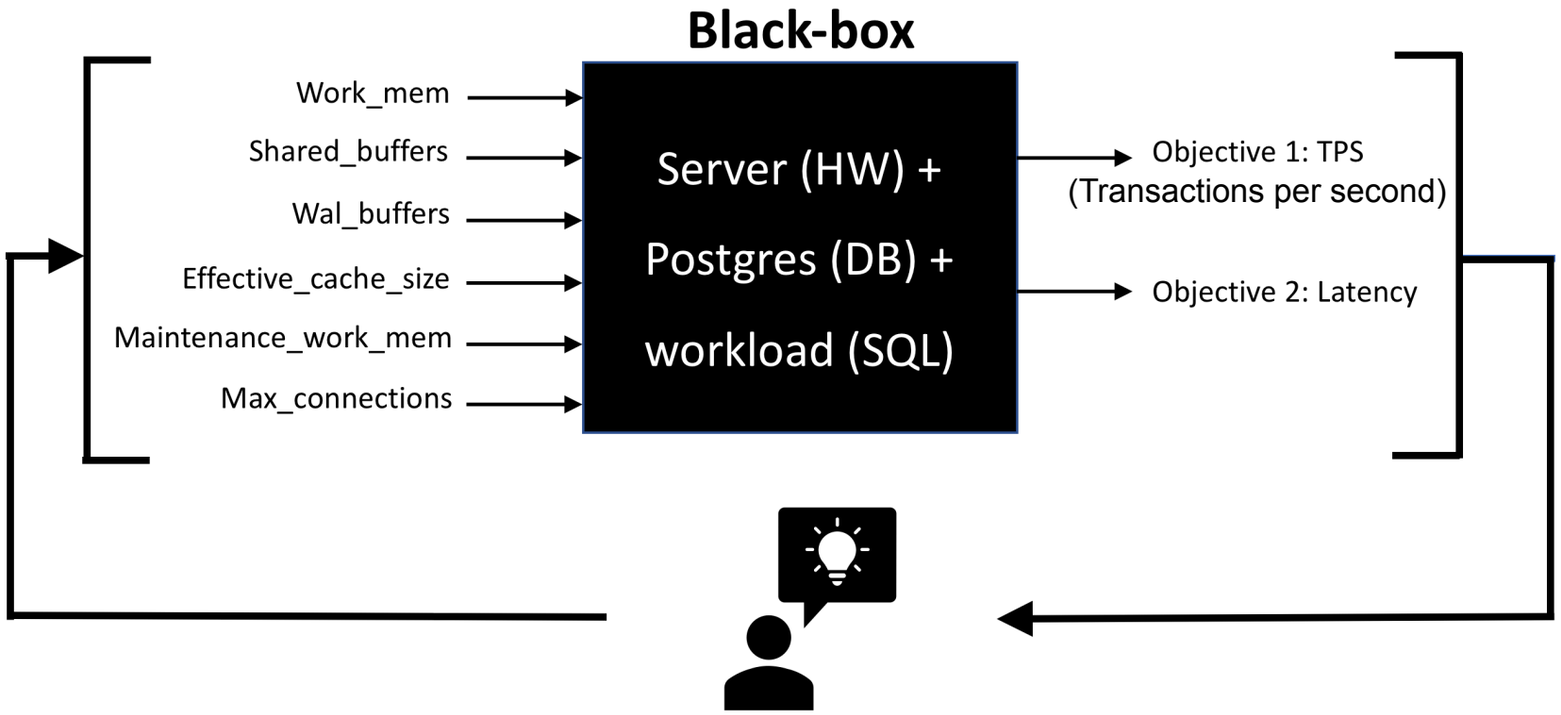
Black-box Optimization

- Define an objective function. Examples:
  - Accuracy, runtime, throughput, latency, area, energy, etc.
- Standard search procedures:
  - Grid search
  - Random search (very simple, works surprisingly well)
  - Heuristics: e.g., genetic algorithms
  - Black magic
- Painful:
  - Search may be very expensive (e.g., time or money)
  - Usually limited sampling budget
  - Possibly noisy, unknown feasibility constraints, multi-objective

Black-box Optimization

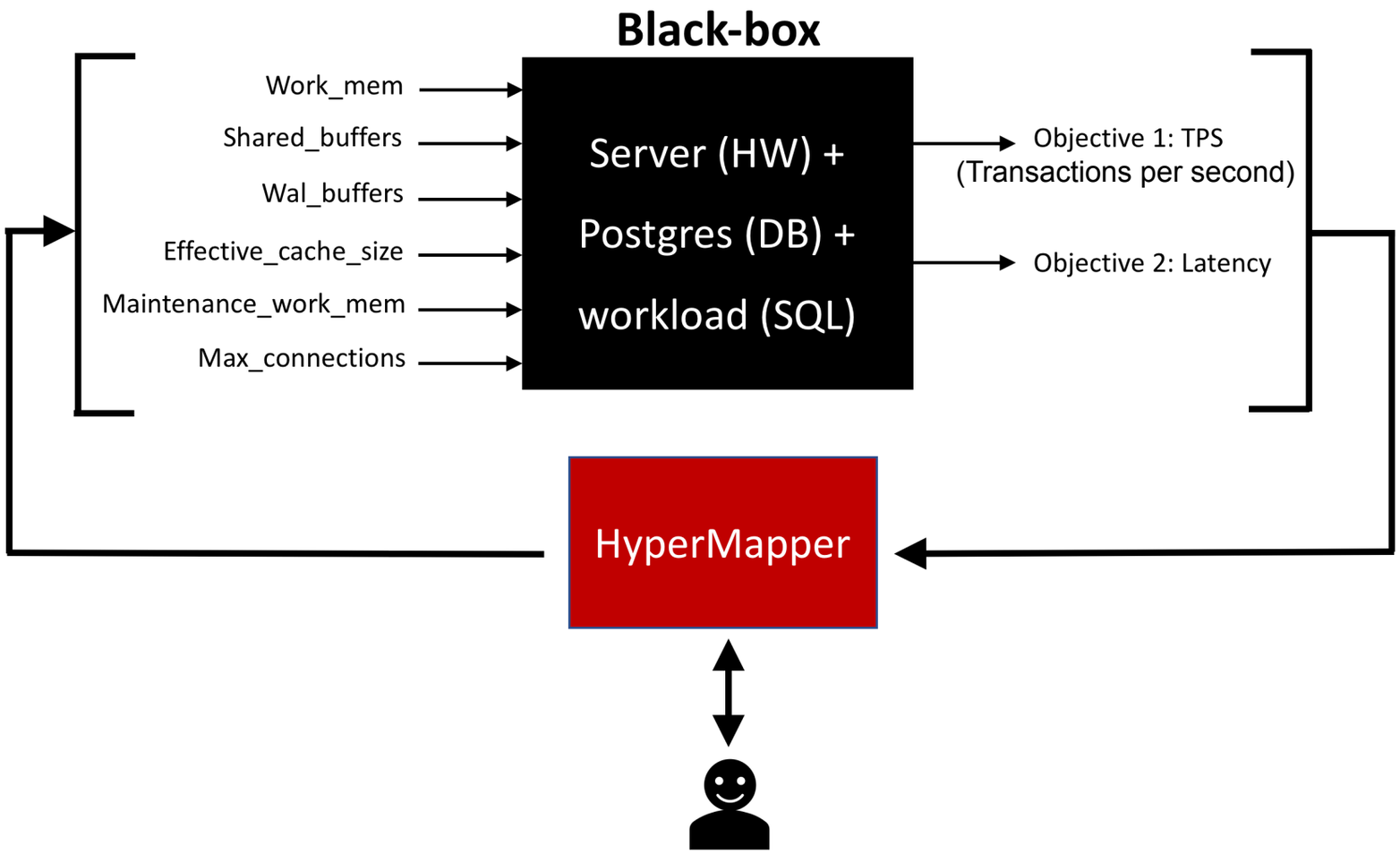
- Define an objective function. Examples:
  - Accuracy, runtime, throughput, latency, area, energy, etc.
- Standard search procedures:
  - Grid search
  - Random search (very simple, works surprisingly well)
  - Heuristics: e.g., genetic algorithms
  - Black magic
- Painful:
  - Search may be very expensive (e.g., time or money)
  - Usually limited sampling budget
  - Possibly noisy, unknown feasibility constraints, multi-objective
- Alternative approach: Bayesian Optimization (BO)

Example: PostgreSQL/Azure Autotuning



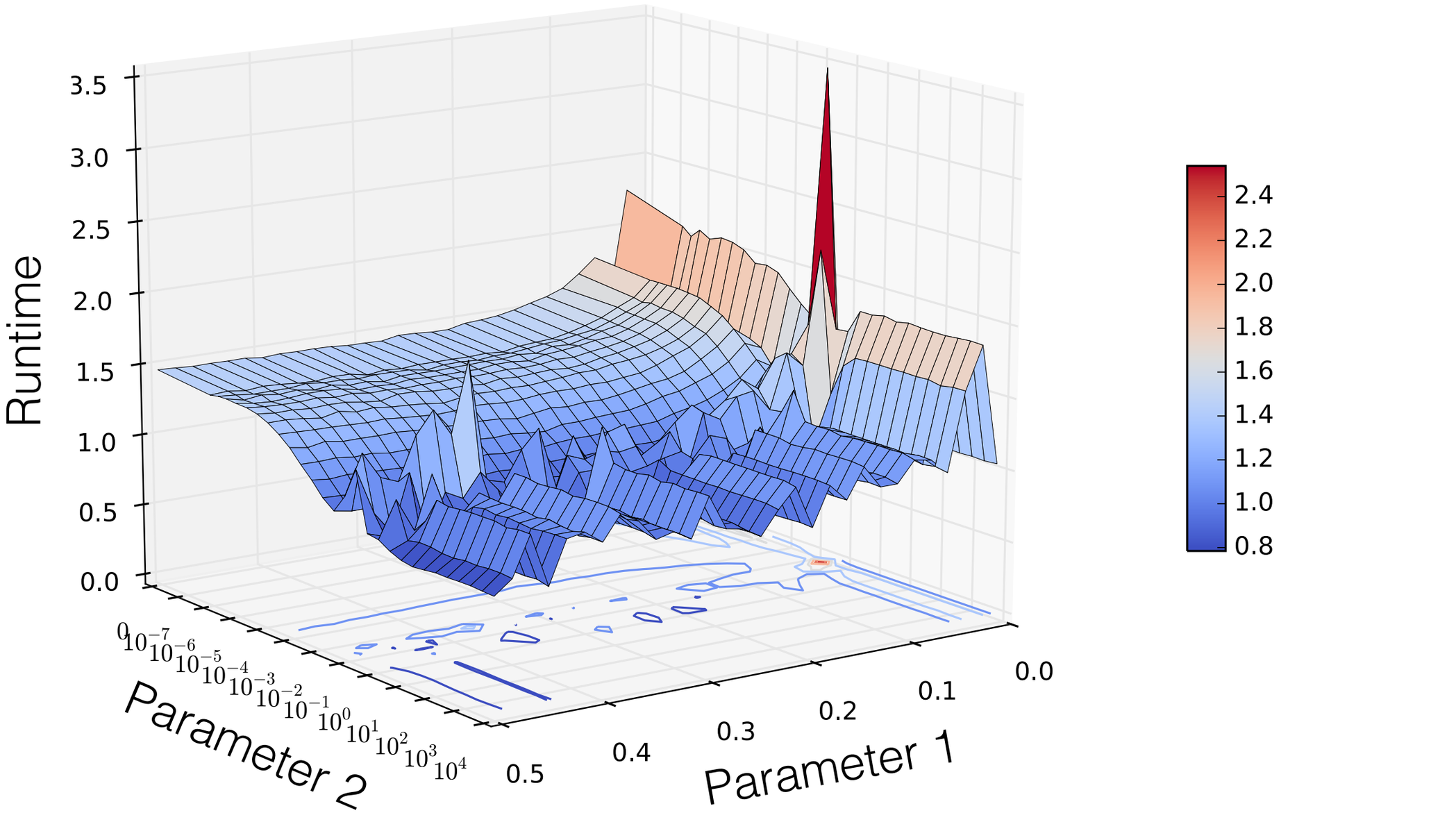
- Estimate of design space size: 1e+18
- This is an on going collaboration with **Microsoft** Seattle
- Need to optimize anytime the black-box changes: server, PostgreSQL, DB, queries
- **100,000 PostgreSQL databases** in Microsoft Azure

Example: PostgreSQL/Azure Autotuning



- Estimate of design space size: 1e+18
- This is an on going collaboration with **Microsoft** Seattle
- Need to optimize anytime the black-box changes: server, PostgreSQL, DB, queries
- **100,000 PostgreSQL databases** in Microsoft Azure

Motivation - Mono-objective



- Benchmark: SLAMBench 1.0 runtime response surface is: non-linear, multi-modal and non-smooth

Practical BBO: Important Features

1. Real, integer, ordinal and categorical variables (RIOC var.)
    - Example: parallelism is categorical; # of CPUs is ordinal

Practical BBO: Important Features

1. Real, integer, ordinal and categorical variables (RIOC var.)
    - Example: parallelism is categorical; # of CPUs is ordinal
2. User prior knowledge in the search (Prior)
    - Example: # threads on a CPU with 4 cores? Gaussian-shaped around 4

Practical BBO: Important Features

1. Real, integer, ordinal and categorical variables (RIOC var.)
    - Example: parallelism is categorical; # of CPUs is ordinal
2. User prior knowledge in the search (Prior)
    - Example: # threads on a CPU with 4 cores? Gaussian-shaped around 4
3. Multi-objective optimization (Multi)
    - Example: trade-off accuracy and inference time

Practical BBO: Important Features

1. Real, integer, ordinal and categorical variables (RIOC var.)
   - Example: parallelism is categorical; # of CPUs is ordinal
2. User prior knowledge in the search (Prior)
   - Example: # threads on a CPU with 4 cores? Gaussian-shaped around 4
3. Multi-objective optimization (Multi)
   - Example: trade-off accuracy and inference time
4. Unknown feasibility constraints (Constr.)
   - Example: black-box function crash; a design doesn't fit in the FPGA

BBO Tools Taxonomy

None of the tools available support all these BBO features

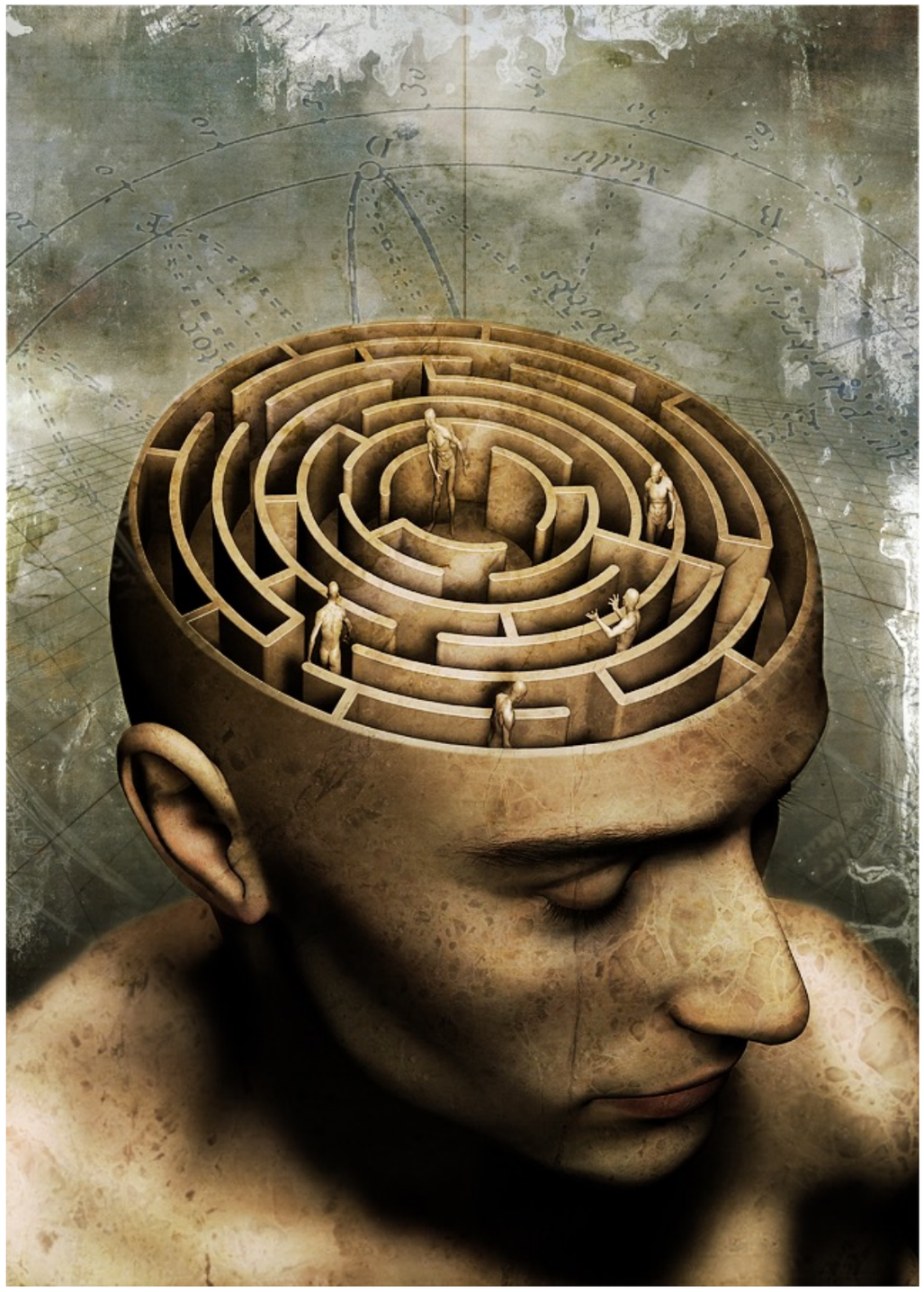
We introduce a new framework dubbed **HyperMapper**

Name	Multi	RIOC var.	Constr.	Prior
GpyOpt	✗	✗	✗	✗
OpenTuner	✗	✓	✗	✗
SURF	✗	✓	✗	✗
SMAC	✗	✓	✗	✗
Spearmint	✗	✗	✓	✗
Hyperopt	✗	✓	✗	✓
Hyperband	✗	✓	✗	✗
GPflowOpt	✓	✗	✓	✗
cBO	✗	✗	✓	✗
BOHB	✗	✓	✗	✗
HyperMapper	✓	✓	✓	✓

[Bodin, et al.] "Integrating algorithmic parameters into benchmarking and design space exploration in 3D scene understanding", PACT, 2016.
[Nardi, et al.] "Algorithmic performance-accuracy trade-off in 3D vision applications using HyperMapper", IPDPS-iWAPT, 2017.
[Nardi, et al., 2019] "Practical Design Space Exploration", MASCOTS, 2019.

Outline

1. Black-box Optimization
   - Problem Setting
   - Motivation Examples
   - Taxonomy
2. **The HyperMapper Framework**
   - **Bayesian Optimization**
   - **Pareto and Hypervolume Indicator**
   - **Prior Distribution**
3. Experimental Results
   - Hardware Design
   - Computer Vision
   - Databases



Bayesian Optimization

Setting

Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

Bayesian Optimization

Setting

Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

- Find global minimum of objective function f: $x^* = \arg\min_{x \in \mathbb{X}} f(x)$

Bayesian Optimization

Setting

Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

- Find global minimum of objective function f: $x^* = \arg\min_{x \in \mathbb{X}} f(x)$
- We can evaluate the objective f pointwise
- But we do not have an easy functional form or gradients

Bayesian Optimization

Setting

Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

- Find global minimum of objective function f: $x^* = \arg\min_{x \in \mathbb{X}} f(x)$
- We can evaluate the objective f pointwise
  - But we do not have an easy functional form or gradients
- Build a **probabilistic proxy model** for f
  - Using outcomes of past experiments as training data

Bayesian Optimization

Setting

Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

- Find global minimum of objective function f: $x^* = \arg\min_{x \in \mathbb{X}} f(x)$
- We can evaluate the objective f pointwise
  - But we do not have an easy functional form or gradients
- Build a **probabilistic proxy model** for f
  - Using outcomes of past experiments as training data
- Proxy model is **cheaper to evaluate** than the objective

Bayesian Optimization

Setting

Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

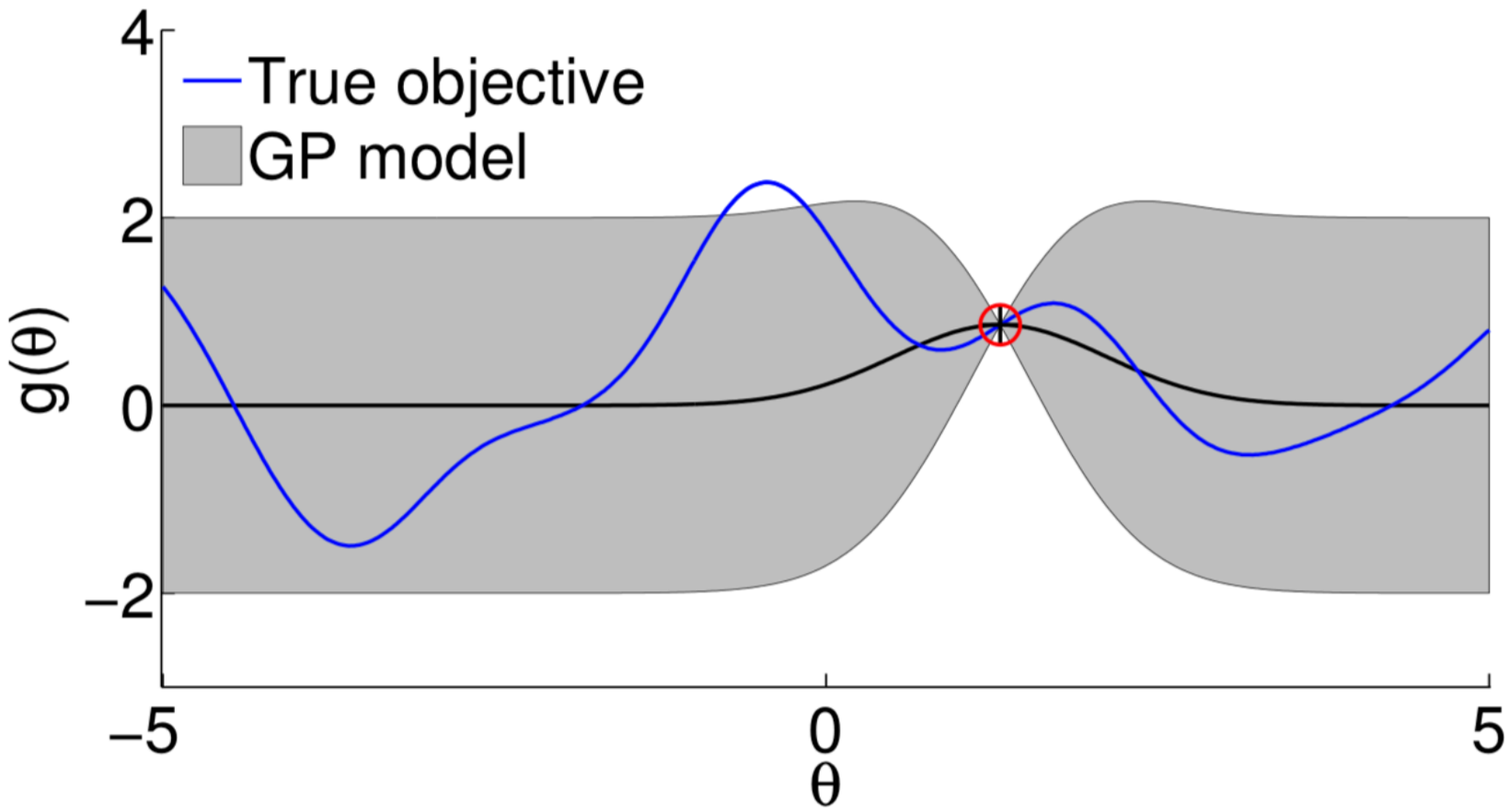
- Find global minimum of objective function f: $x^* = \arg\min_{x \in \mathbb{X}} f(x)$
- We can evaluate the objective f pointwise
  - But we do not have an easy functional form or gradients
- Build a **probabilistic proxy model** for f
  - Using outcomes of past experiments as training data
- Proxy model is **cheaper to evaluate** than the objective
- **Optimize cheap proxy** function
  - To determine where to evaluate the true objective next

Bayesian Optimization

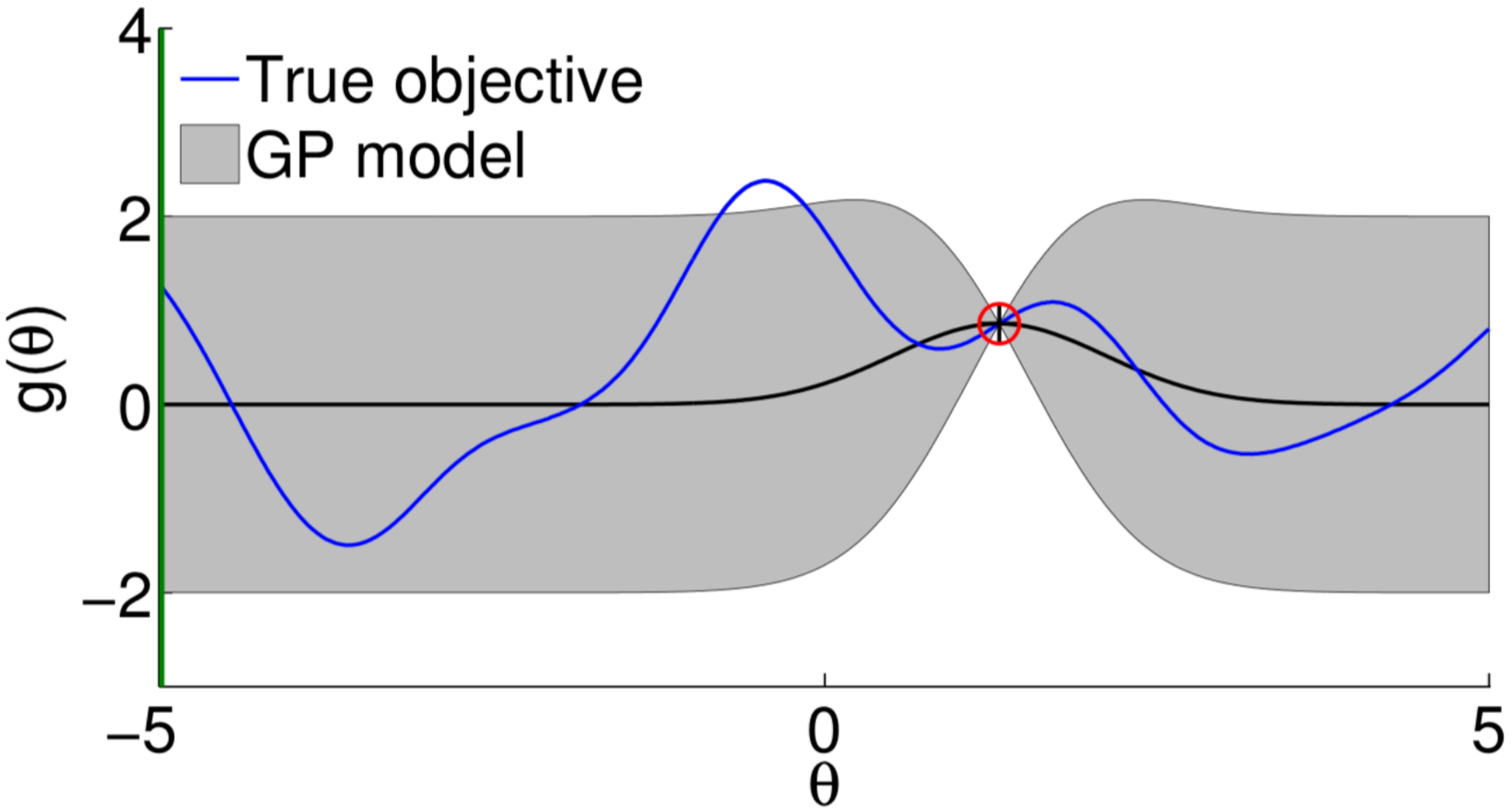
Setting
Globally optimize an objective function that is expensive to evaluate (e.g., PostgreSQL example above)

- Find global minimum of objective function f: $x^* = \arg\min_{x \in \mathbb{X}} f(x)$
- We can evaluate the objective f pointwise
  - But we do not have an easy functional form or gradients
- Build a **probabilistic proxy model** for f
  - Using outcomes of past experiments as training data
- Proxy model is **cheaper to evaluate** than the objective
- **Optimize cheap proxy** function
  - To determine where to evaluate the true objective next
- Standard proxy: Gaussian process
  - But also: Random Forests, Tree Parzen Estimators, etc.

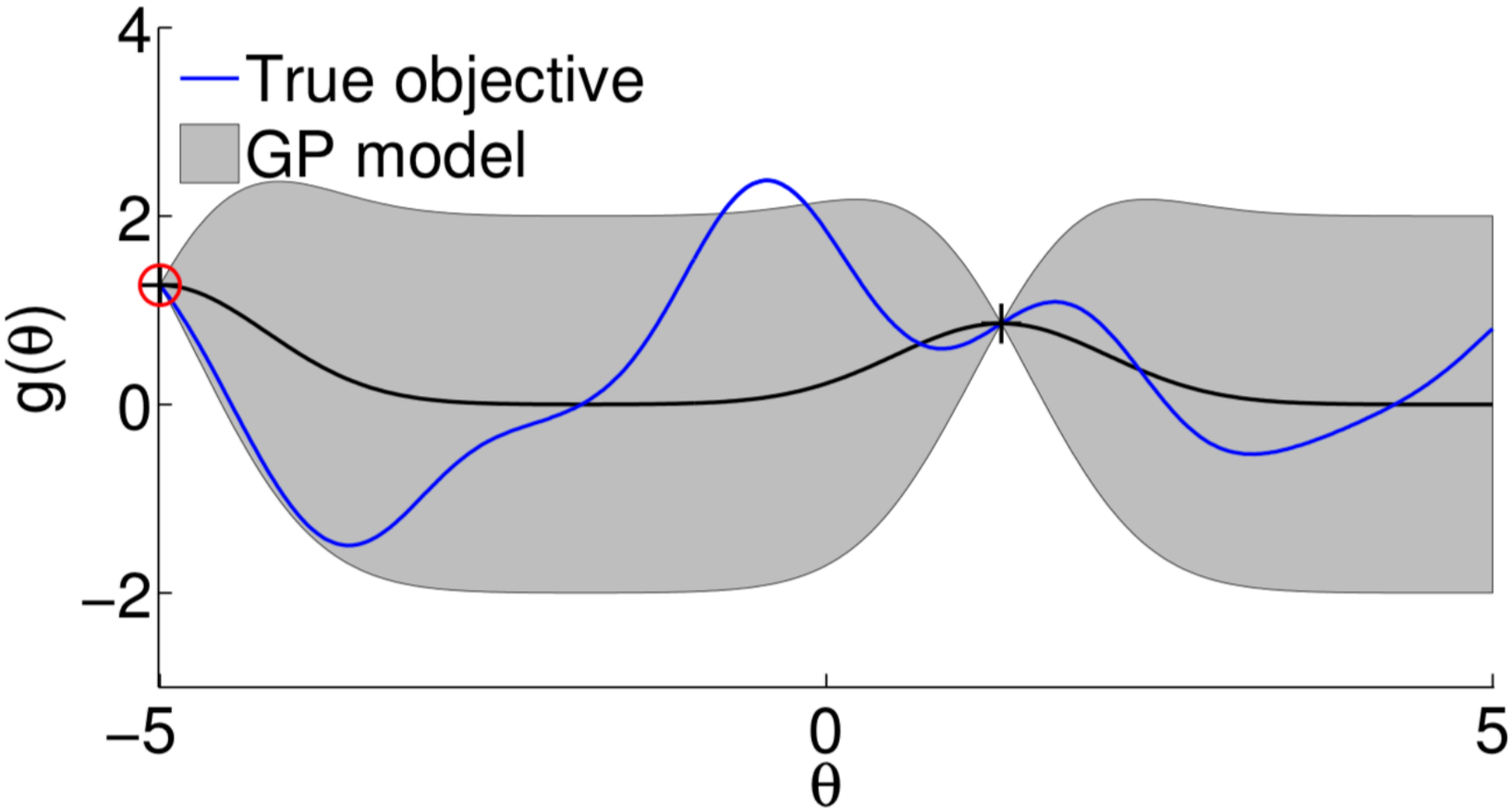
Bayesian Optimization: Illustration



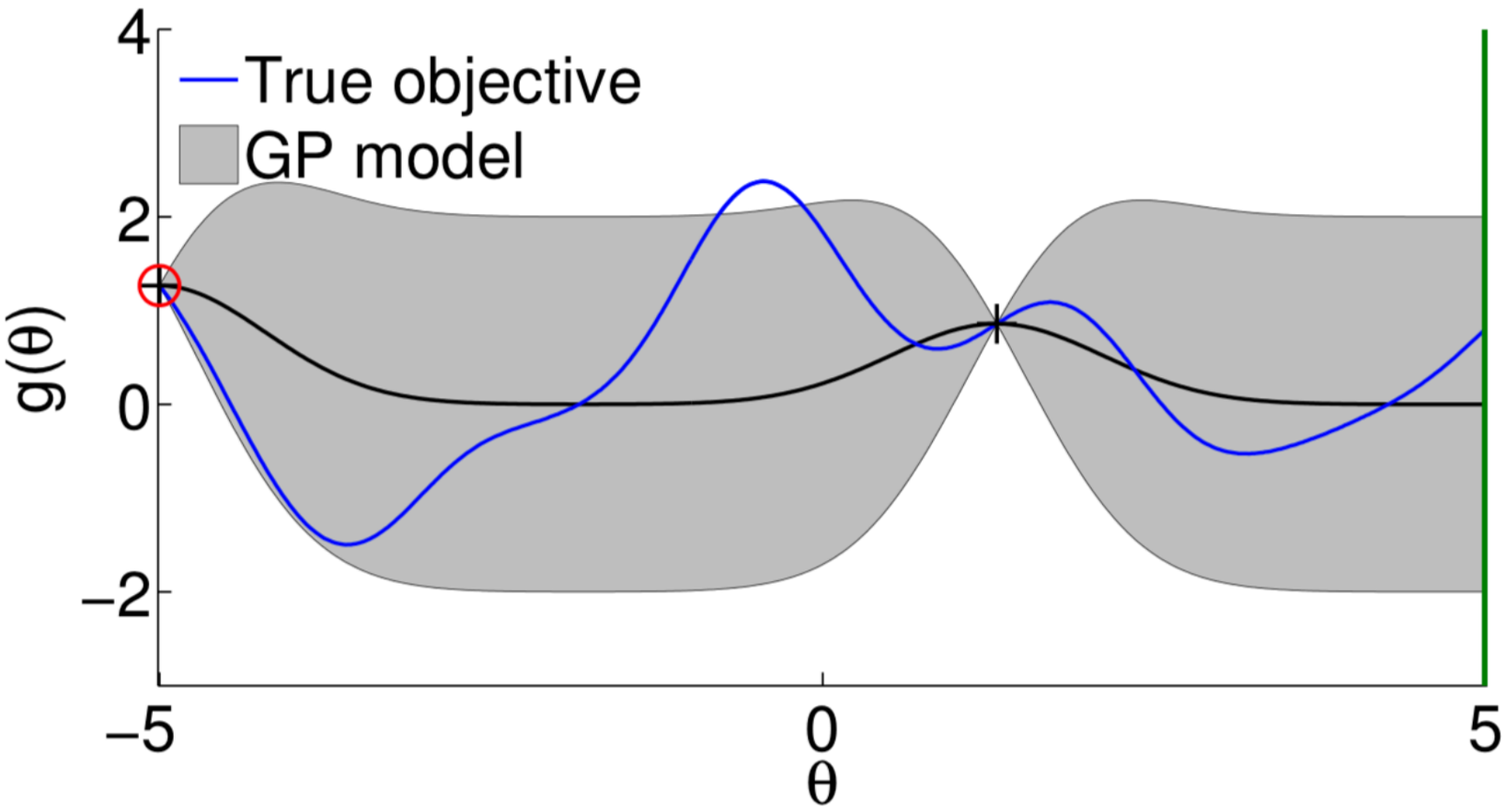
Bayesian Optimization: Illustration



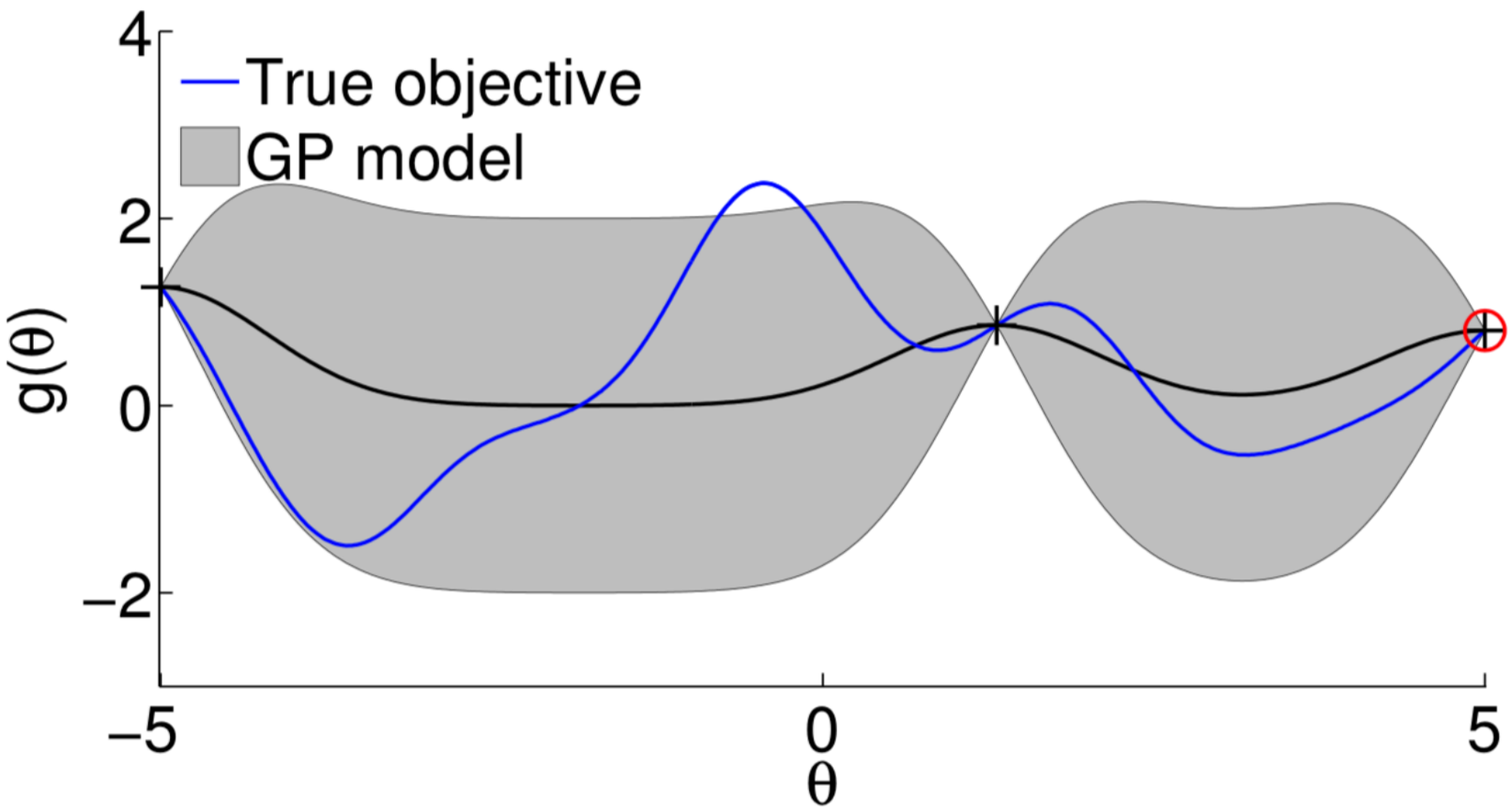
Bayesian Optimization: Illustration



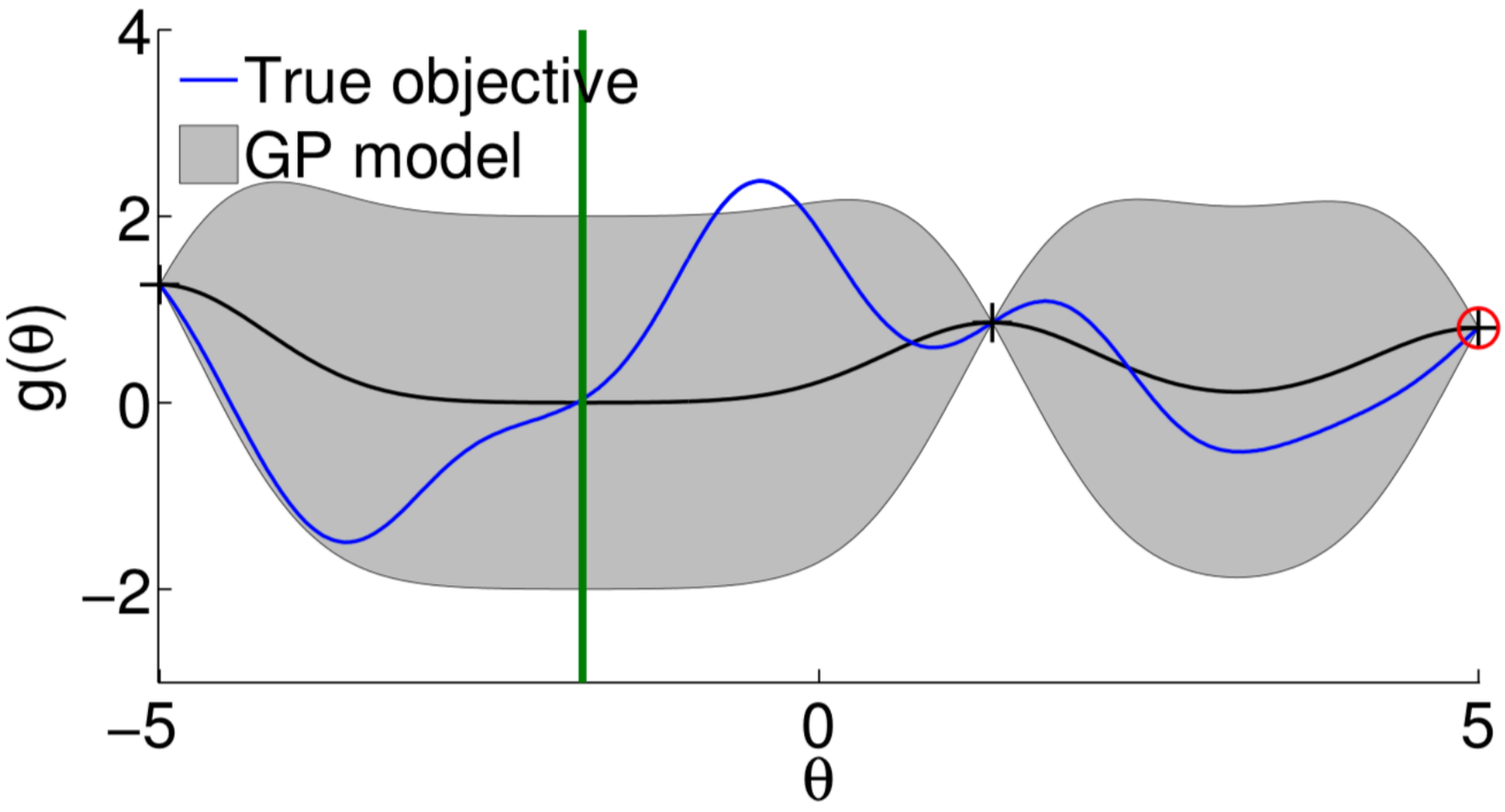
Bayesian Optimization: Illustration



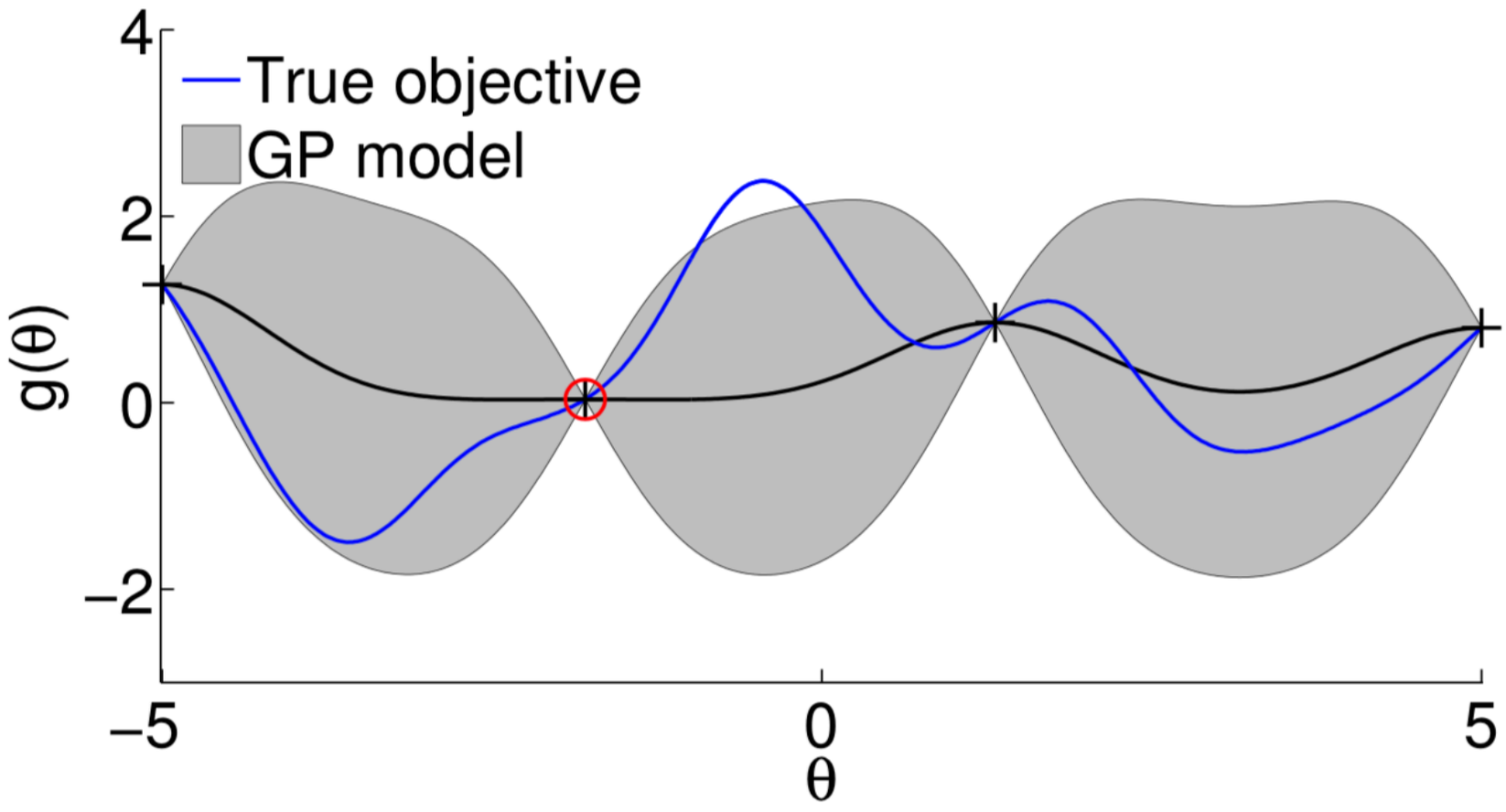
Bayesian Optimization: Illustration



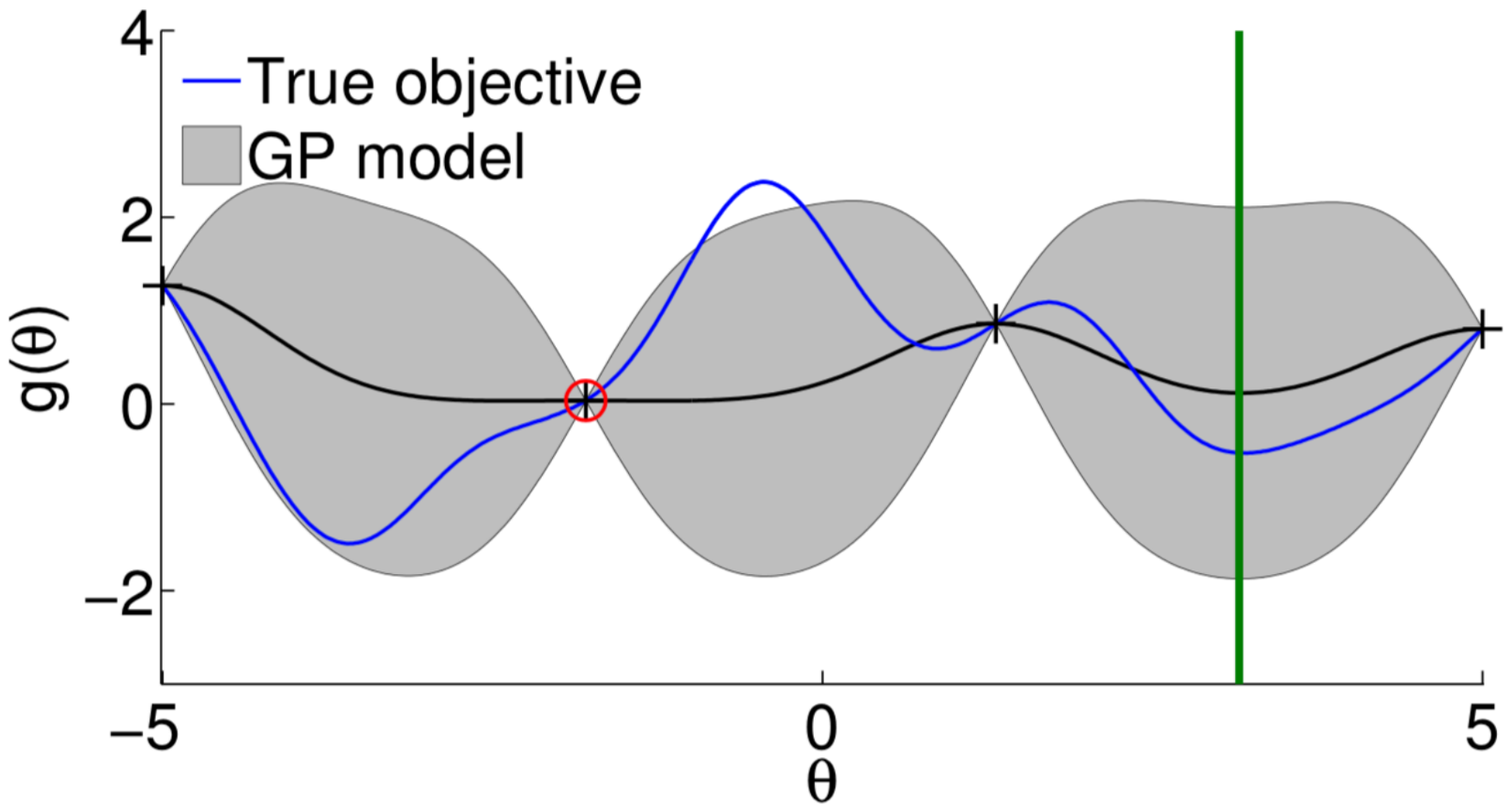
Bayesian Optimization: Illustration



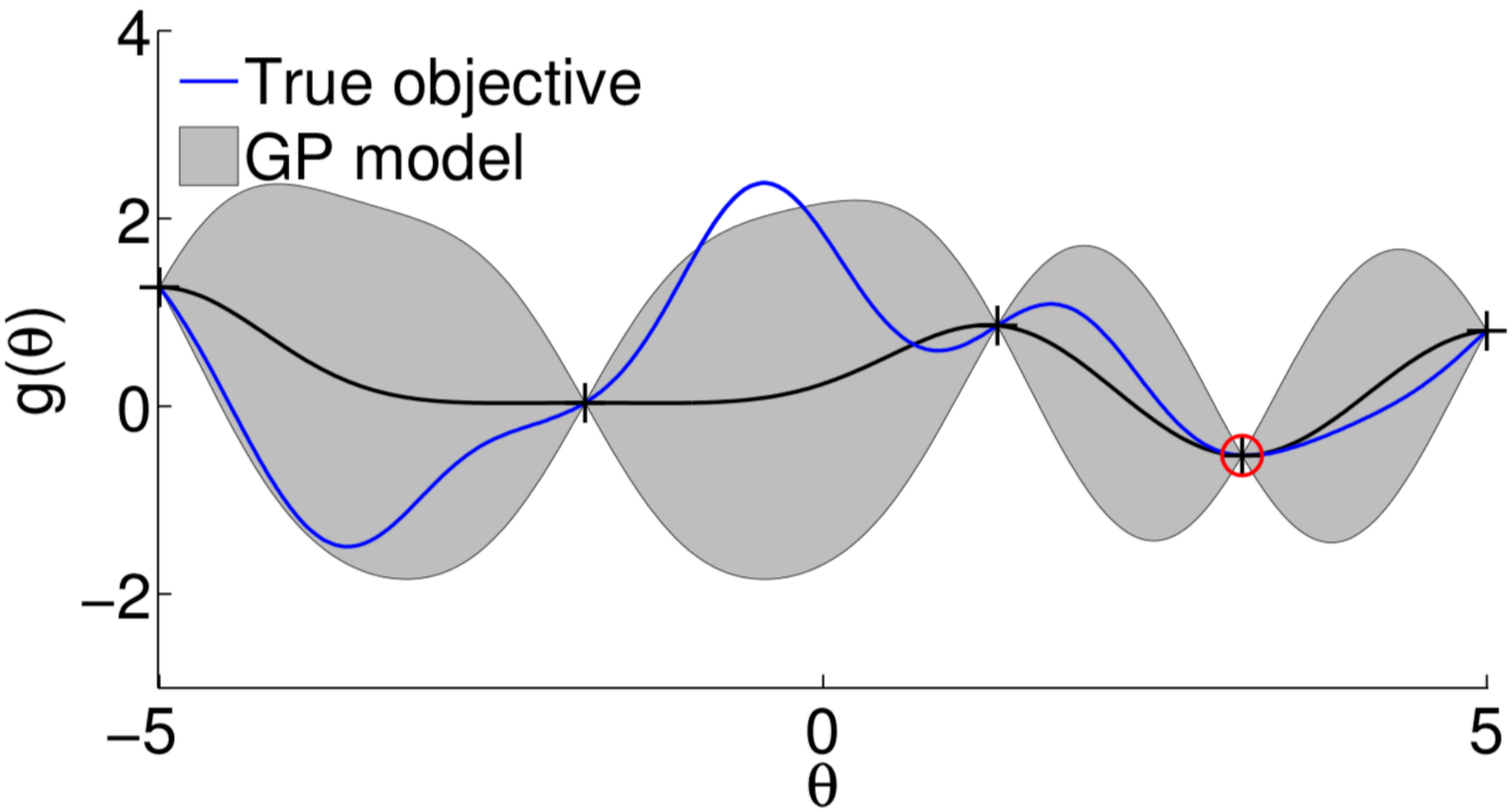
Bayesian Optimization: Illustration



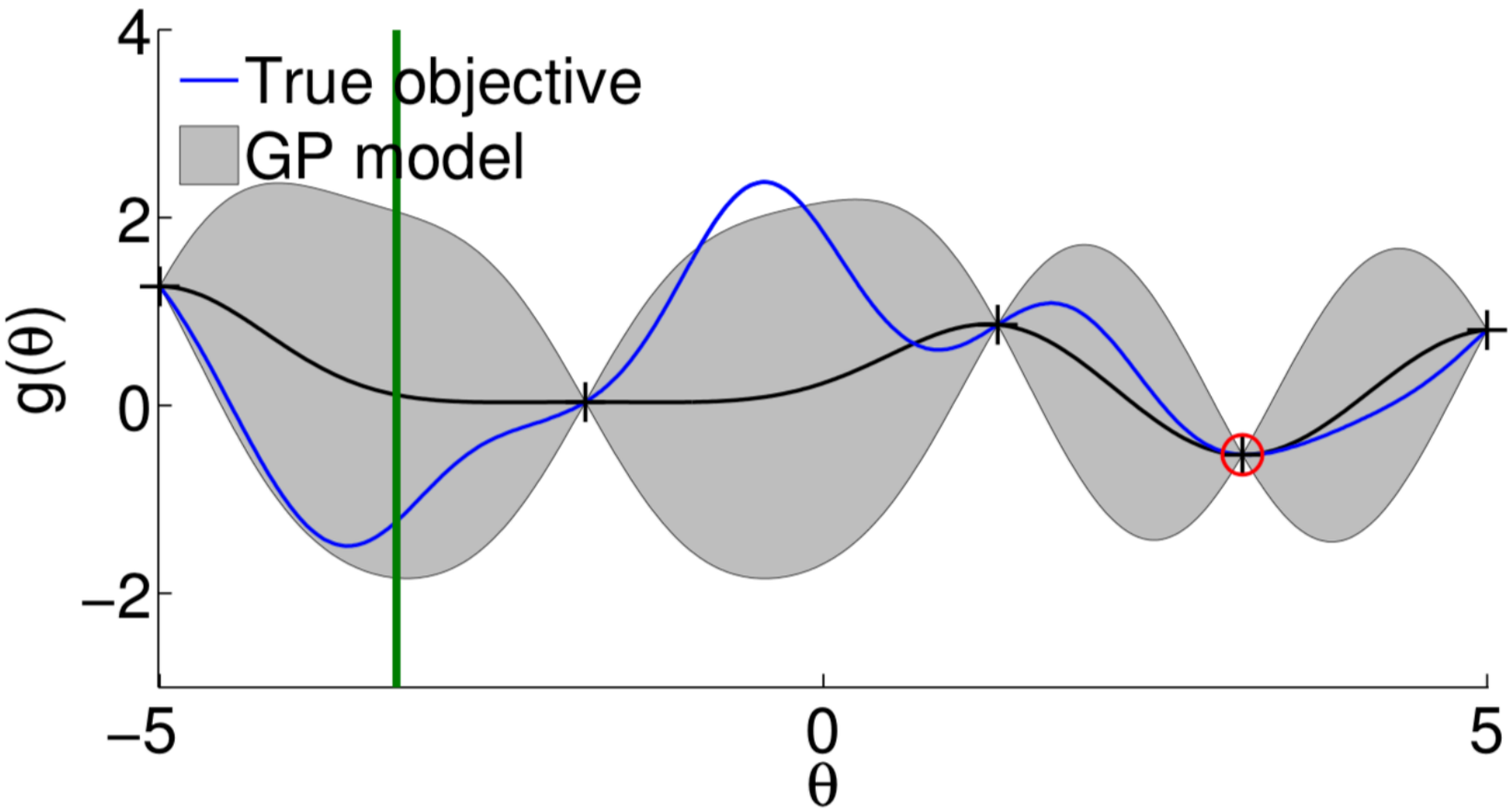
Bayesian Optimization: Illustration



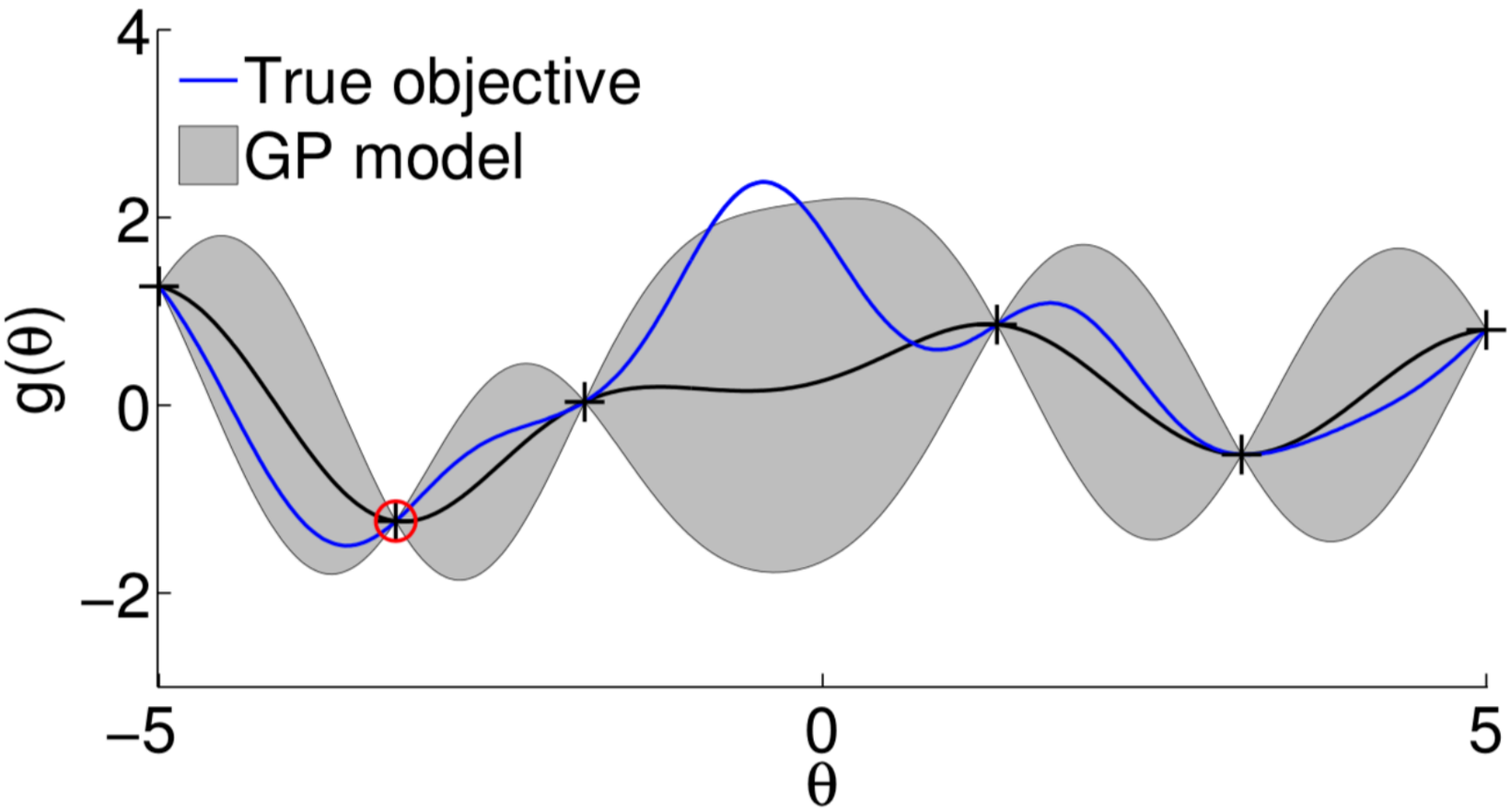
Bayesian Optimization: Illustration



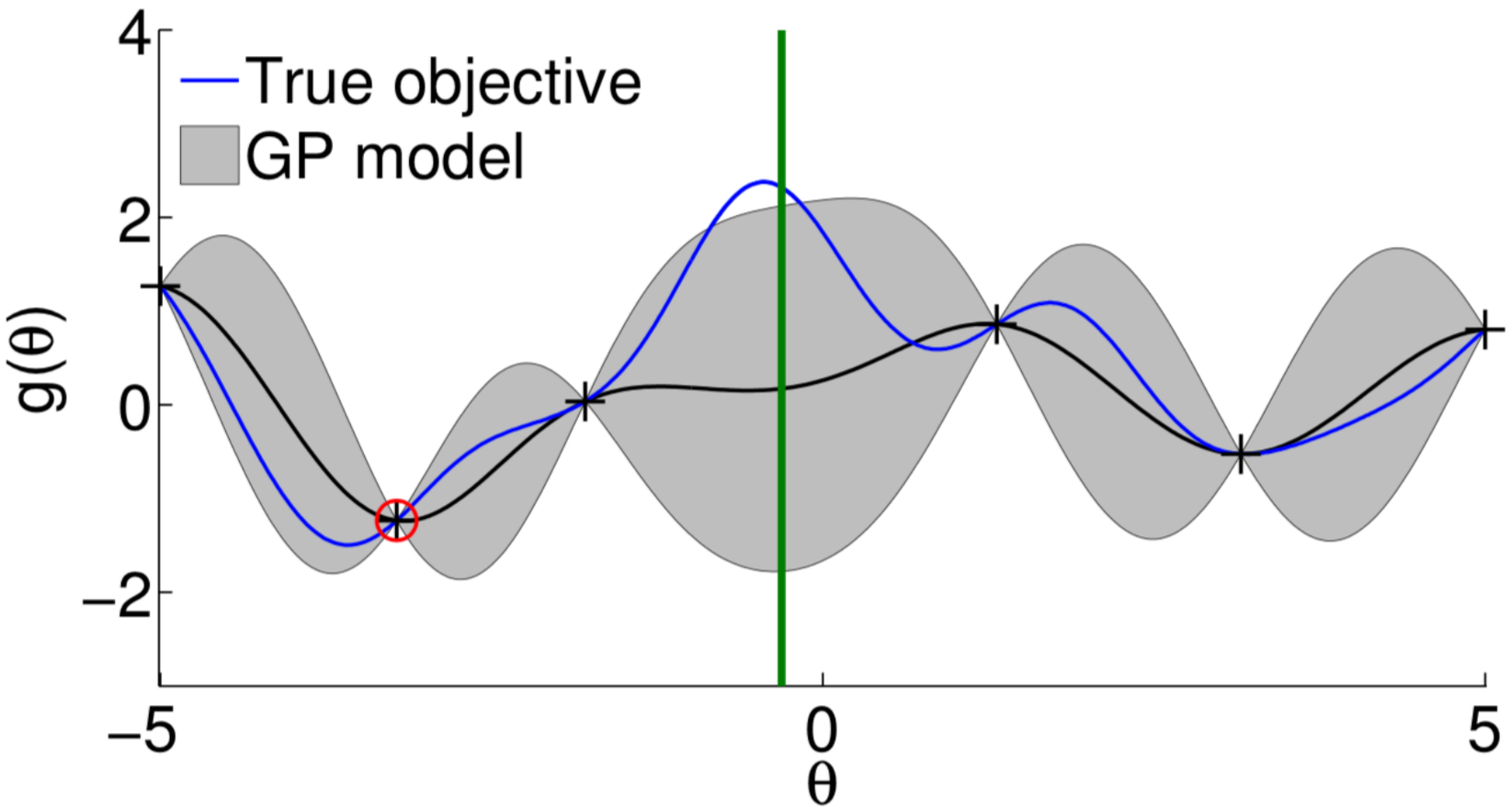
Bayesian Optimization: Illustration



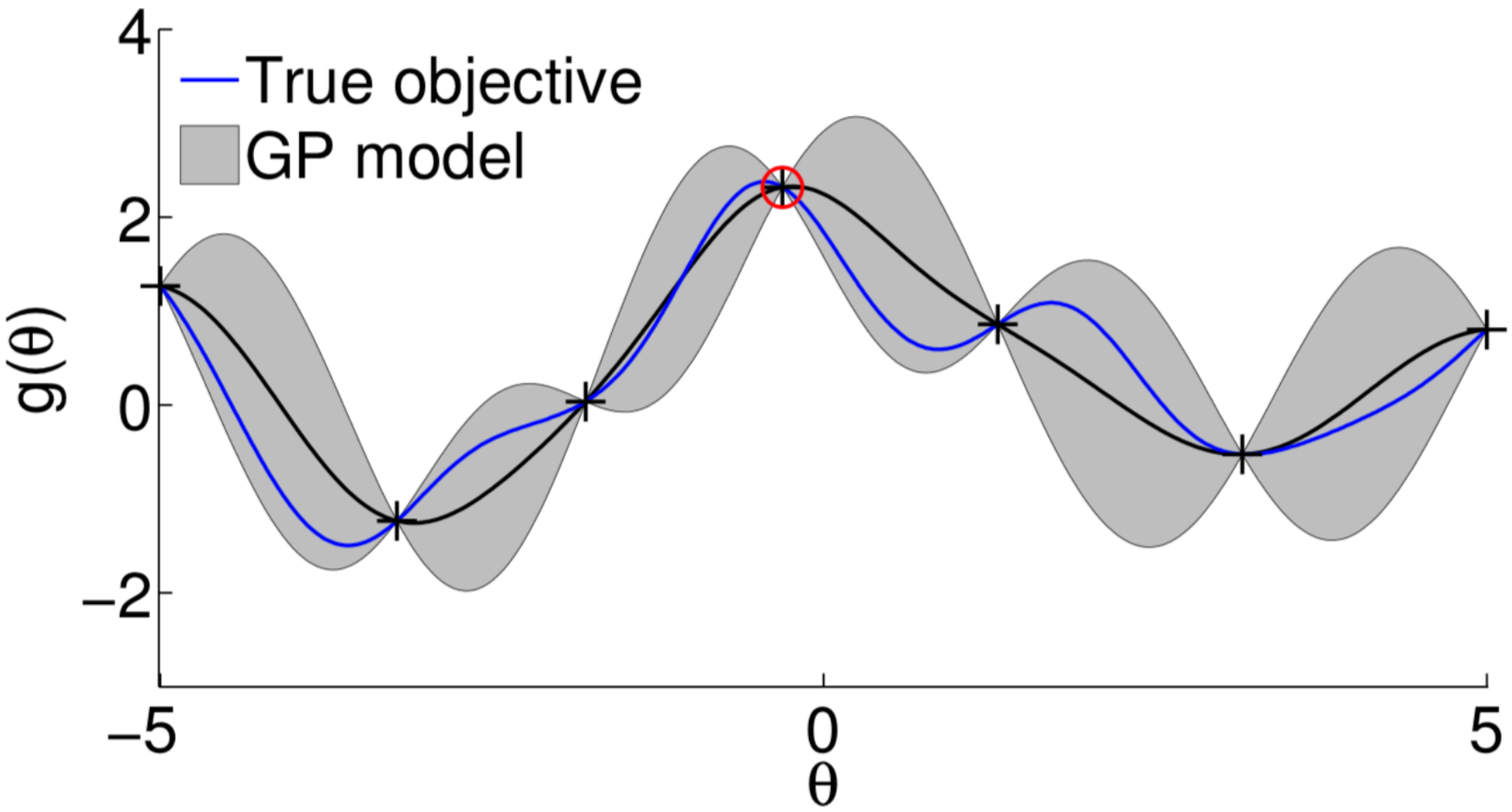
Bayesian Optimization: Illustration



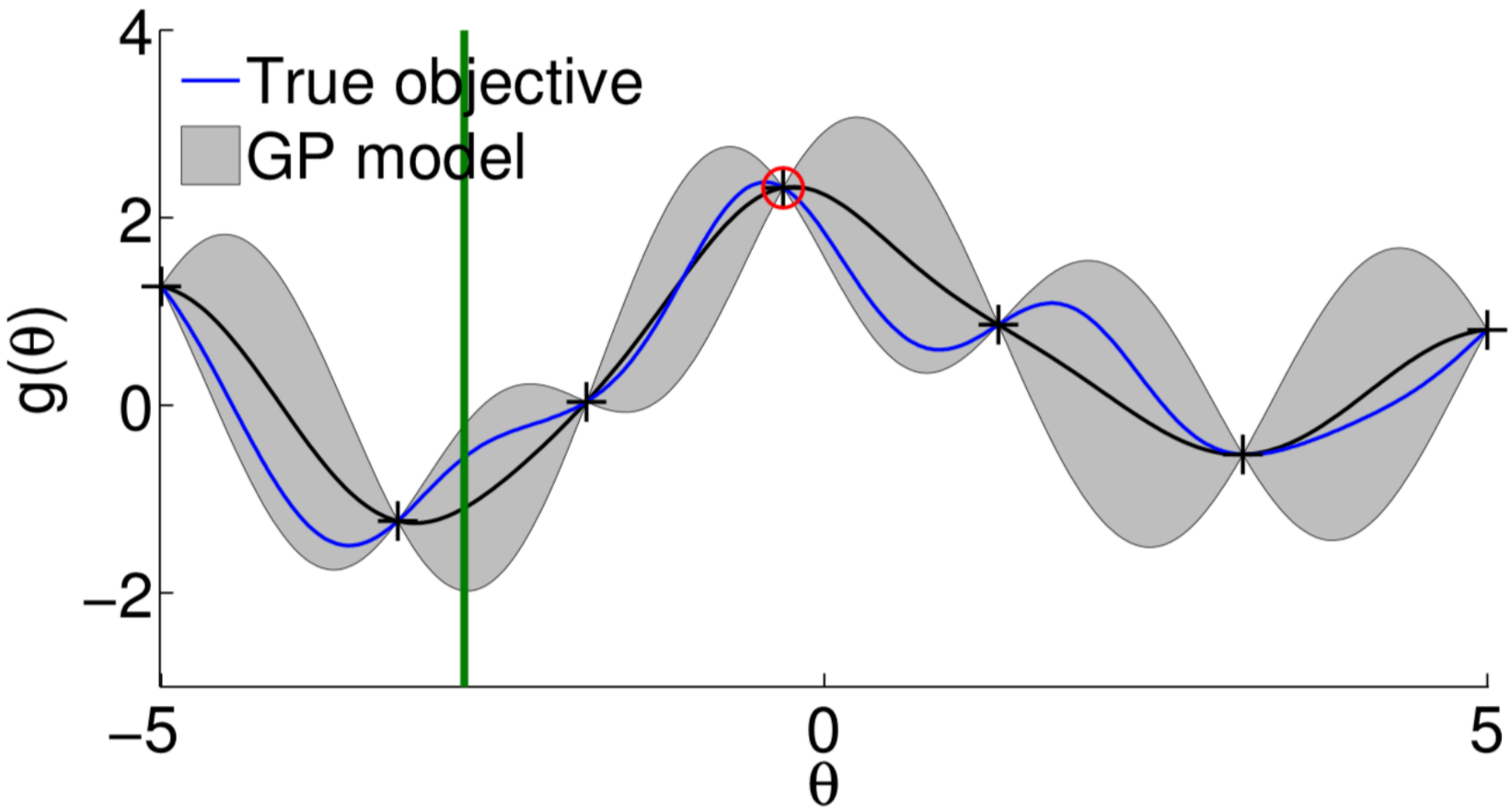
Bayesian Optimization: Illustration



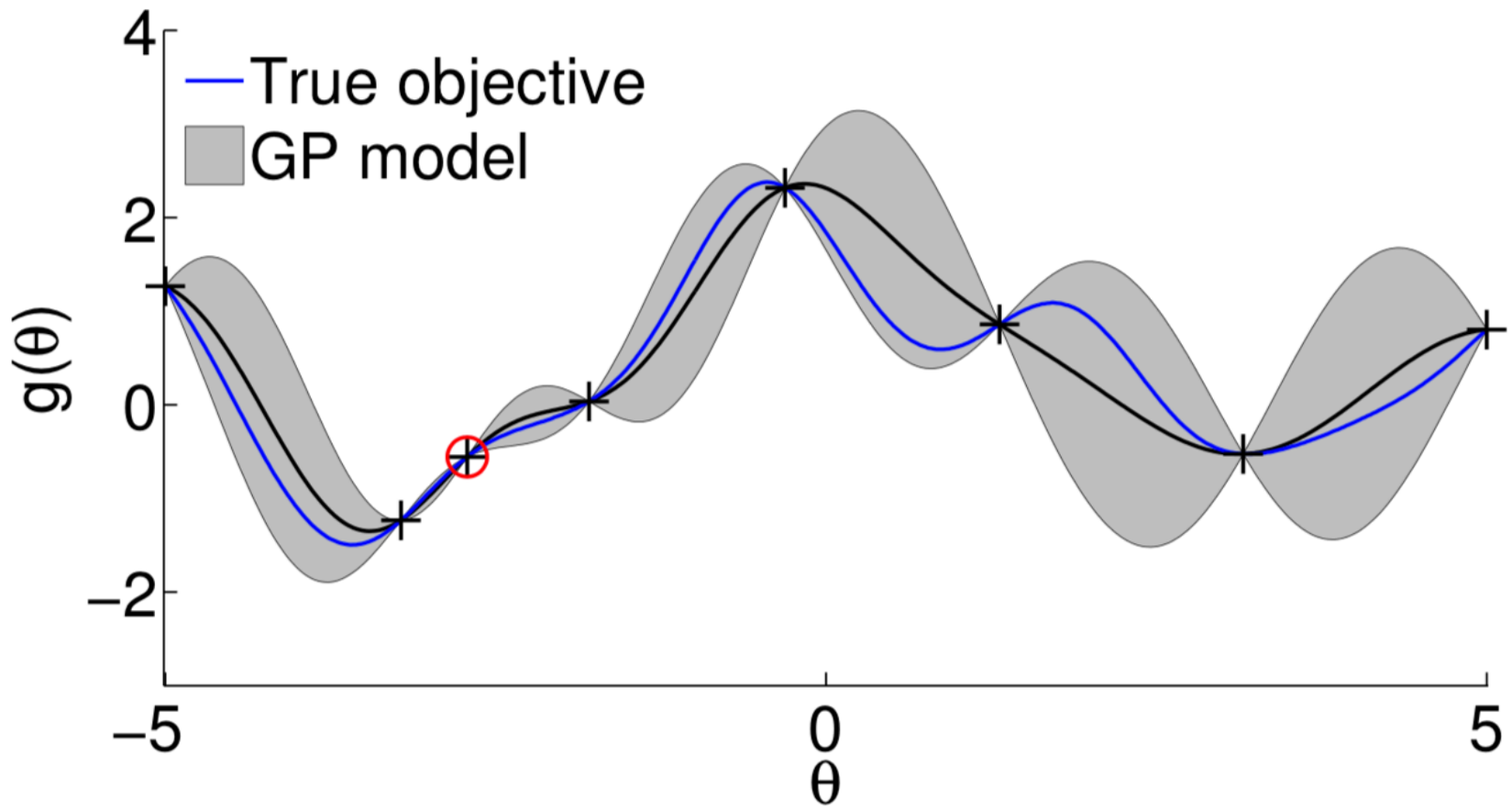
Bayesian Optimization: Illustration



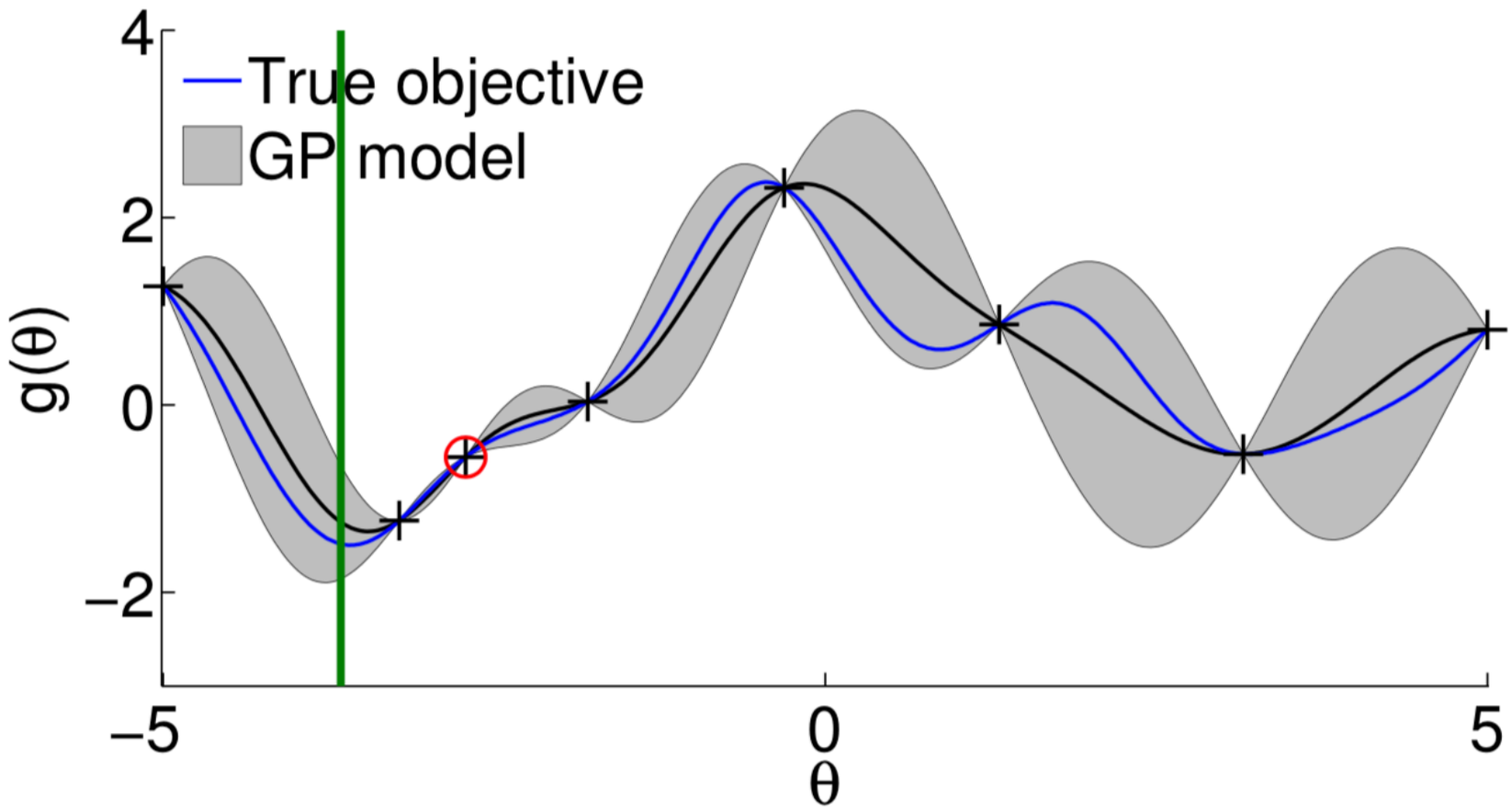
Bayesian Optimization: Illustration



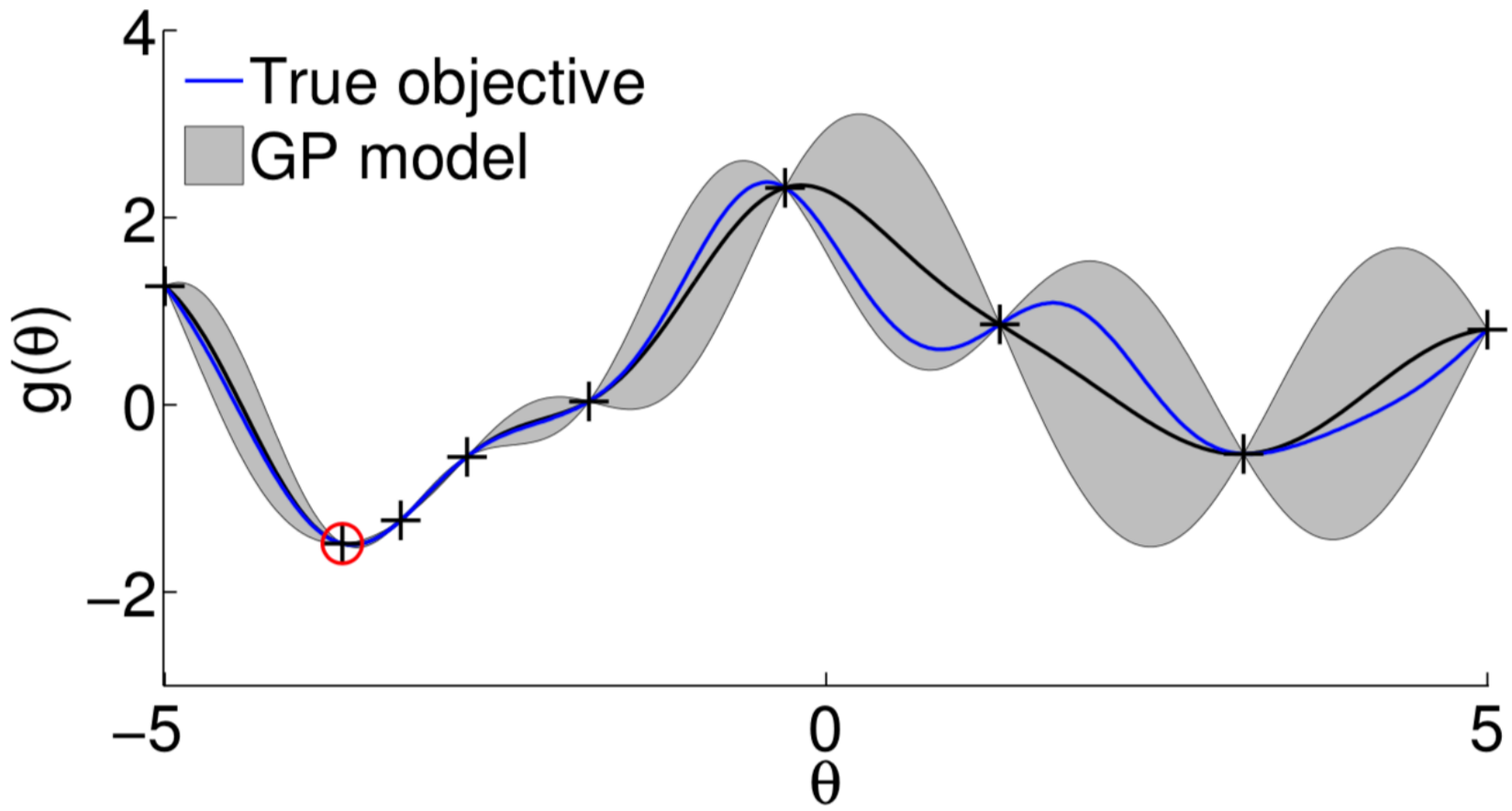
Bayesian Optimization: Illustration



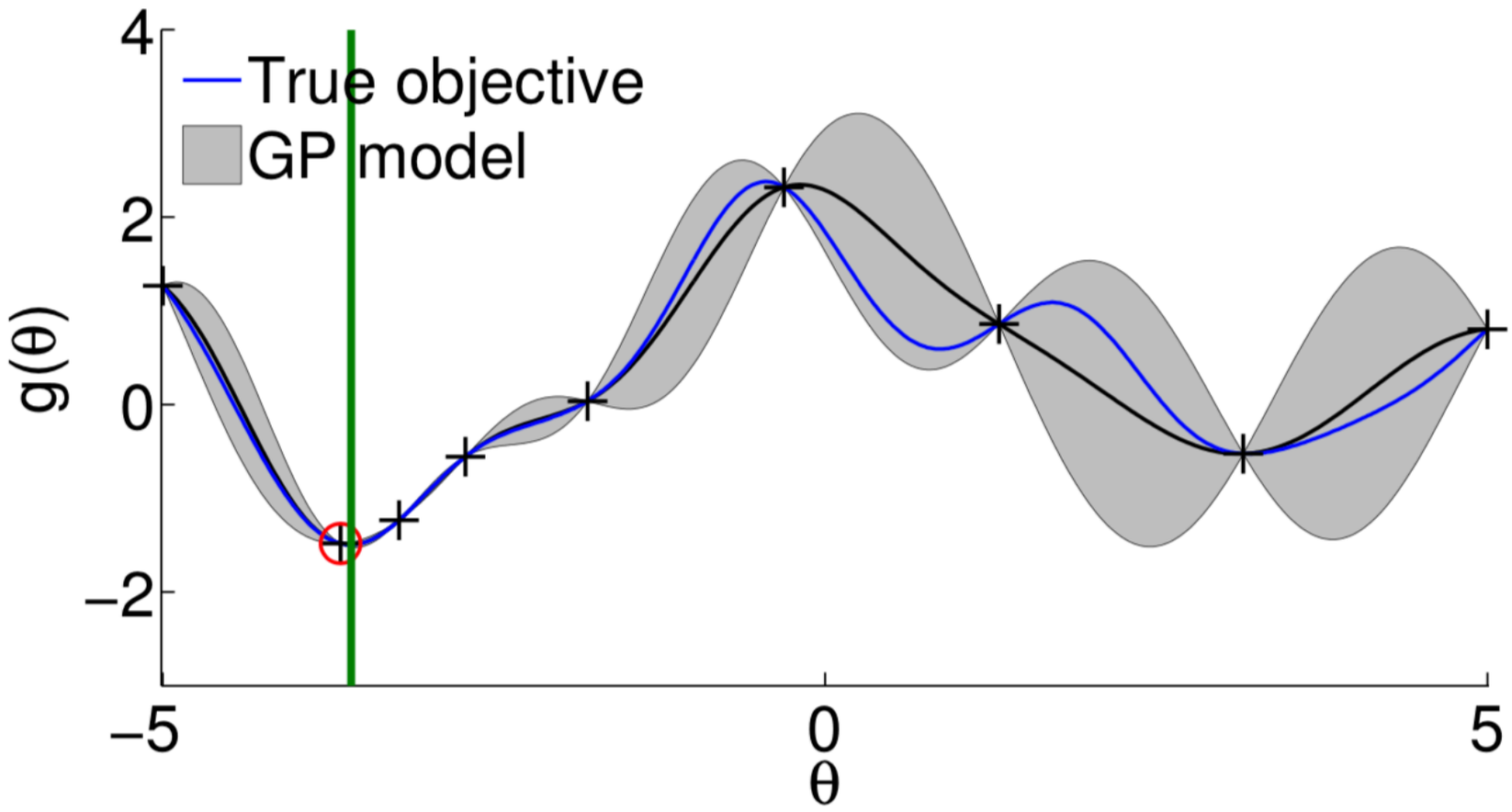
Bayesian Optimization: Illustration



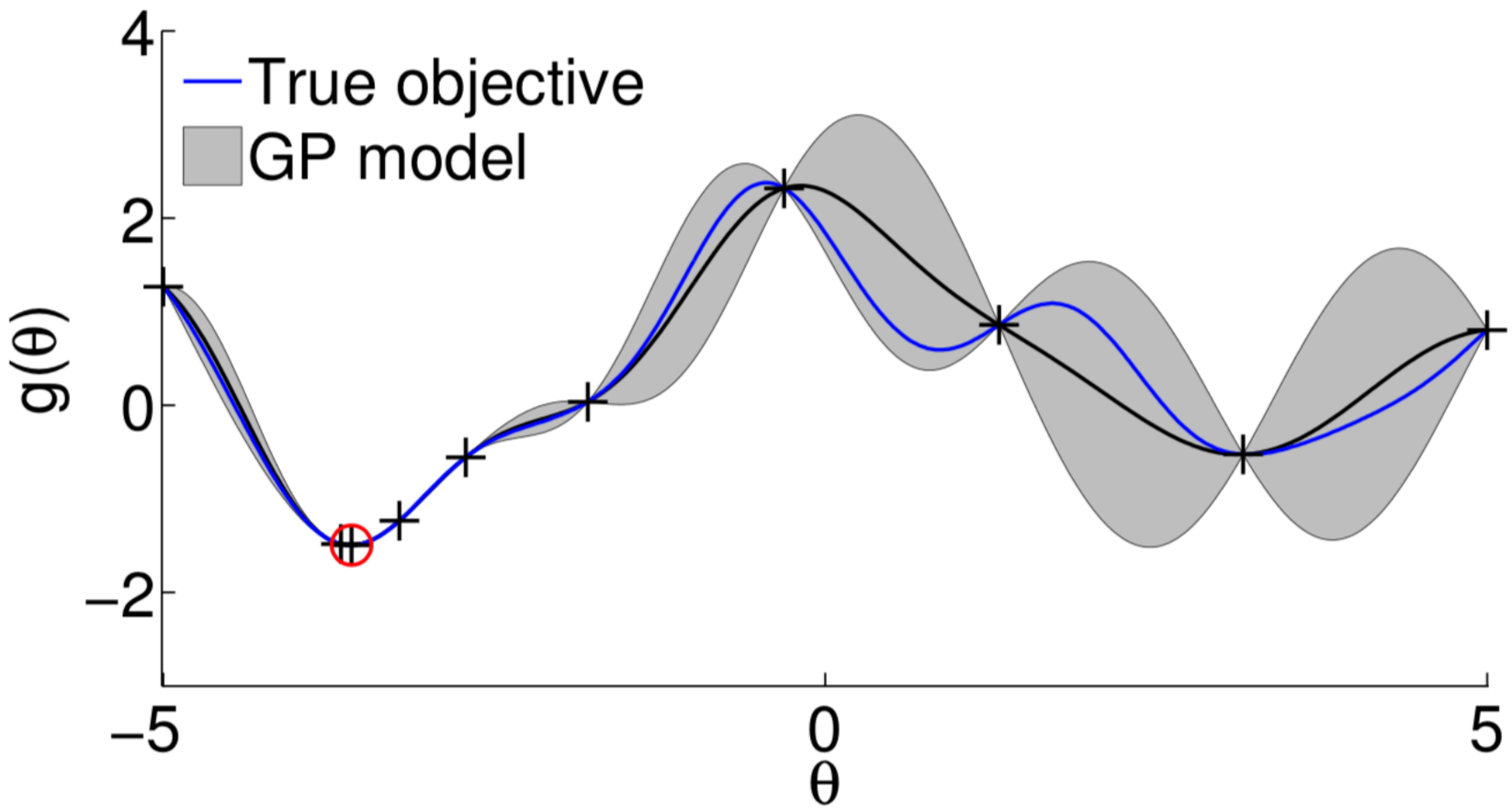
Bayesian Optimization: Illustration



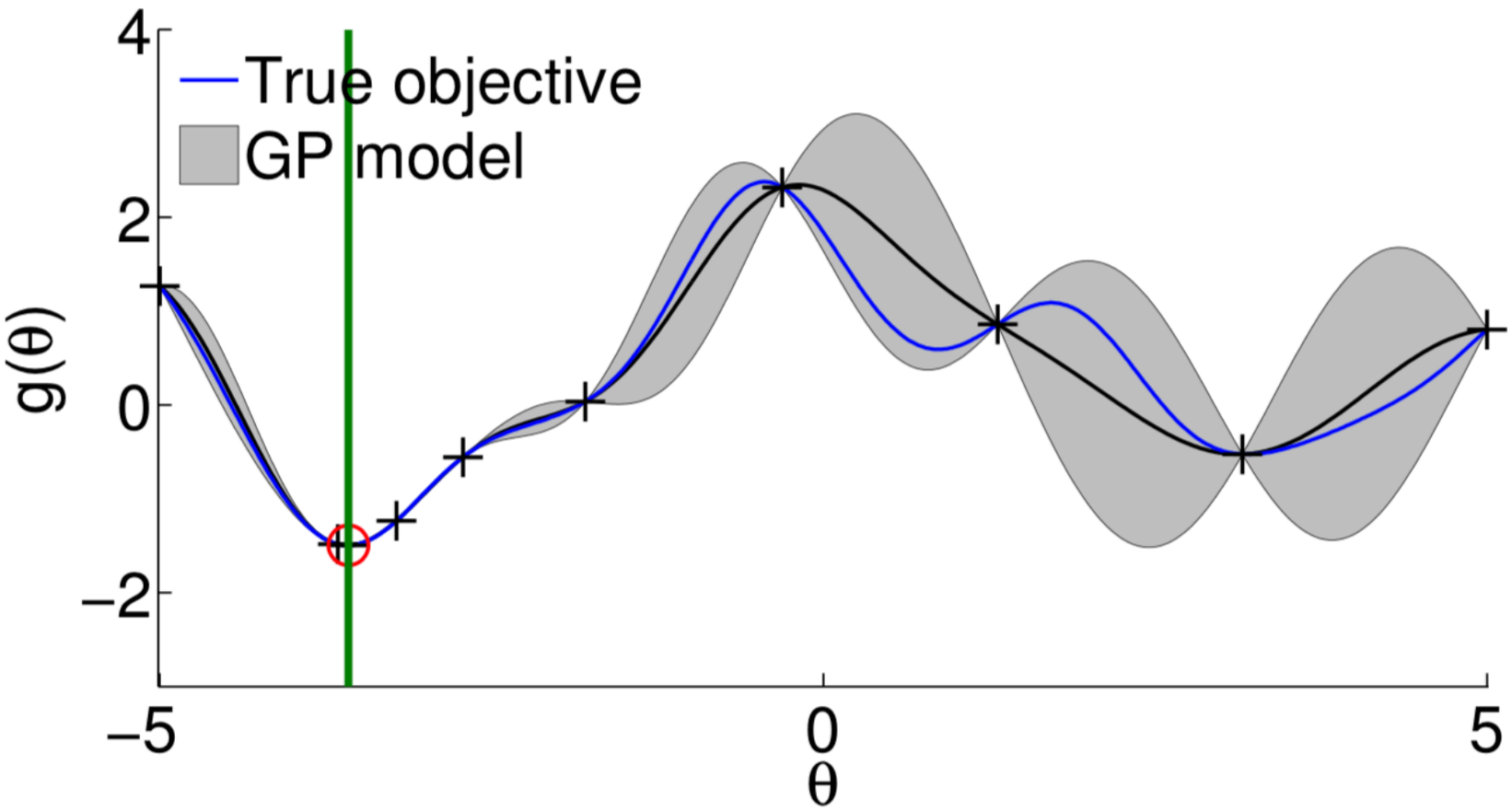
Bayesian Optimization: Illustration



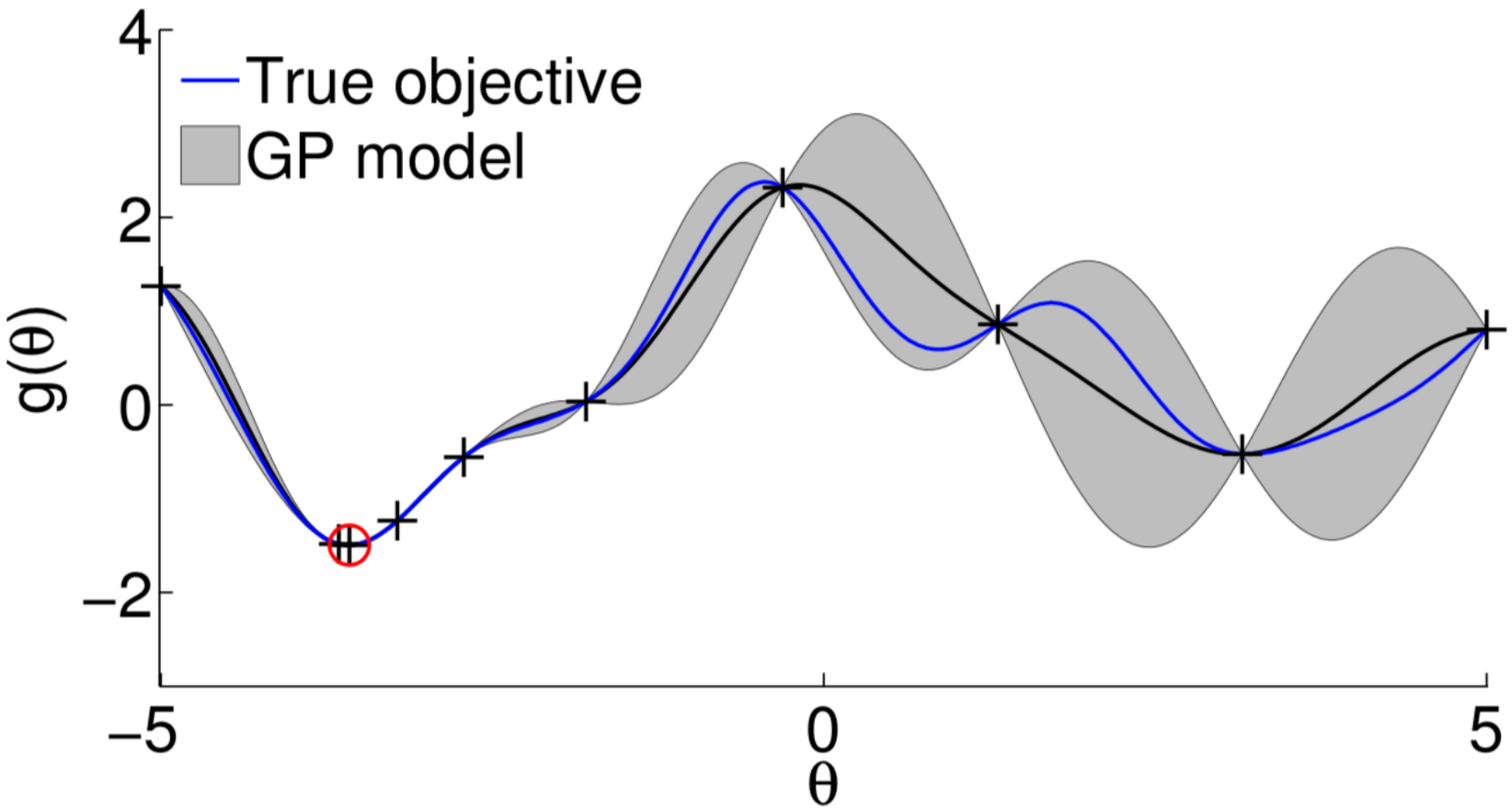
Bayesian Optimization: Illustration



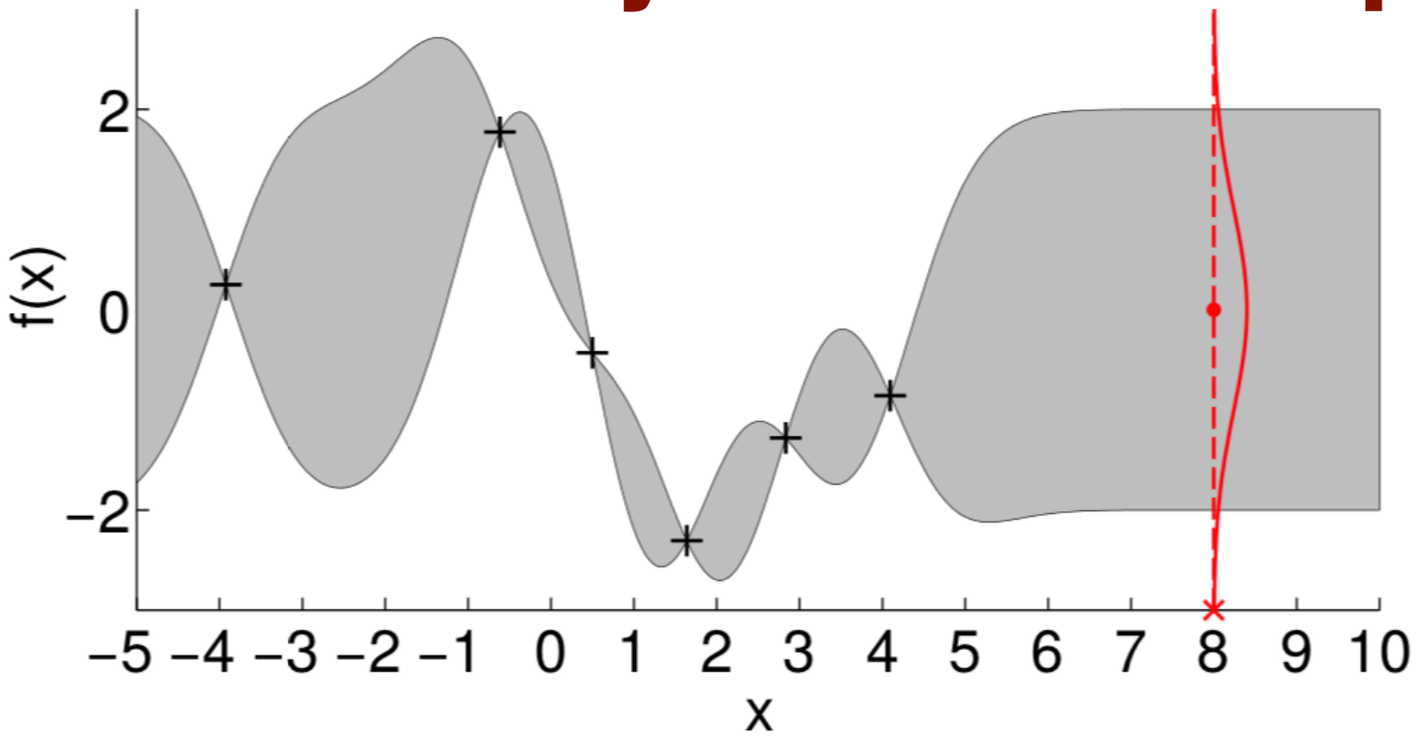
Bayesian Optimization: Illustration



Bayesian Optimization: Illustration

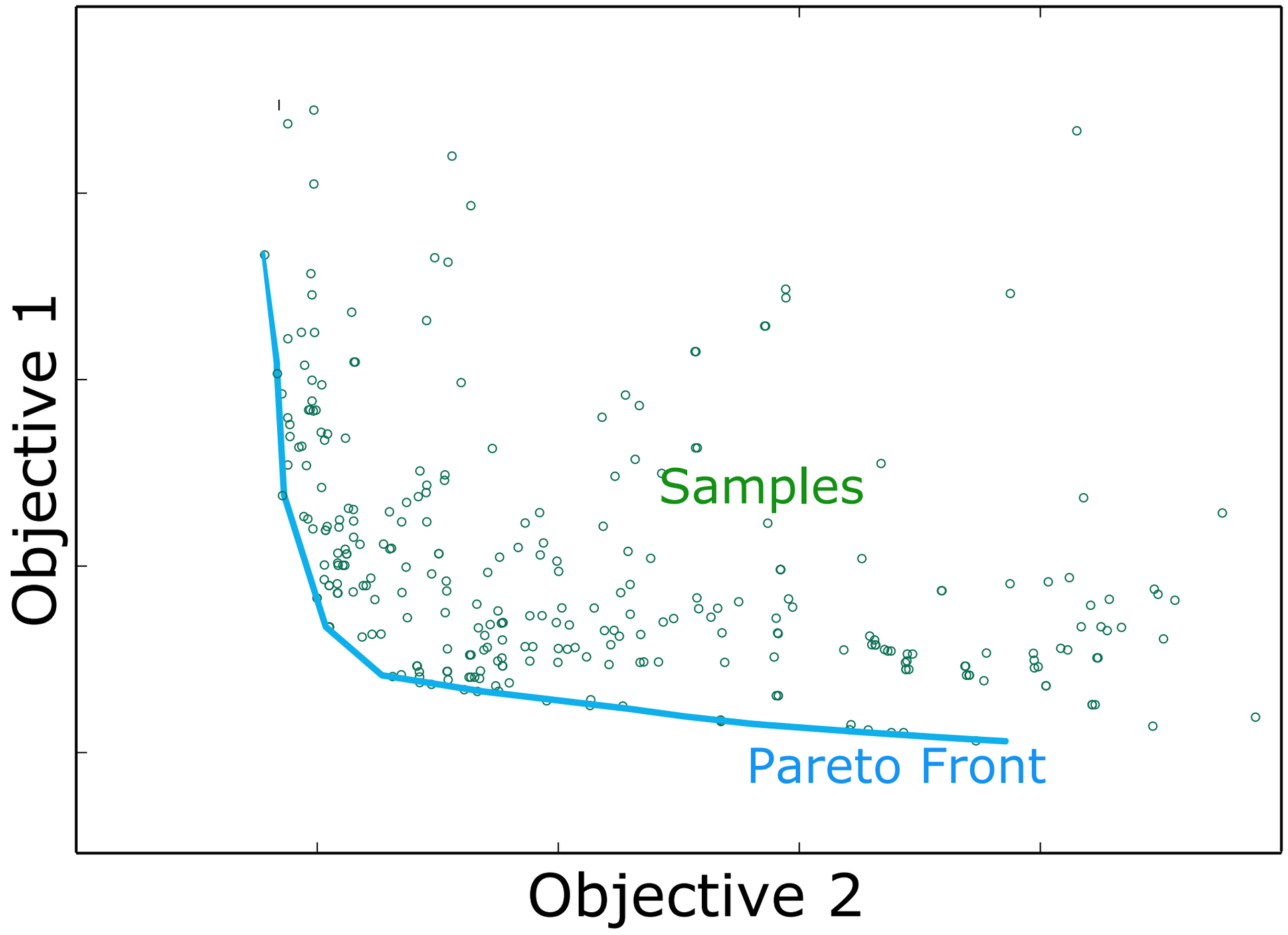


Using Uncertainty in Global Optimization



- Goal: choosing the next point to evaluate the true objective
- Find a good (global) optimum
  - Need to get out of local optima
  - Probabilistic model gives us means and variances
- Trade off exploration and exploitation
  - **Exploration**: seek places with high variance
  - **Exploitation**: seek places with low mean
- **Acquisition function** combines the two

Multi-Objective Goal: Pareto Front



Multi-objective Optimization

Random scalarizations [Paria et al.] adapted for:

- RIOC variables
- Acquisition functions

Given K objectives $f_1(\boldsymbol{x}), ..., f_K(\boldsymbol{x})$ and a weight distribution $\mathcal{L}$

defined as the simplex $\boldsymbol{\lambda} \in \mathbb{R}^K_+$, $||\boldsymbol{\lambda}||_1 = 1$, a random scalarization is

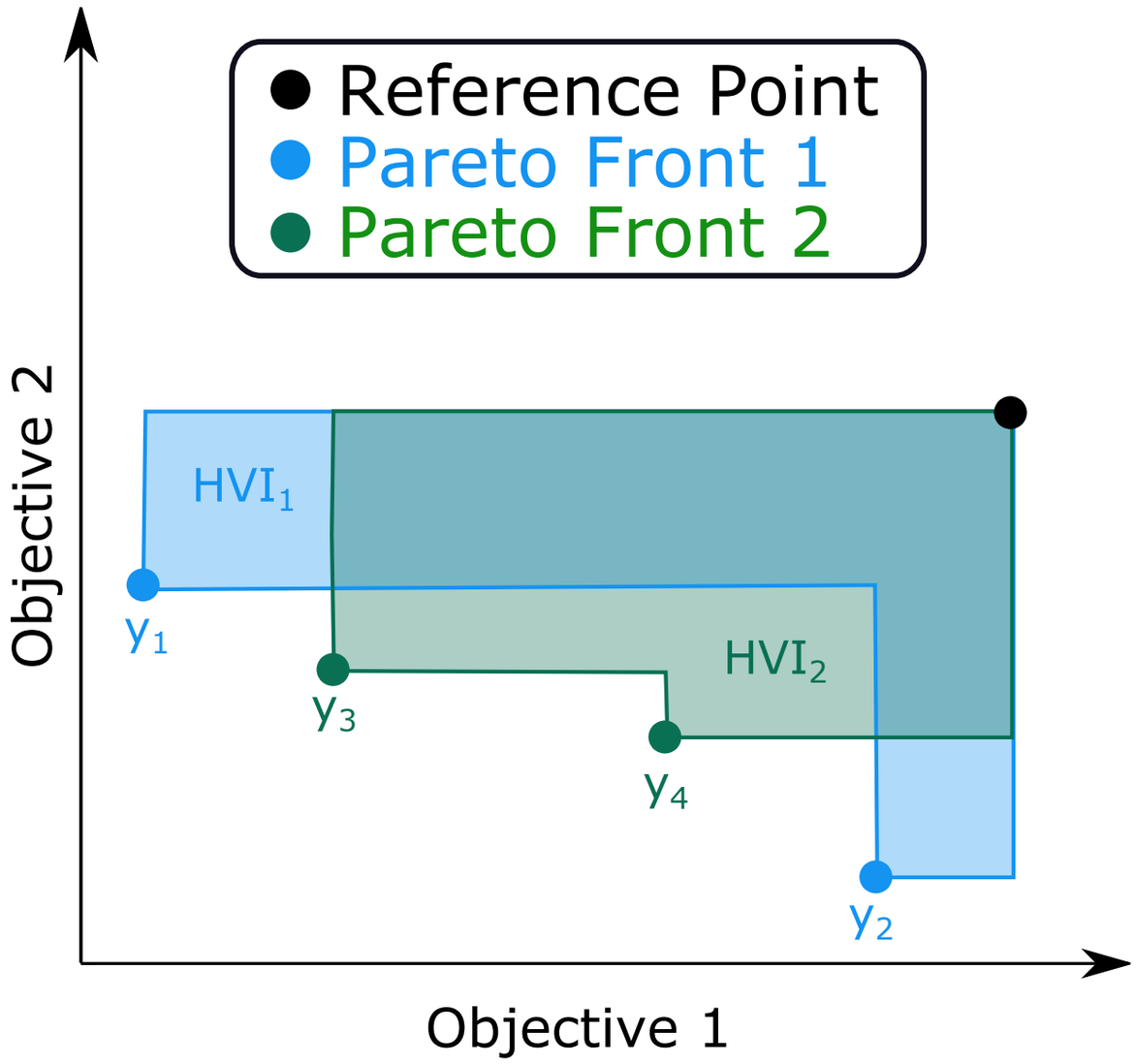
defined as $g(\boldsymbol{\lambda}, \boldsymbol{x})$ where $\boldsymbol{\lambda} \sim \mathcal{L}$

HyperMapper provides two scalarization functions:

linear and Tchebyshev

Linear scalarization: $g_{lin}(\boldsymbol{\lambda}, \boldsymbol{x}) = \sum_{k=1}^{K} \lambda_k f_k(\boldsymbol{x})$

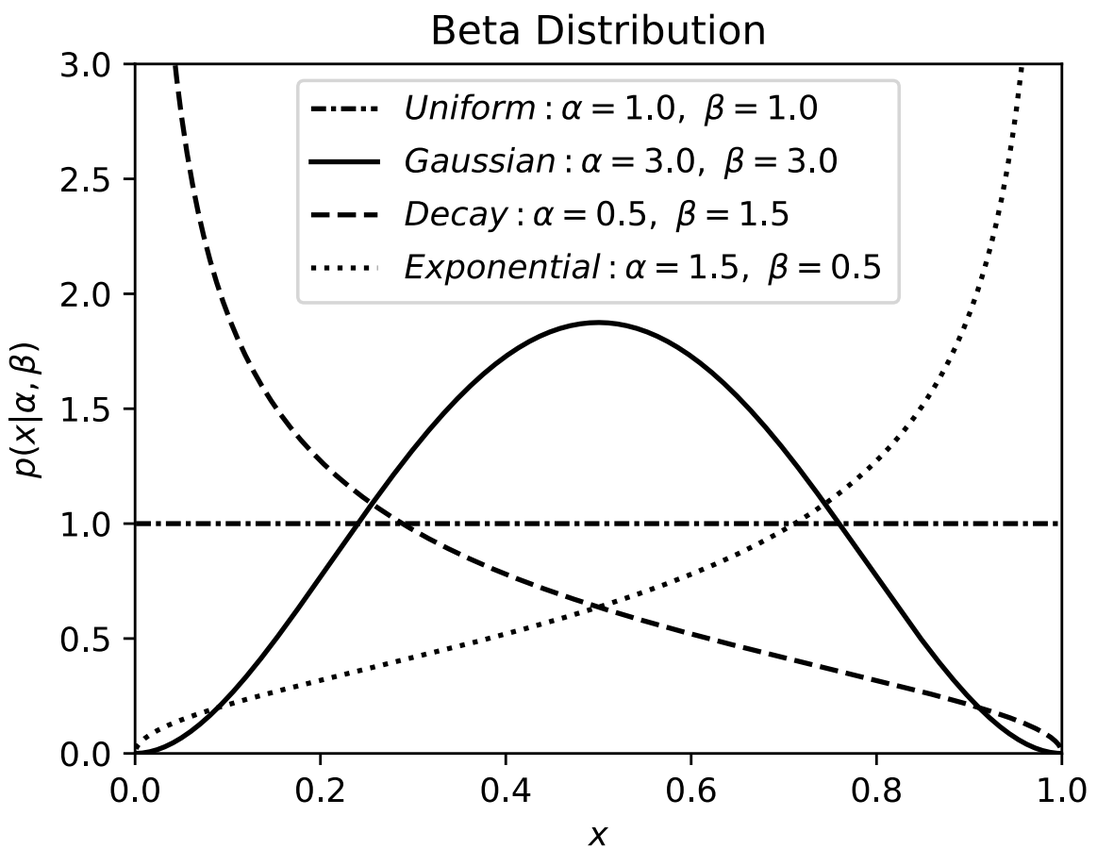
HyperVolume Indicator (HVI)



- Used to compare 2 Paretos
- HVI is a hypervolume
  - with respect to a reference
- It is always a scalar
  - even with multiple objectives
- The lower the better:
  HVI_{total} - HVI_i

Injecting Prior Knowledge

- Need a probability distribution that:
  1. Has a finite domain
  2. Can flexibly model shapes including:
     Bell-shape and J-shape
- Beta distribution - parameters alpha and beta
- In addition, need of a discrete distribution for categorical variables



PDF given by:

$$f(x|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$$

for $x \in [0,1]$ and $\alpha, \beta > 0$,

where Γ is the Gamma function

Probabilistic Model in HyperMapper

- Predicts an outcome given an input x

Probabilistic Model in HyperMapper

- Predicts an outcome given an input x
- One surrogate model per objective
  - Random forests (RF) for regression

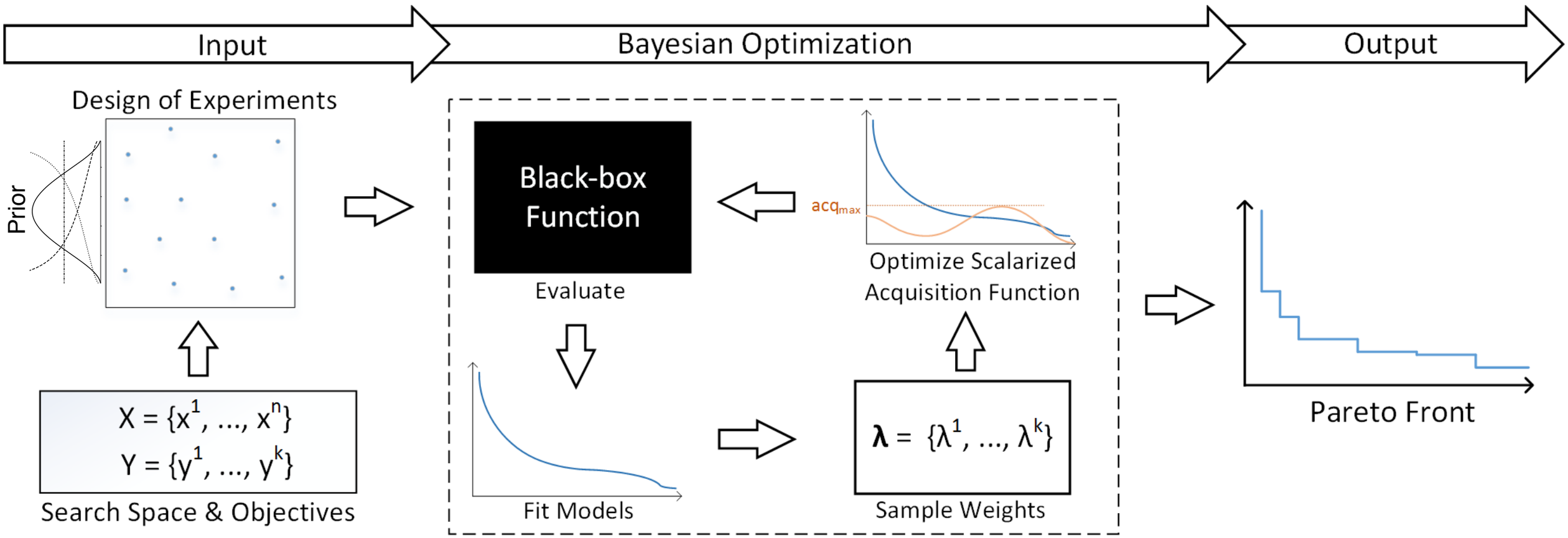
Probabilistic Model in HyperMapper

- Predicts an outcome given an input x
- One surrogate model per objective
  - Random forests (RF) for regression
- Intuition of why RF is a good model:
  - Good at non-linearity, multi-modality and non-smoothness

Probabilistic Model in HyperMapper

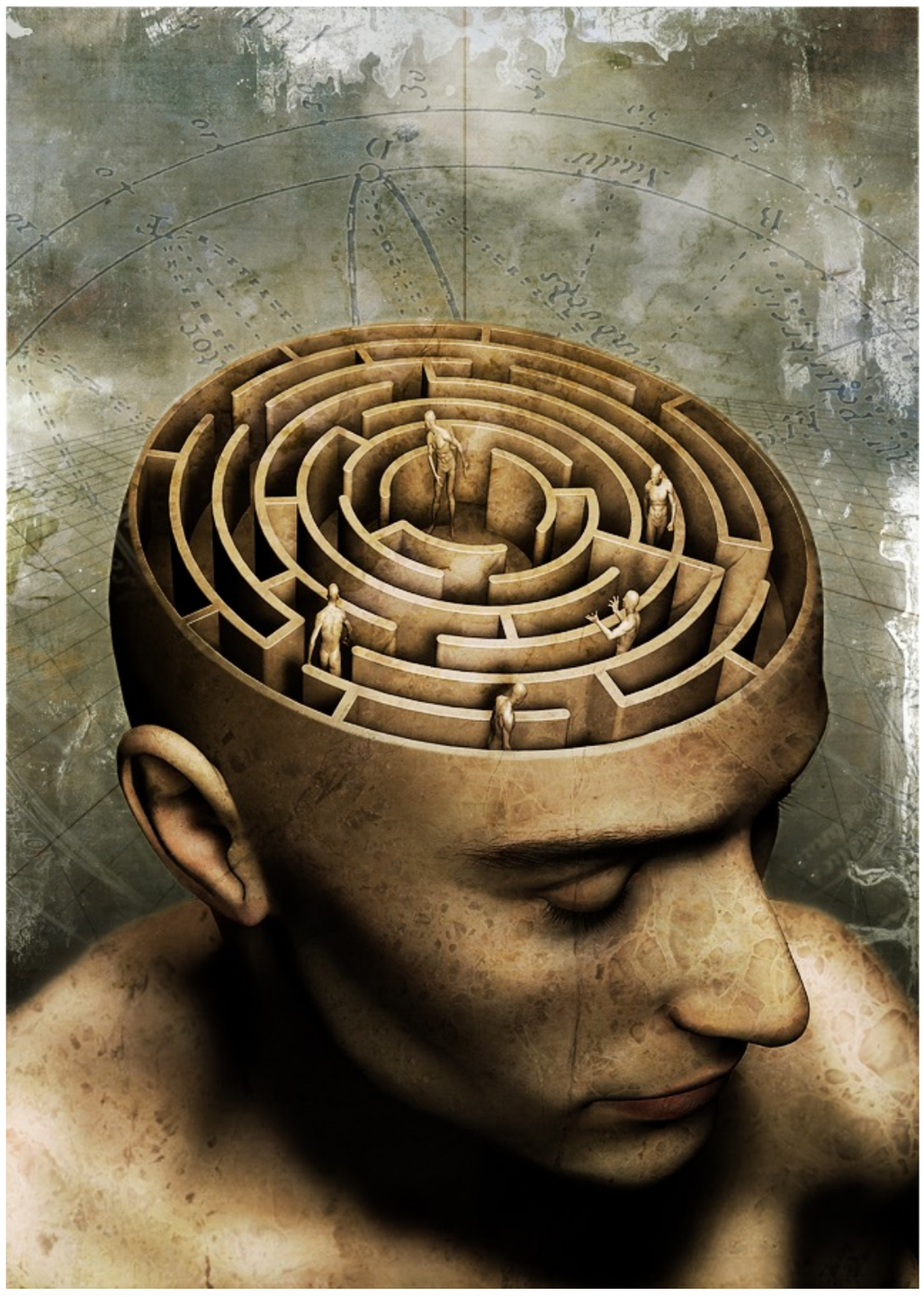
- Predicts an outcome given an input x
- One surrogate model per objective
  - Random forests (RF) for regression
- Intuition of why RF is a good model:
  - Good at non-linearity, multi-modality and non-smoothness
- The model has not to be perfect
  - Interested in optima not the best model
  - Coefficient of determination (R^2) is the wrong indicator of success

The HyperMapper Framework

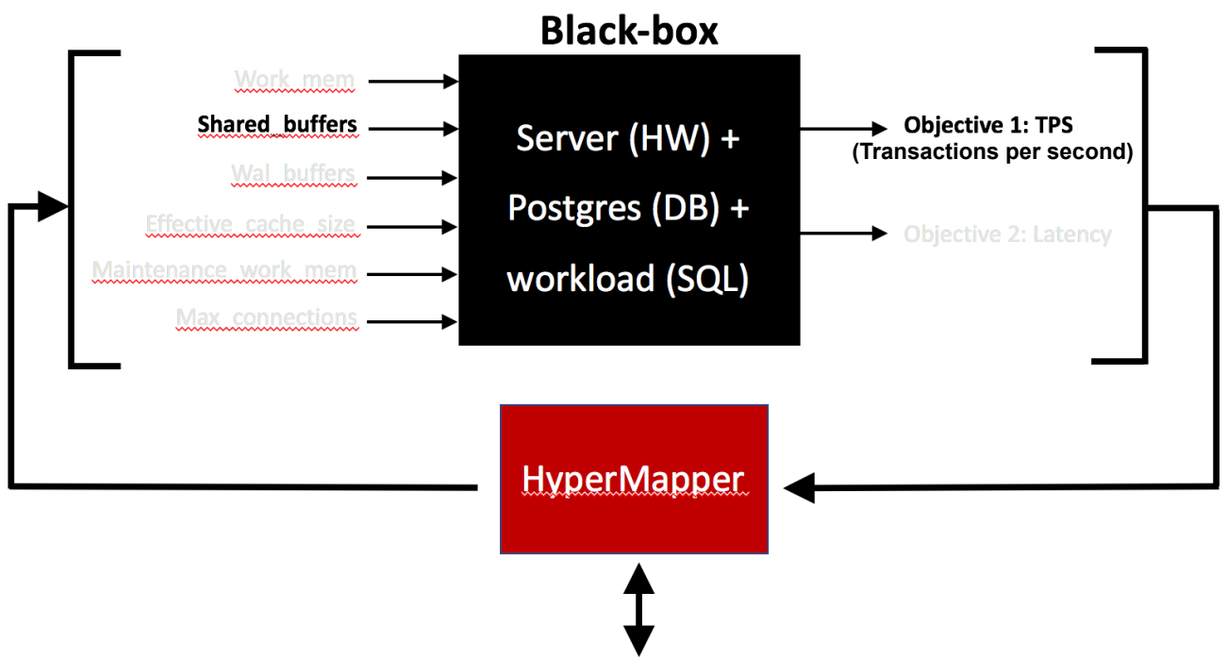


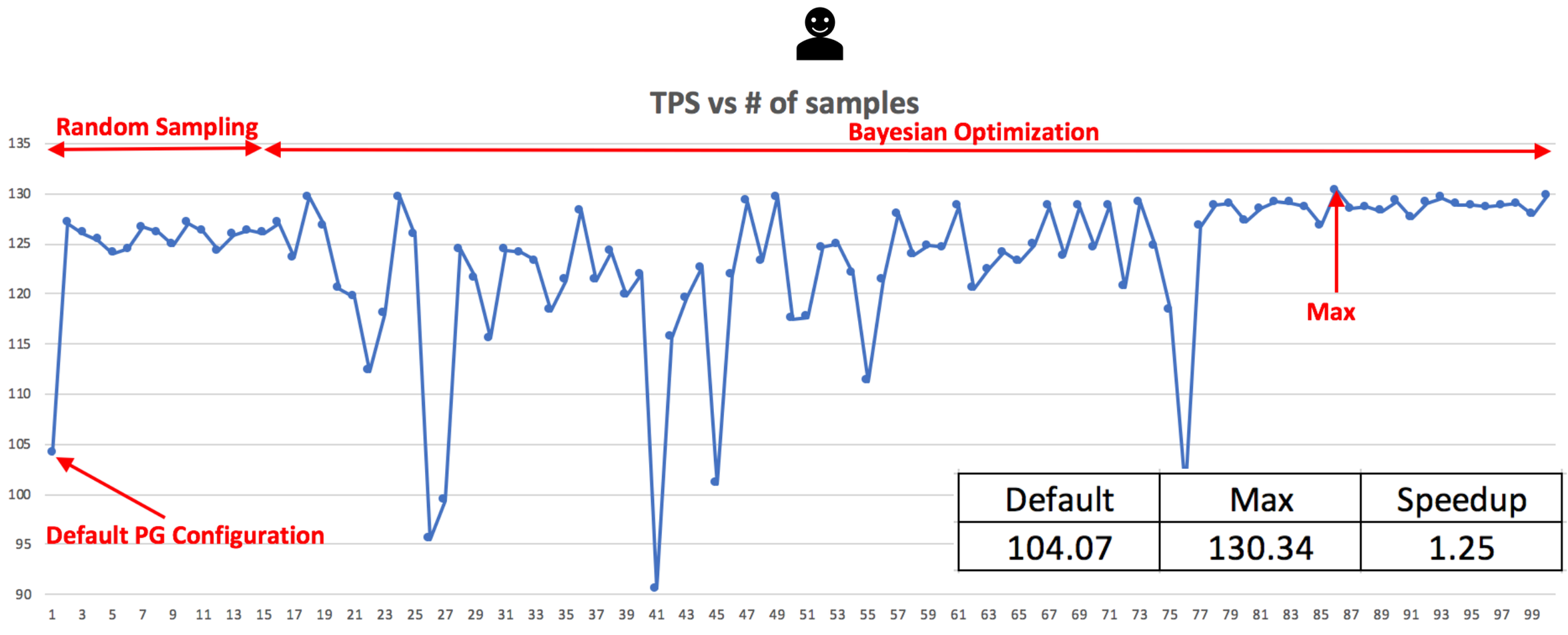
Outline

1. Black-box Optimization
   - Problem Setting
   - Motivation Examples
   - Taxonomy
2. The HyperMapper Framework
   - Bayesian Optimization
   - Pareto and Hypervolume Indicator
   - Prior Distribution
3. **Experimental Results**
   - **Hardware Design**
   - **Computer Vision**
   - **Databases**

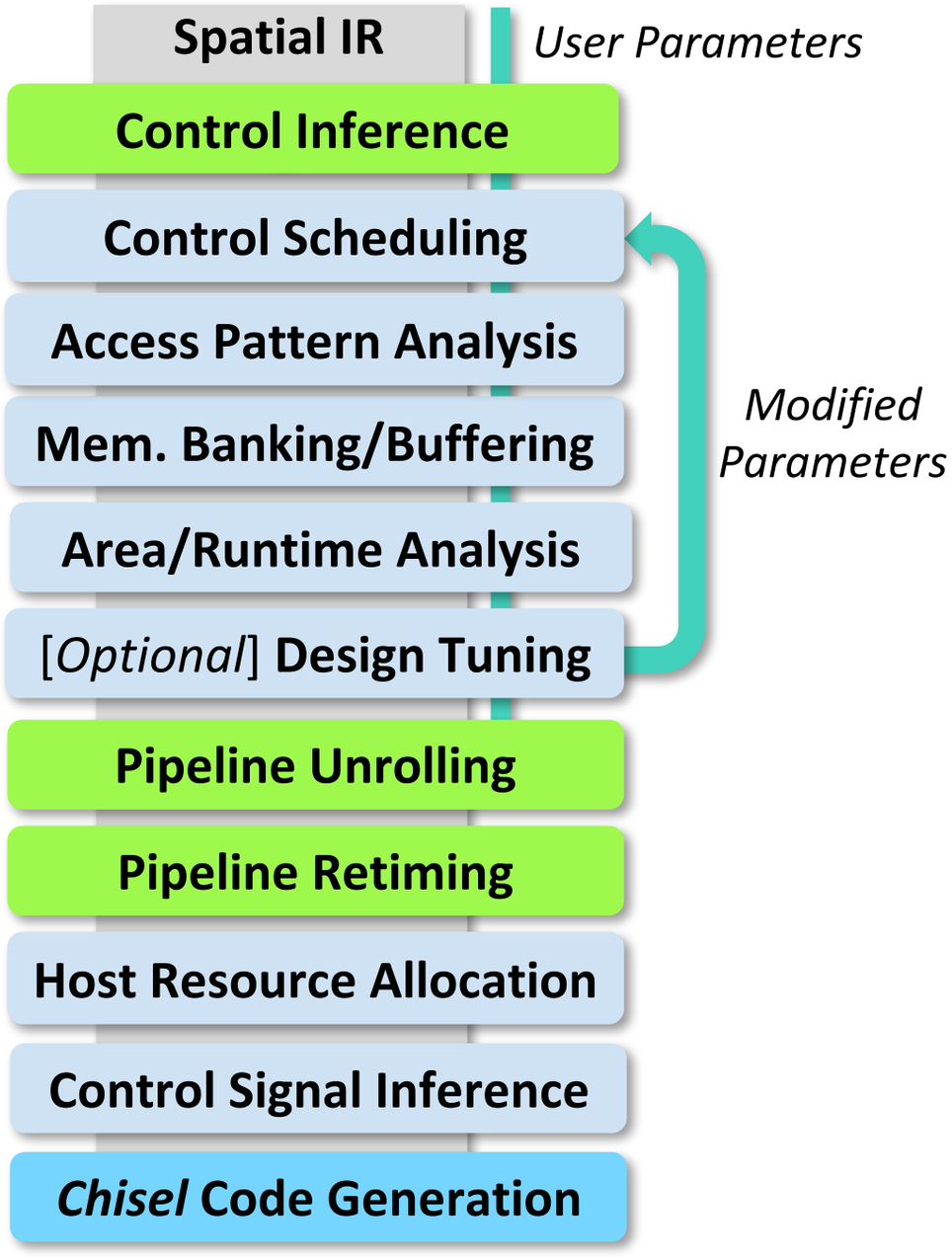


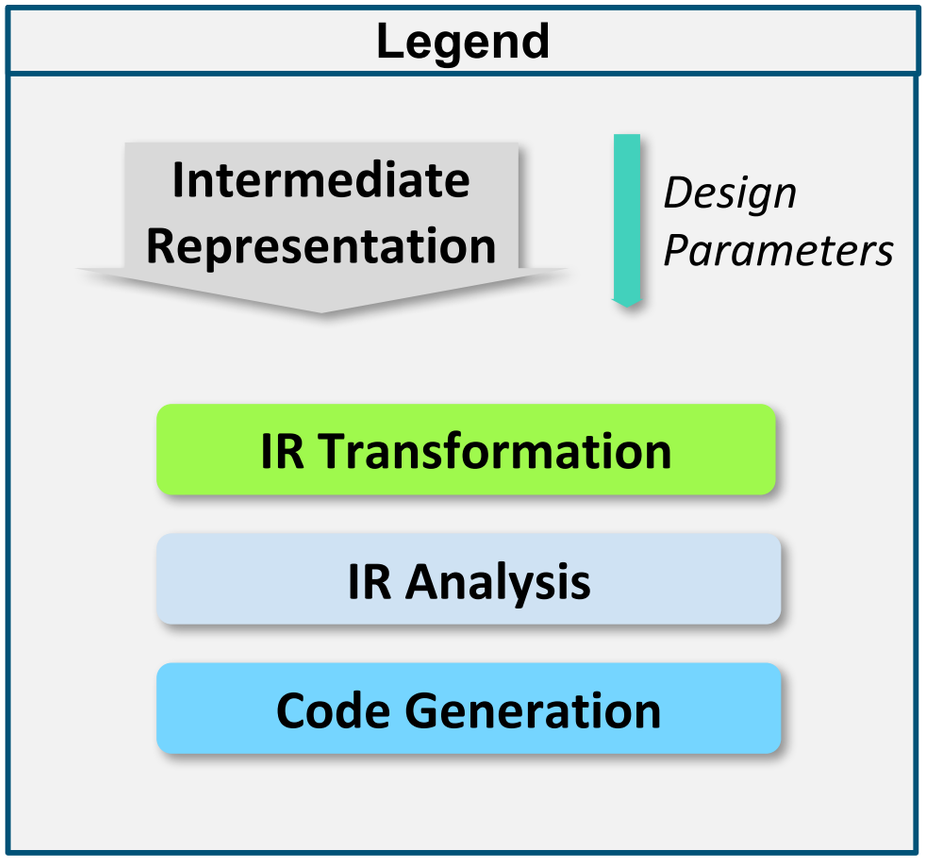
Application 1 - Microsoft Collaboration Preliminary Results on a Simple Scenario





Application 2: FPGA Design Space Exploration (Spatial Compiler)

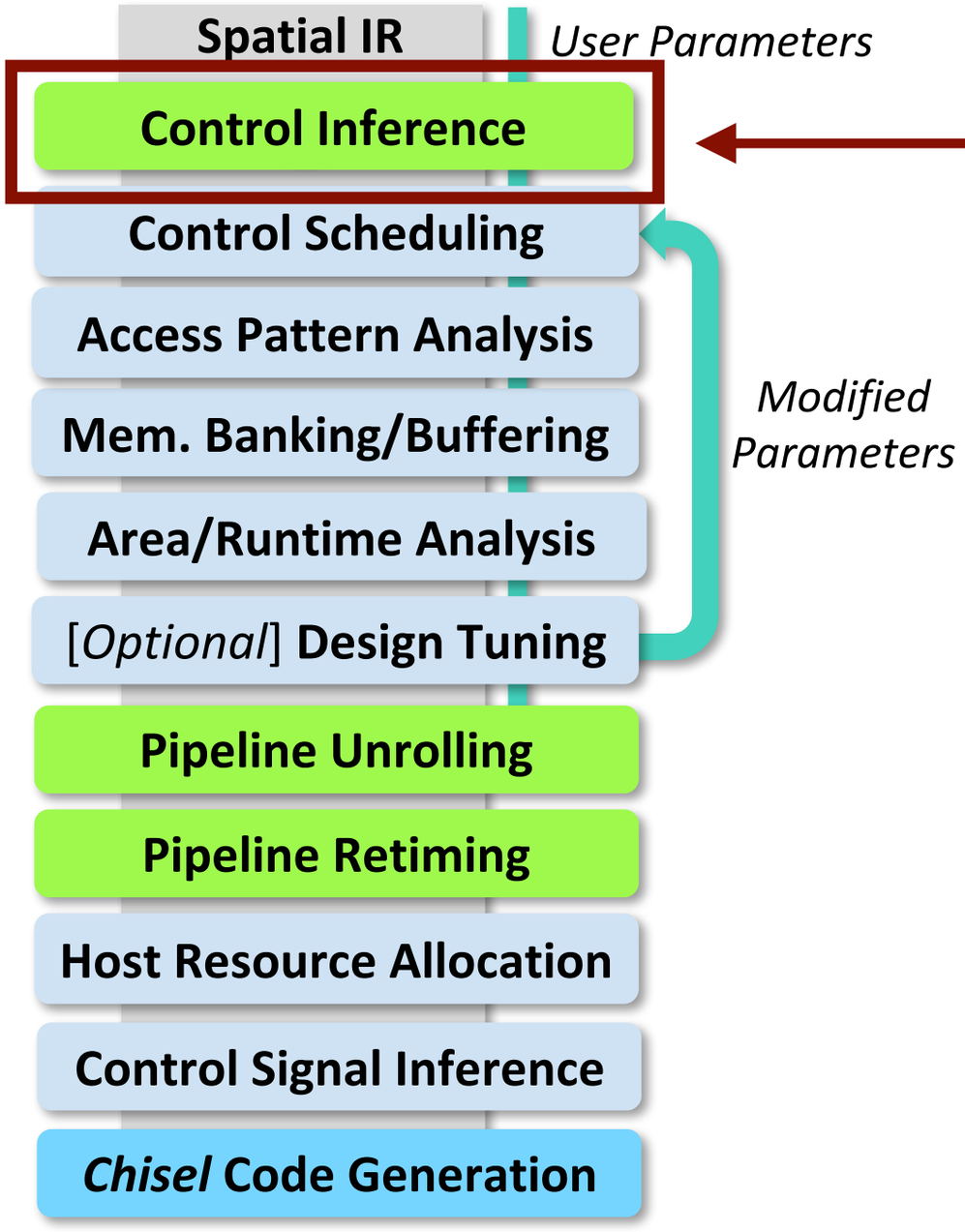




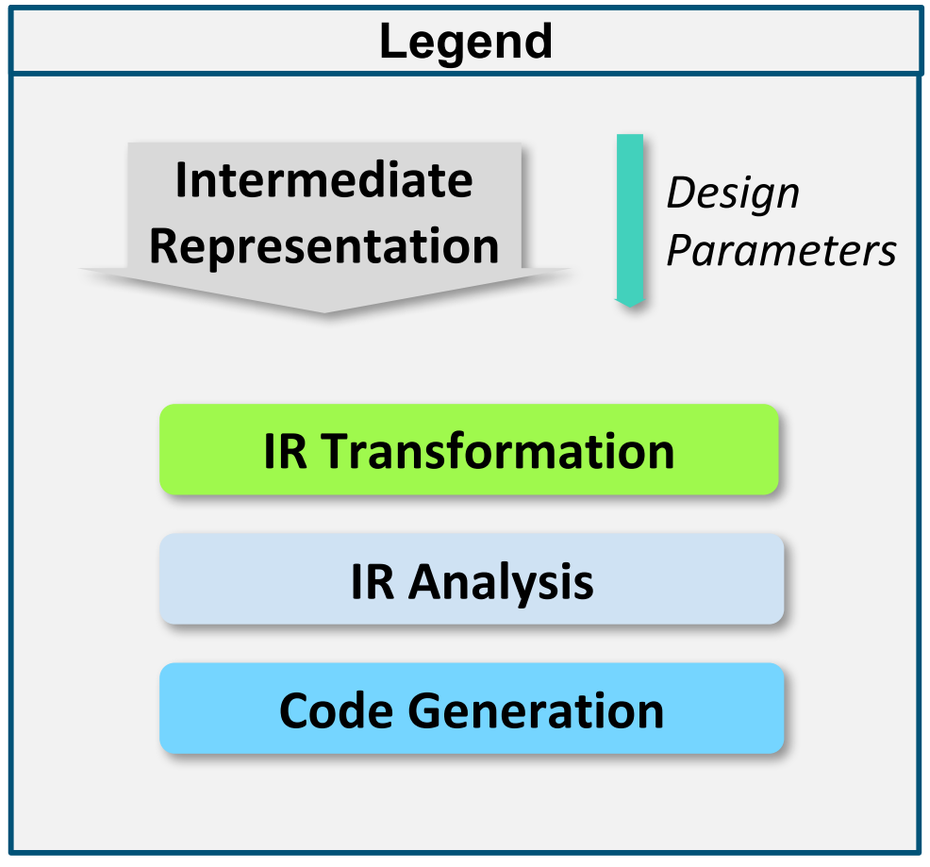
- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Application 2: FPGA Design Space Exploration (Spatial Compiler)



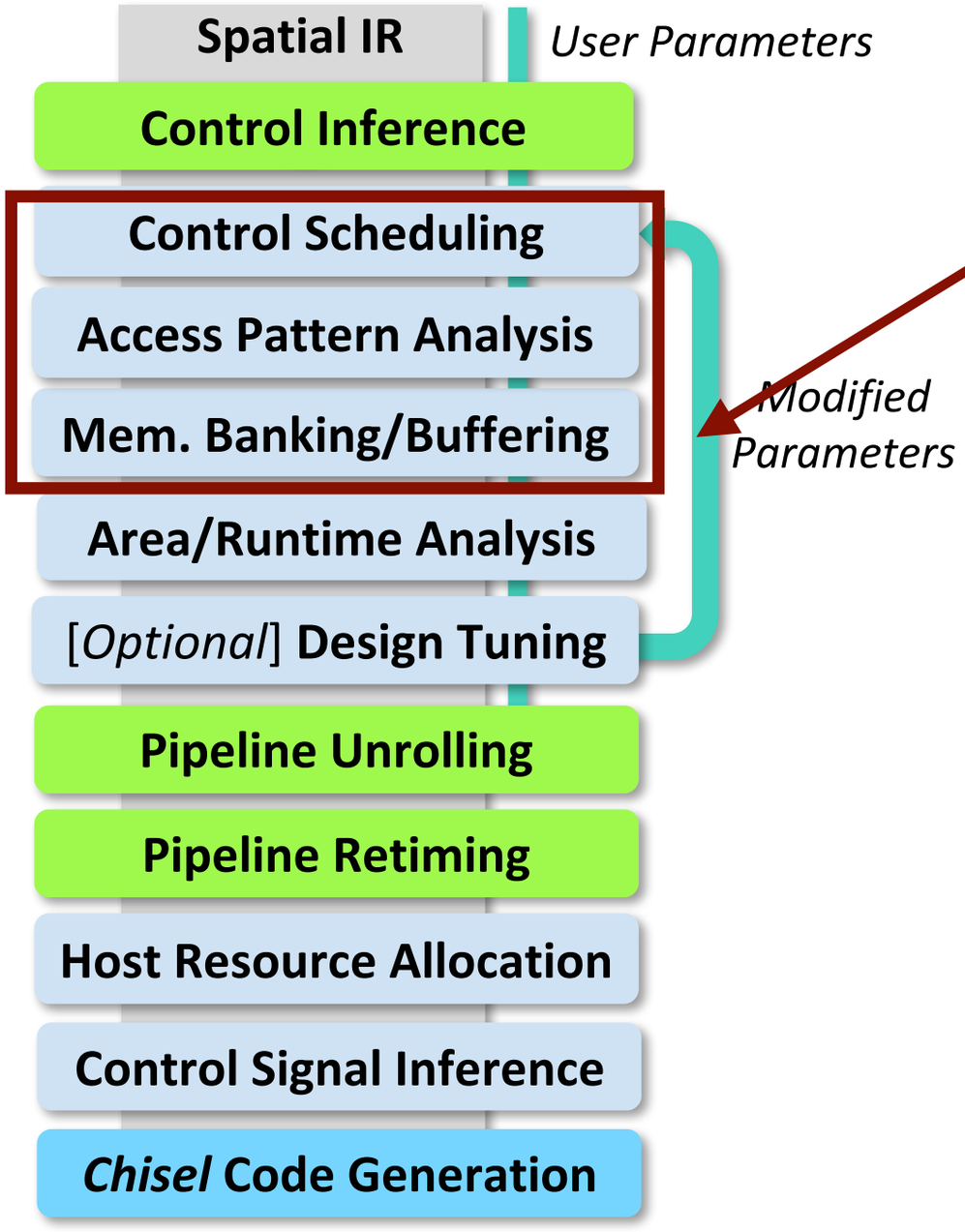
Performs basic hardware optimizations and estimates a domain for each design parameter



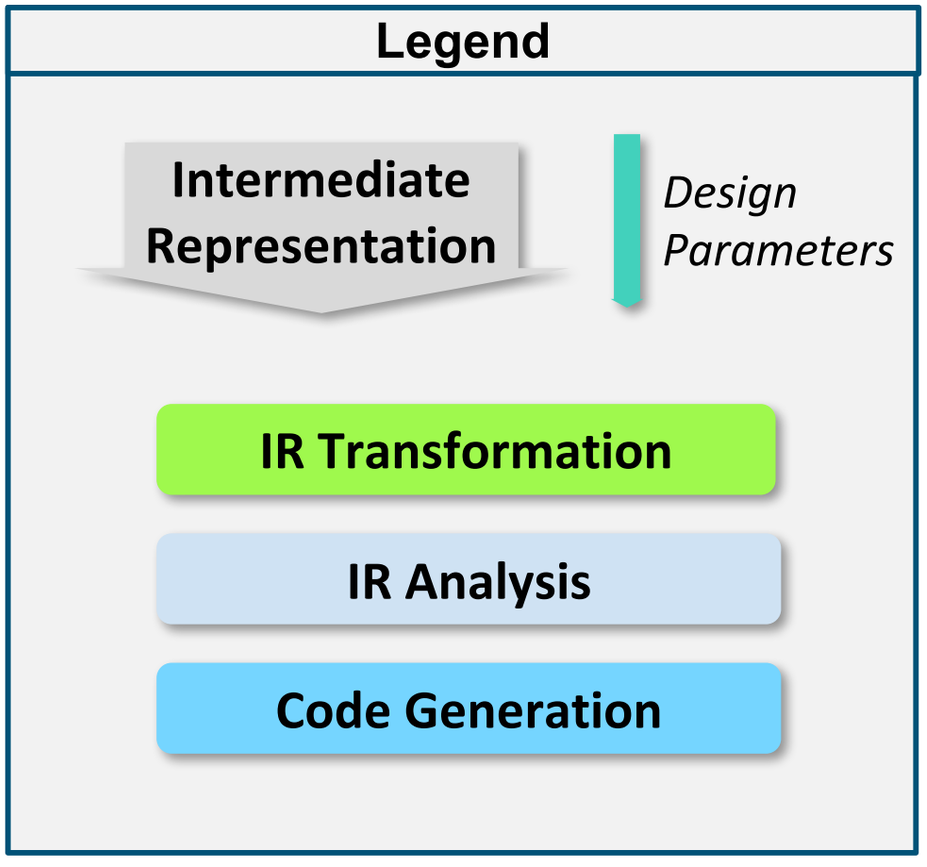
- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Application 2: FPGA Design Space Exploration (Spatial Compiler)



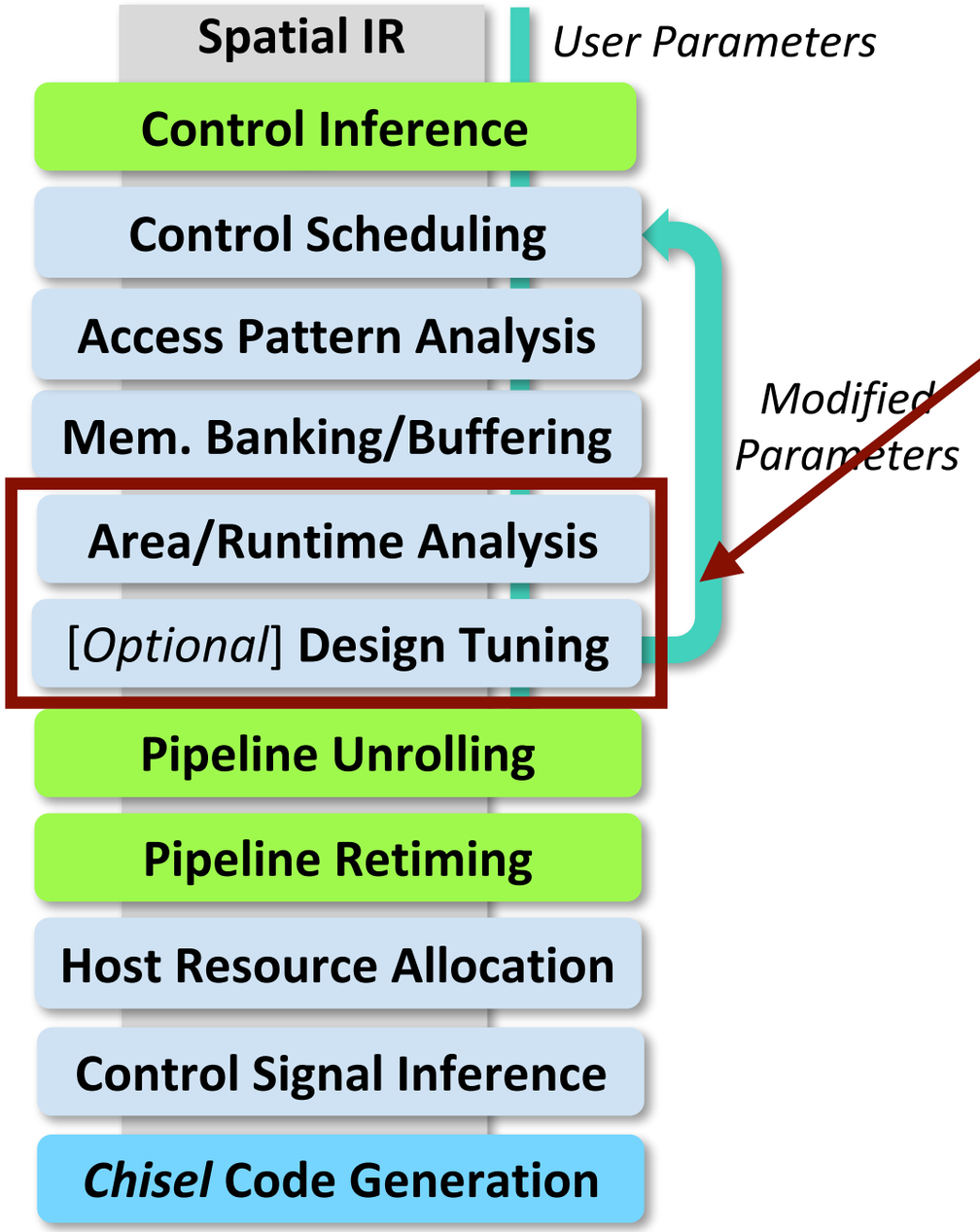
Computes loop pipeline schedules and on-chip memory layouts for some given value for each parameter



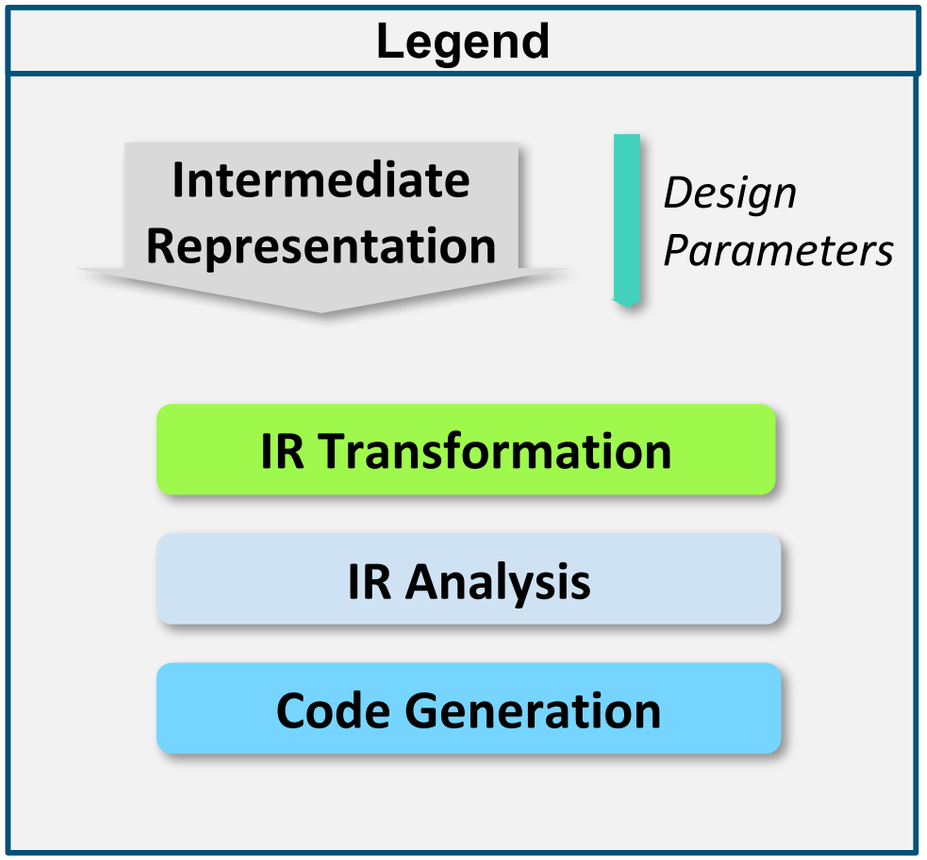
- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Application 2: FPGA Design Space Exploration (Spatial Compiler)



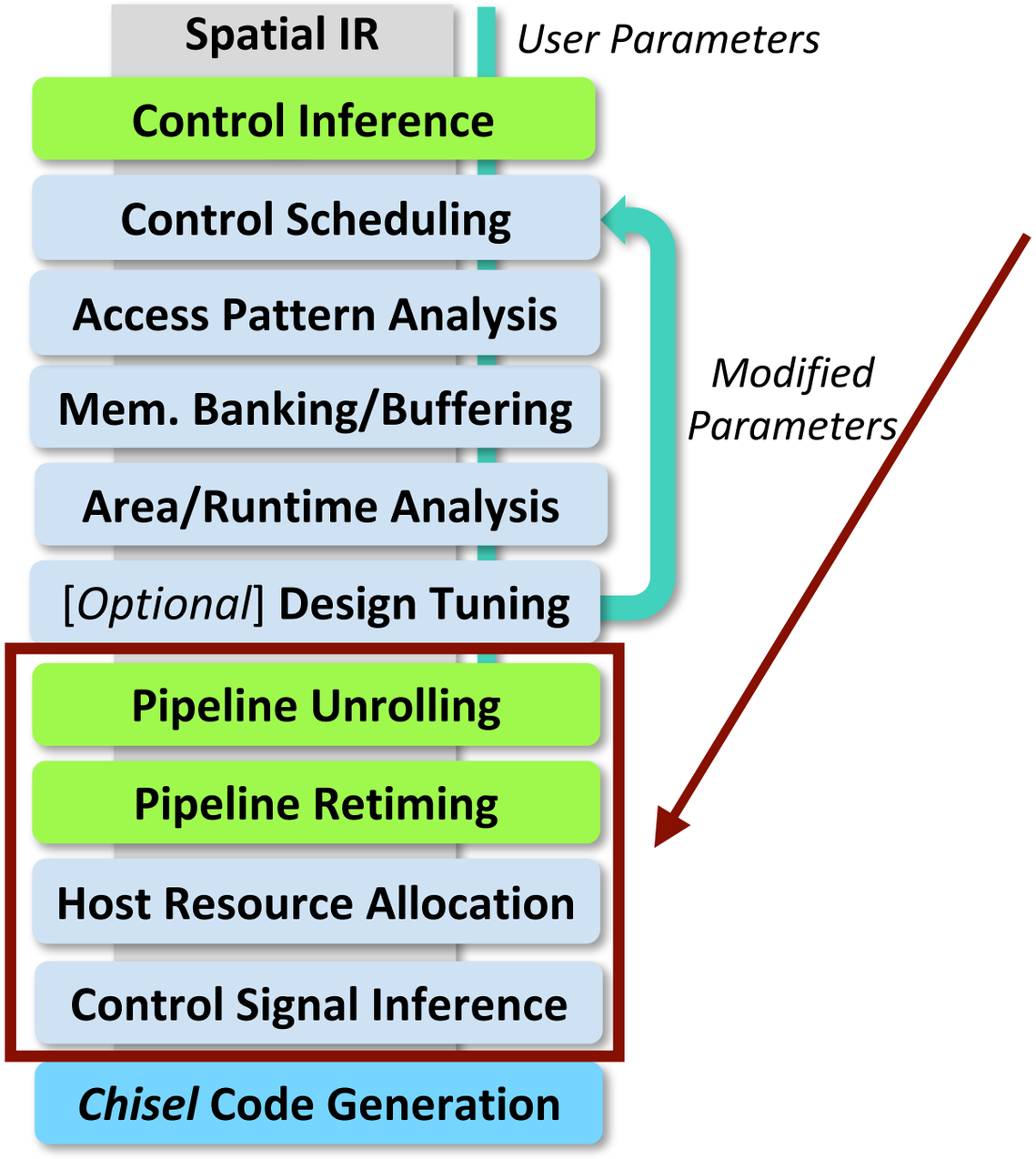
Estimates the amount of hardware resources and the runtime of the application



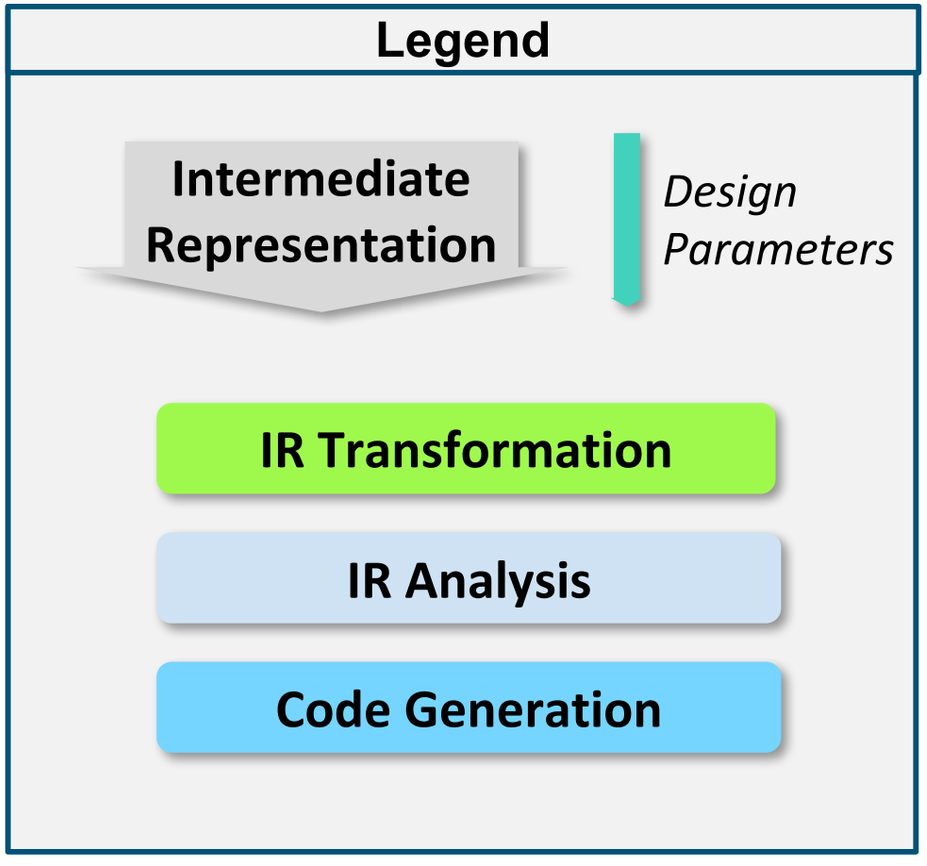
- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Application 2:
FPGA Design Space Exploration (Spatial Compiler)



Unrolls loops, retimes pipelines, and performs on-chip memory layout.
The optimizations are computed based on the analyses of the previous phase

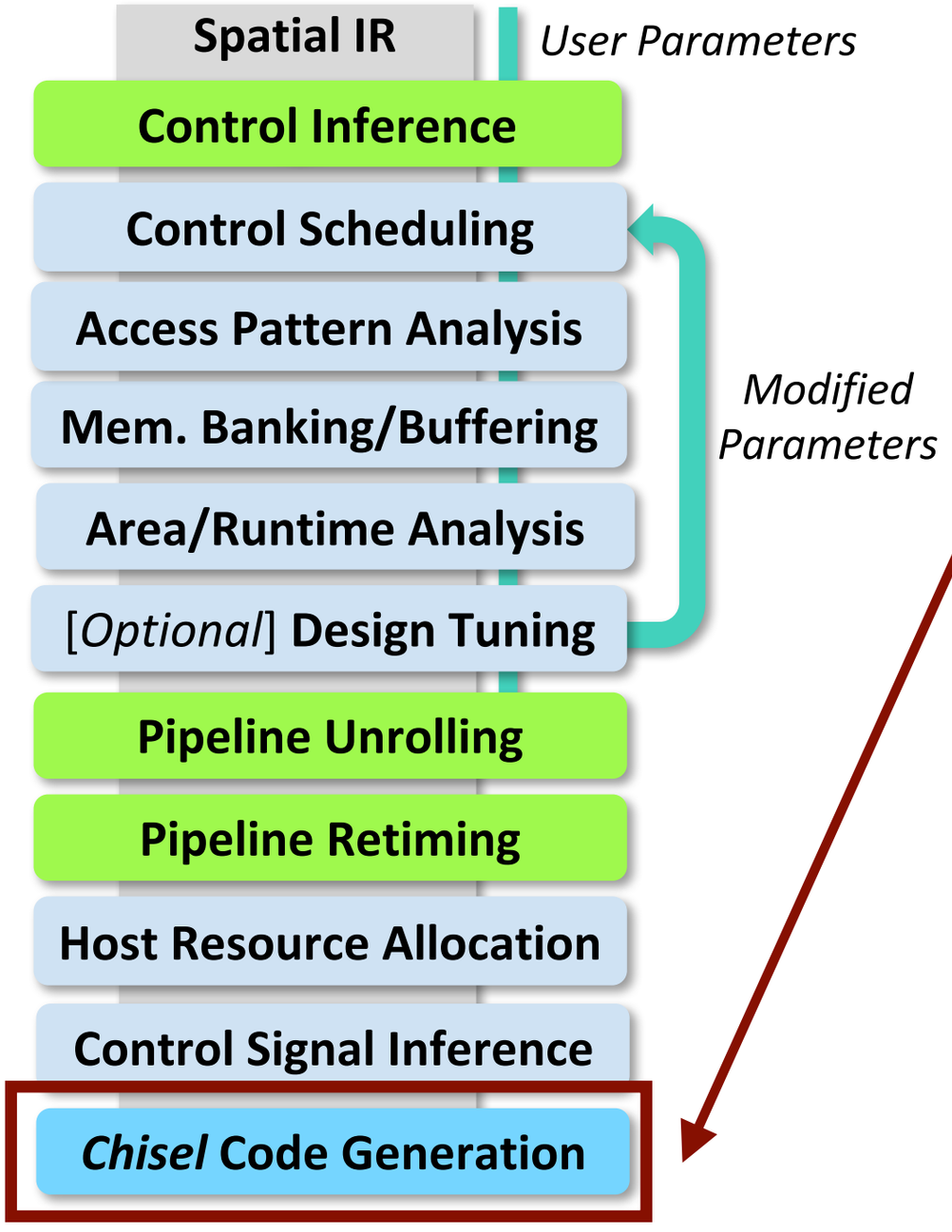


- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

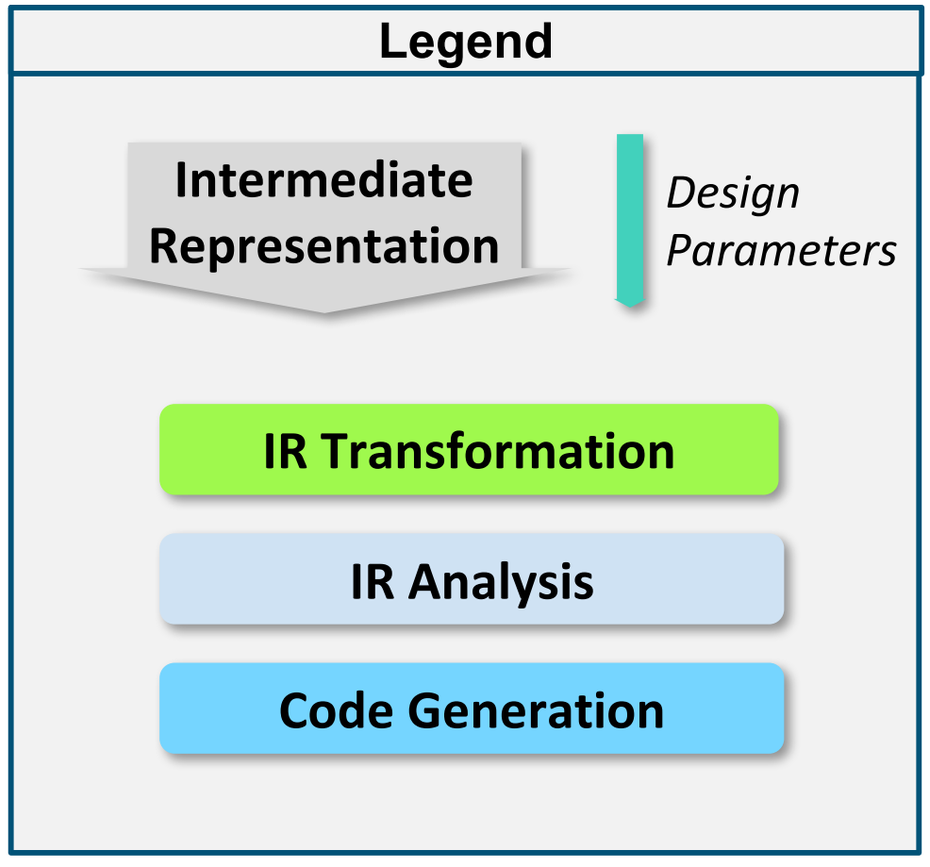
[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Luigi Nardi, Ph.D. - Lund/Stanford

Application 2: FPGA Design Space Exploration (Spatial Compiler)



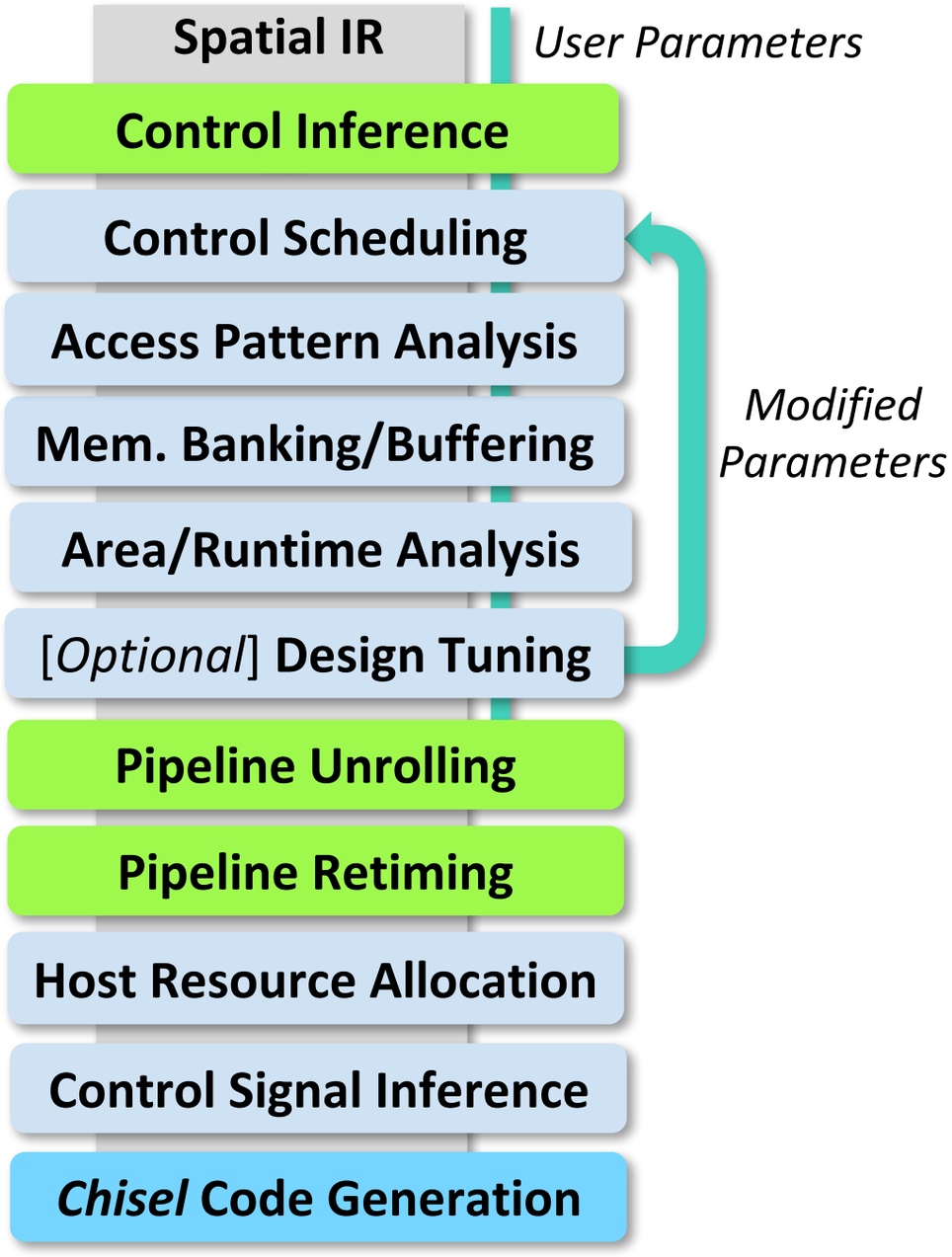
Generates a Chisel design which can be synthesized and run on the target FPGA

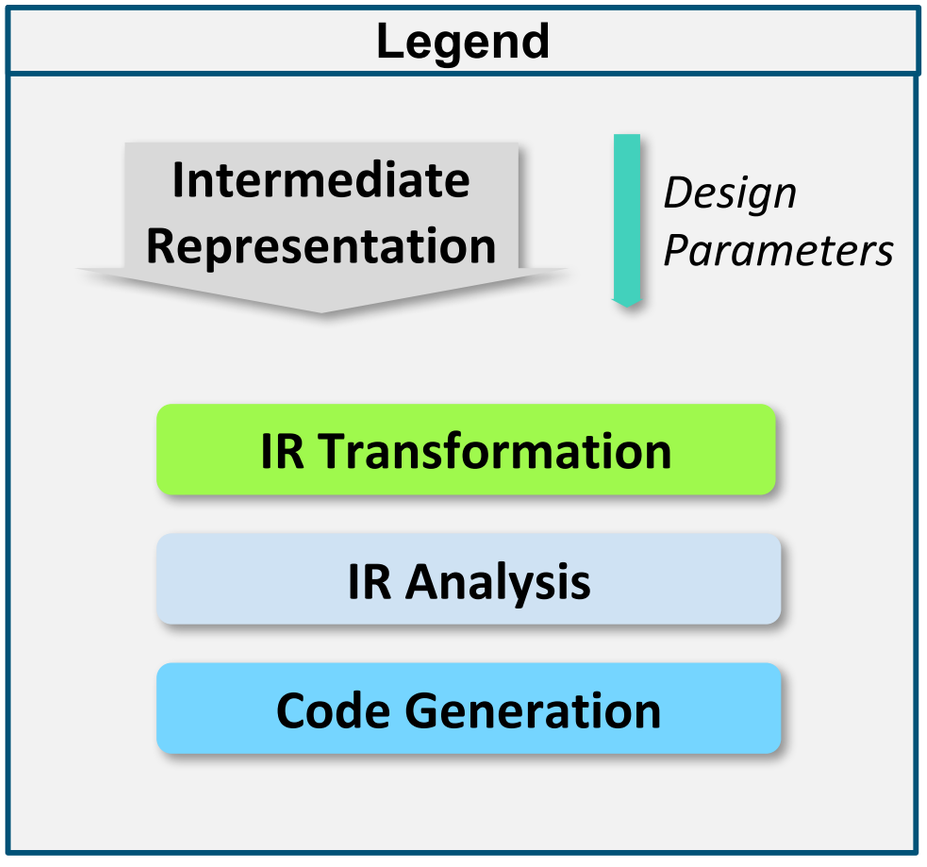


- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Application 2: FPGA Design Space Exploration (Spatial Compiler)



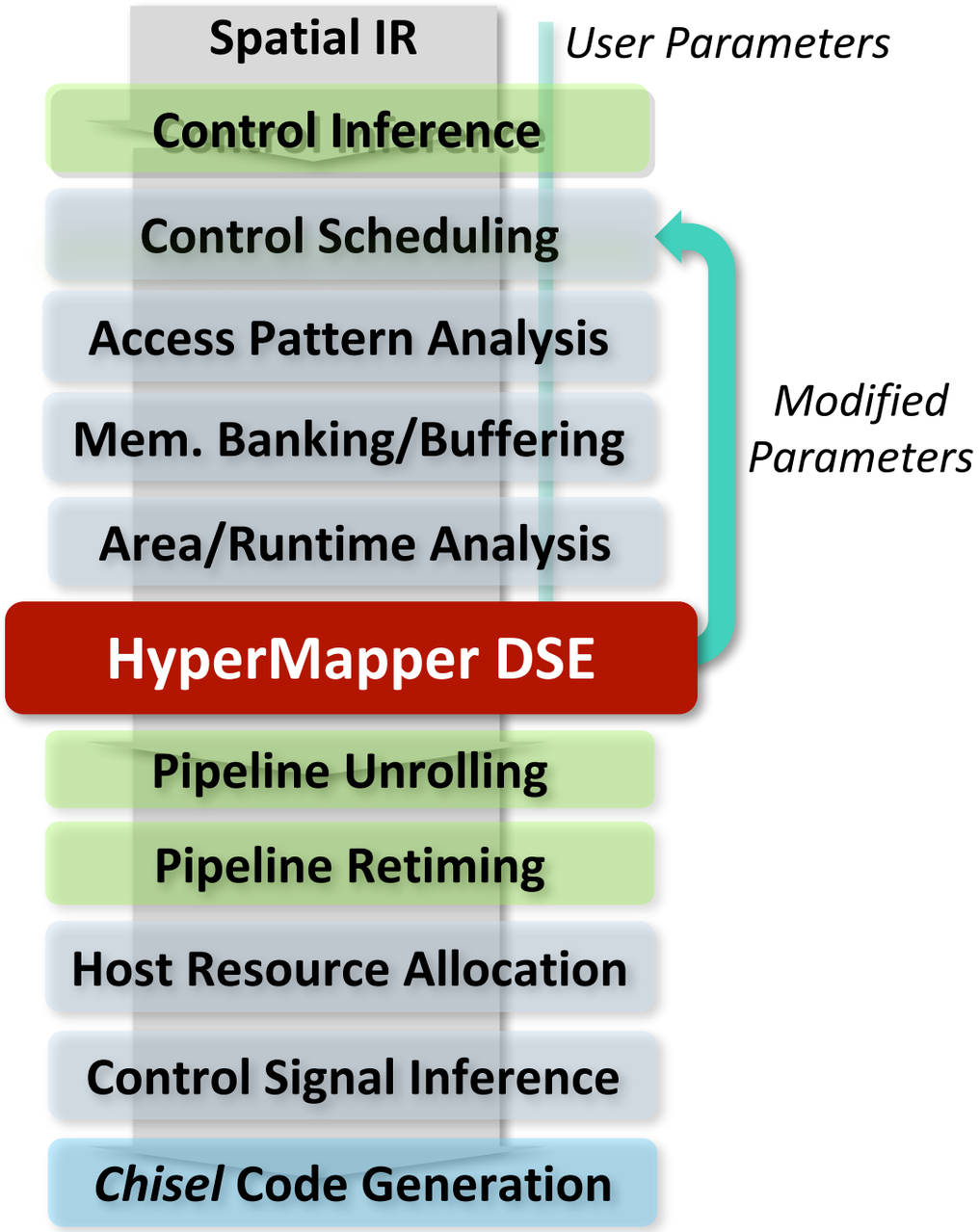


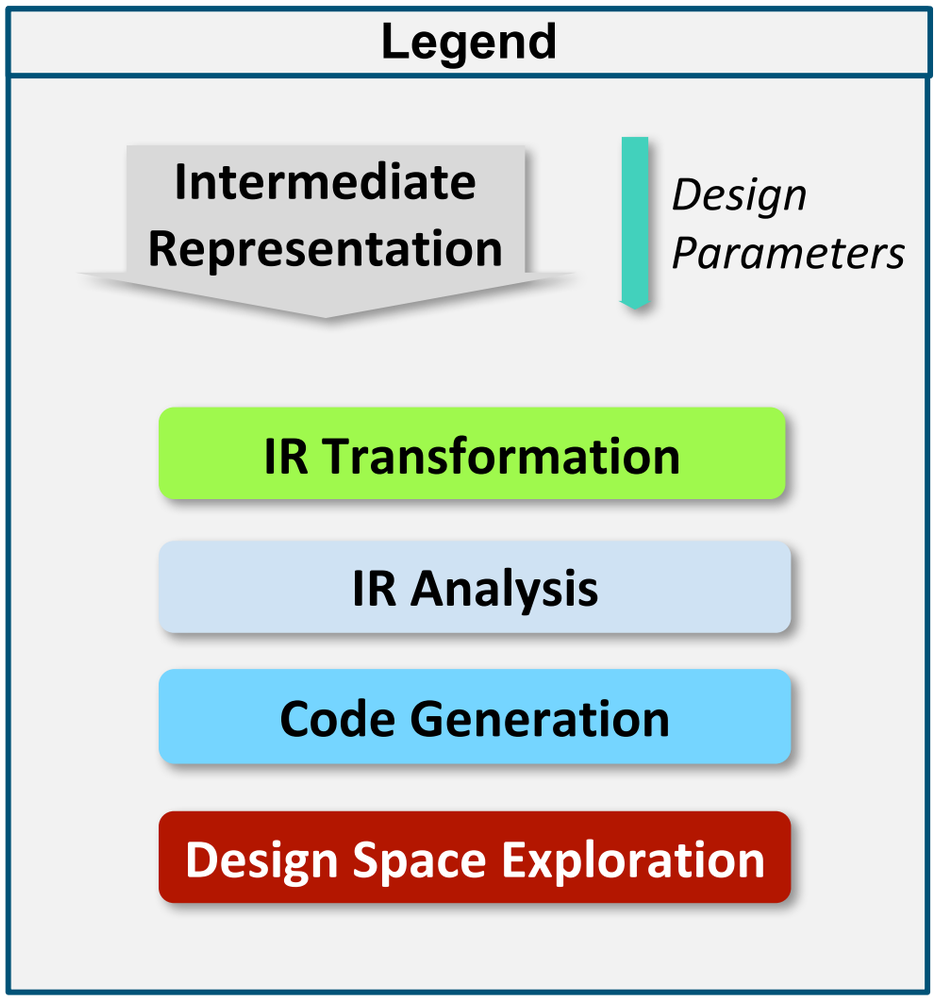
- Goal of Spatial [Koeplinger, et al.]: design of application accelerators
- On reconfigurable architectures FPGAs and CGRAs
- Spatial compiler lowers user programs into synthesizable Chisel [Bachrach, et al.]

[Koeplinger, et al.] "Spatial: a language and compiler for application accelerators." PLDI 2018
[Bachrach, et al.] "Chisel: constructing hardware in a Scala embedded language." DAC 2012.

Luigi Nardi, Ph.D. - Lund/Stanford

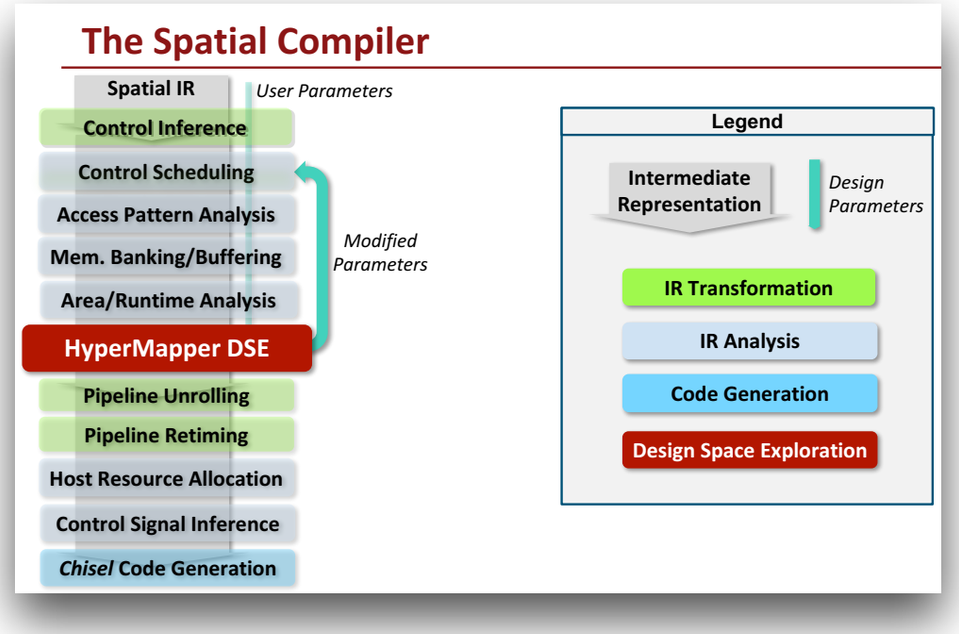
Application 2:
FPGA Design Space Exploration (Spatial Compiler)





Luigi Nardi, Ph.D. - Lund/Stanford

Application 2:
FPGA Design Space Exploration (Spatial Compiler)



Benchmark	Variables	Space Size
BlackScholes	4	7.68×10^4
K-Means	6	1.04×10^6
OuterProduct	5	1.66×10^7
DotProduct	5	1.18×10^8
GEMM	7	2.62×10^8
TPC-H Q6	5	3.54×10^9
GDA	9	2.40×10^{11}

Input

Compiler automatically provides params:

- Tile size (ordinal)
- Inner and outer loop pipelining (ordinal)
- Meta-pipe (categorical)
- Unrolling factor (ordinal)
- Memory banking (ordinal)
- Parallelism (categorical)

Output

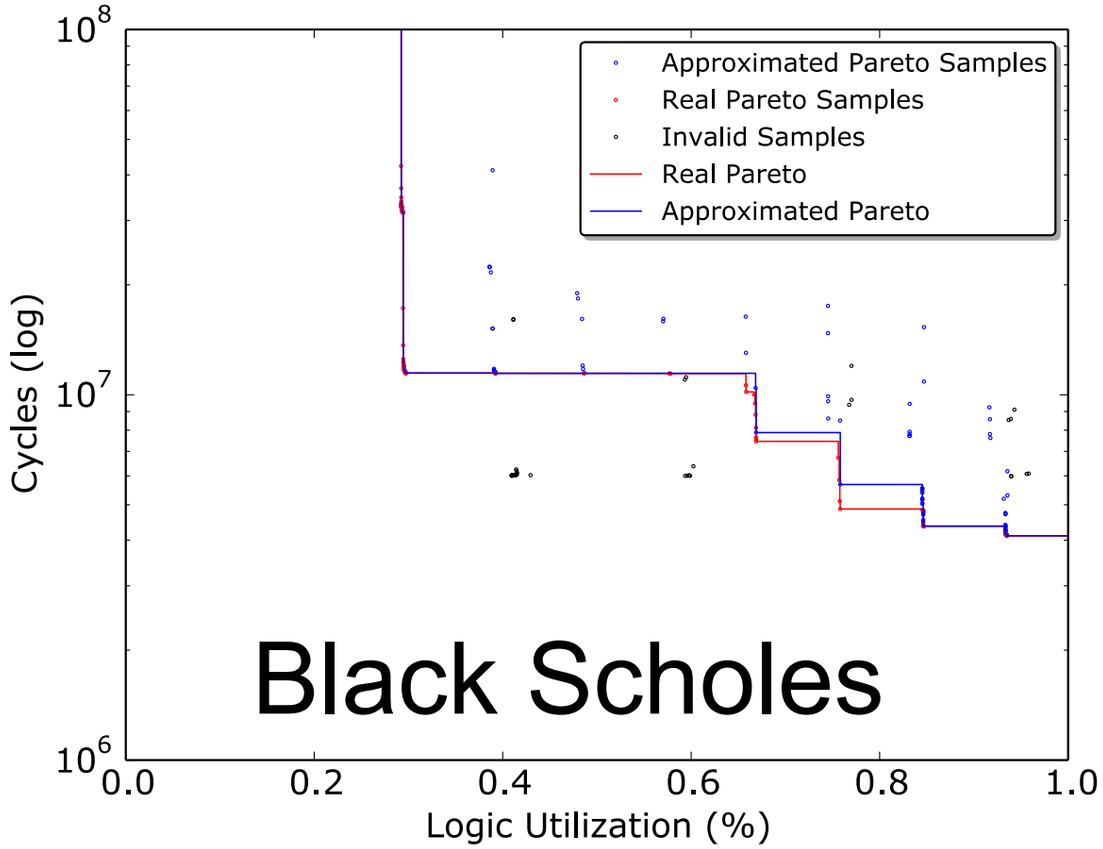
Two objectives:

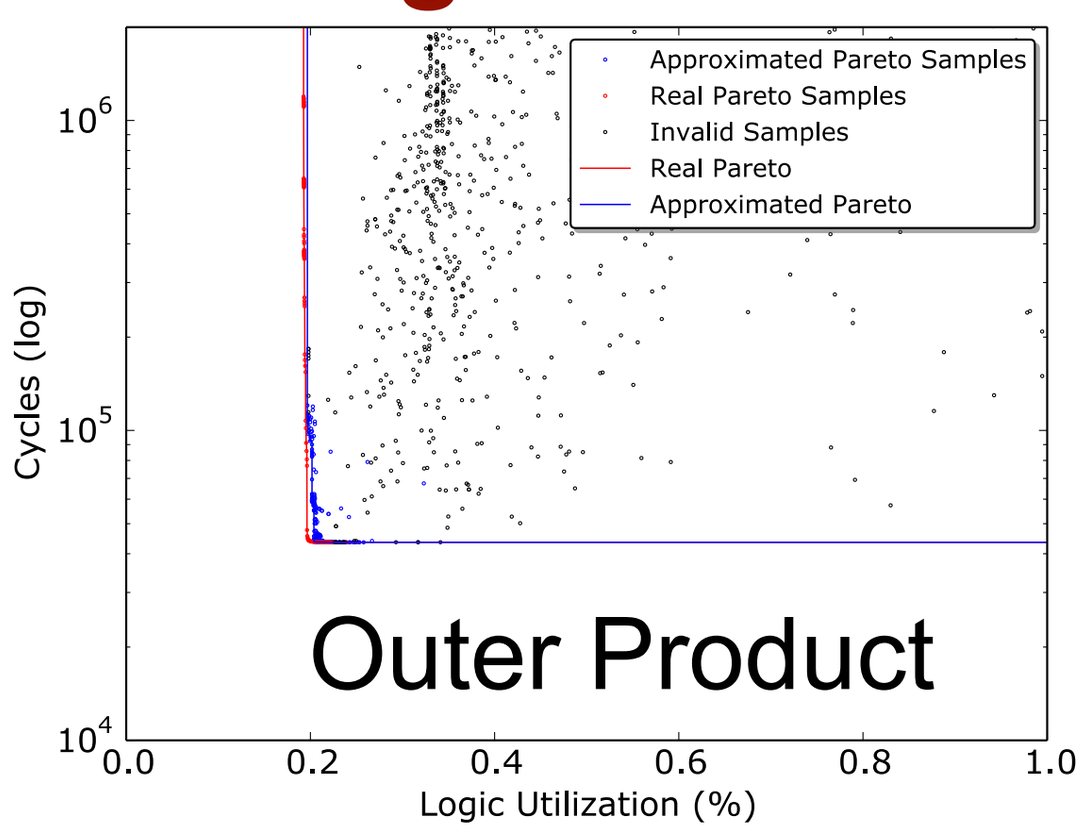
1. Minimize clock cycles (runtime)
2. Minimize FPGA logic utilization
   - For fitting multiple applications
   - Proxy for energy consumption

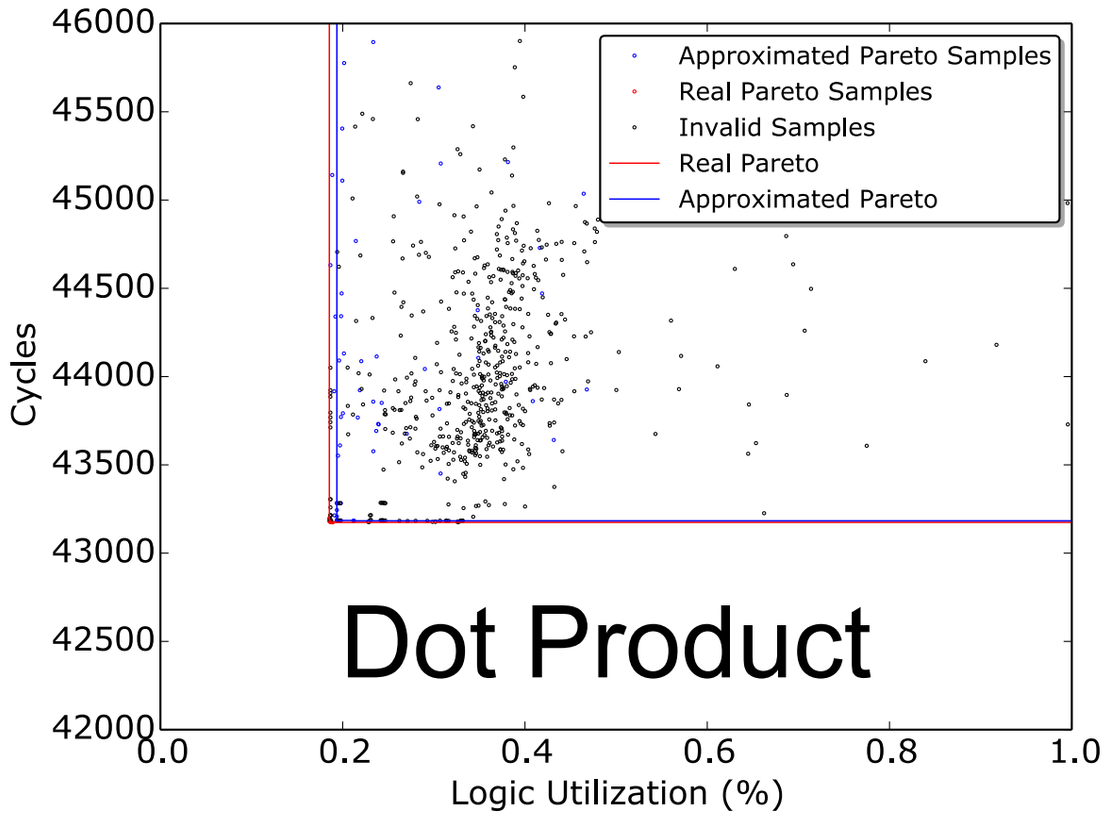
One constraint:

design must fit in the chip

Small benchmarks - Cycles/Logic Utilization



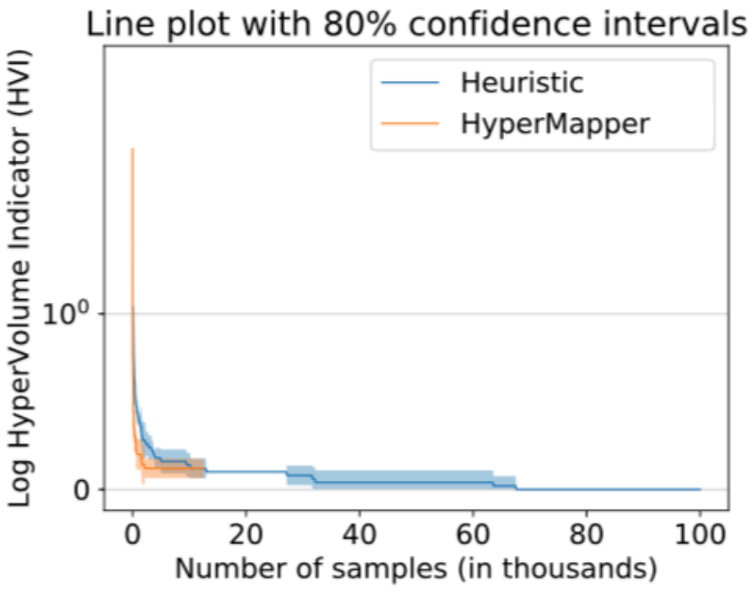




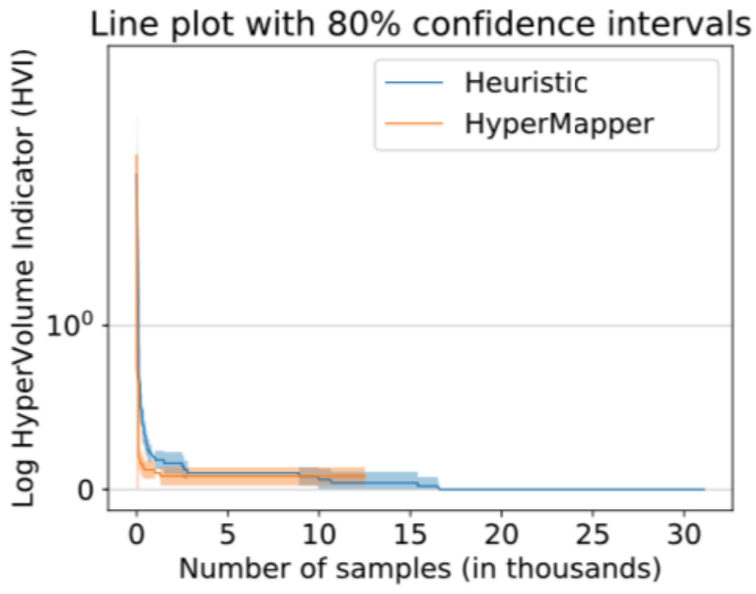
- 3 small benchmarks: compute real Pareto
- Real Pareto is exhaustive search
- Exhaustive: 6 to 12 hours on 16 cores
- Approximated Pareto is HyperMapper

HVI Cycles/Logic Utilization

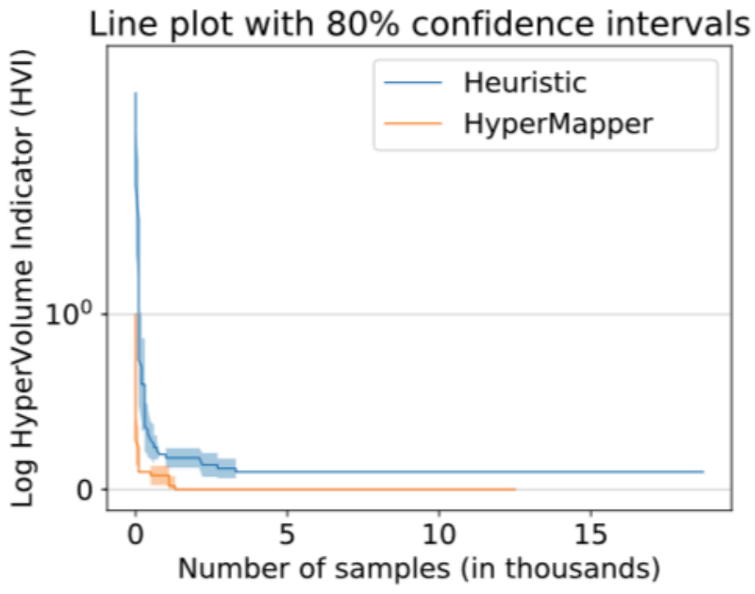
- Hypervolume indicator (HVI): scalar to compare Paretos
- Baseline is a pruning (based on heuristics) and 100K random sampling



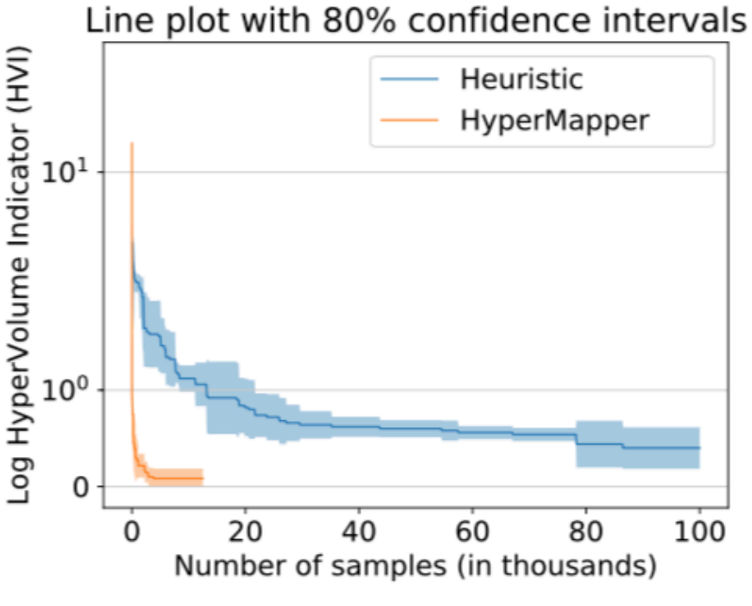
GEMM



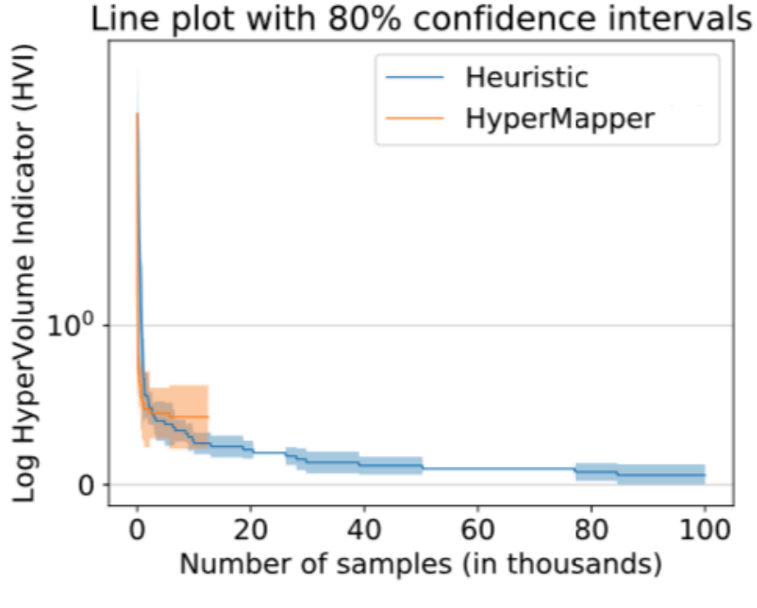
OuterProduct



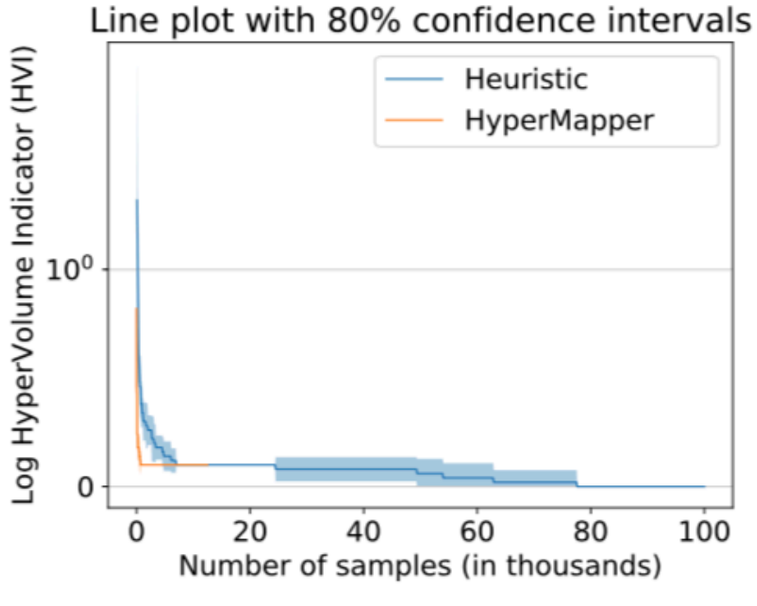
K-Means



GDA



T-PCH Q6



DotProduct

Luigi Nardi, Ph.D. - Lund/Stanford

Application 3:
Simultaneous Localization And Mapping Optimization

Build a coherent world representation and localize the camera in real-time

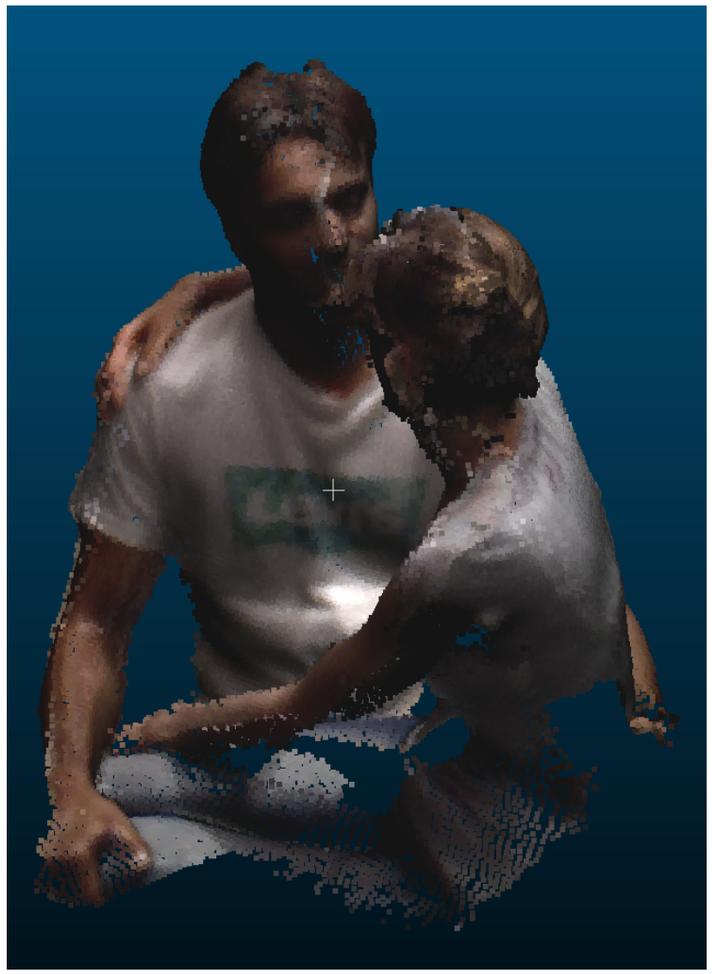
I will focus on two SLAM algorithms from SLAMBench:

- KinectFusion [Newcombe et al. ISMAR 2011]
- ElasticFusion [Whelan et al. RSS 2015]

Jesse Clayton (NVIDIA)
3D reconstruction

Applications, e.g.:

- Robotics
- Autonomous driving
- 3D printing
- Augmented reality
- Telepresence



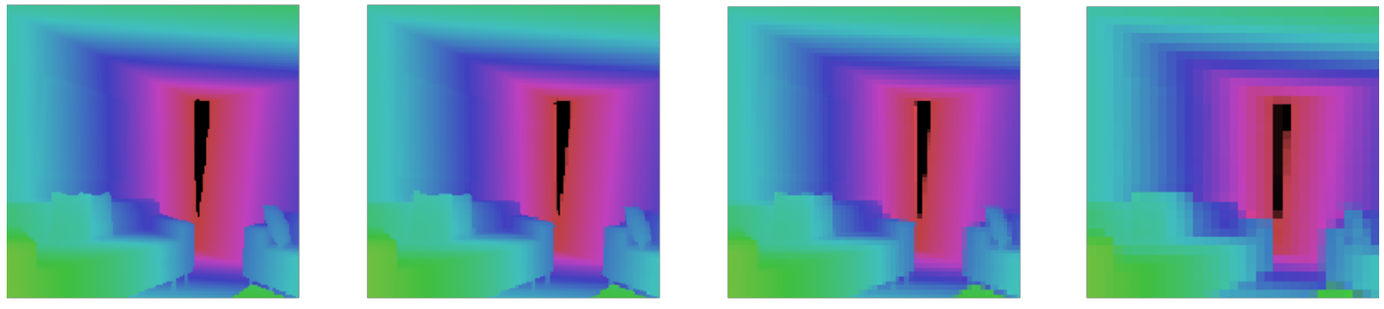
Daniele and Daniela
3D reconstruction

Pictures featured using ElasticFusion

Application 3: SLAM (KinectFusion) algorithmic features

Features	Ranges
Volume resolution	64x64x64, 128x128x128, 256x256x256, 512x512x512
μ distance	0 .. 0.5
Pyramid level iterations (3 levels)	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
Image resolution (image ratio)	1, 2, 4, 8
Tracking rate	1, 2, 3, 4, 5
ICP threshold	10^{-6} .. 10^{2}
Integration rate	1 .. 30

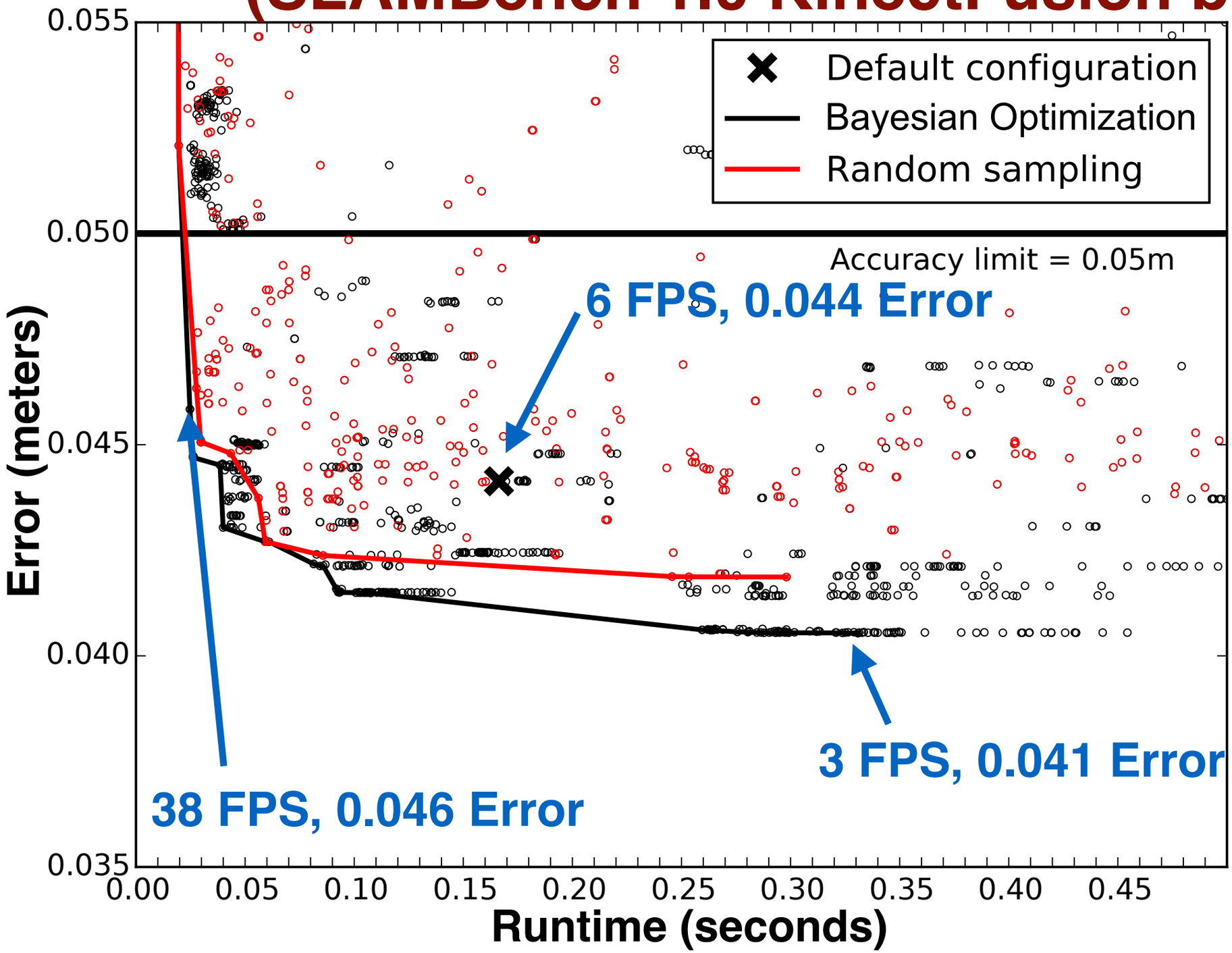
Image resolution (image ratio)

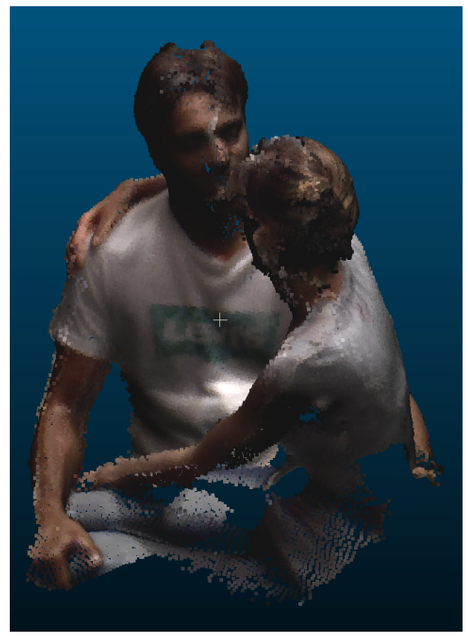


<u>Three objectives</u>:

1. Error
2. Runtime
3. Energy consumption

Example 3: Computer Vision Use Case
(SLAMBench 1.0 KinectFusion benchmark)

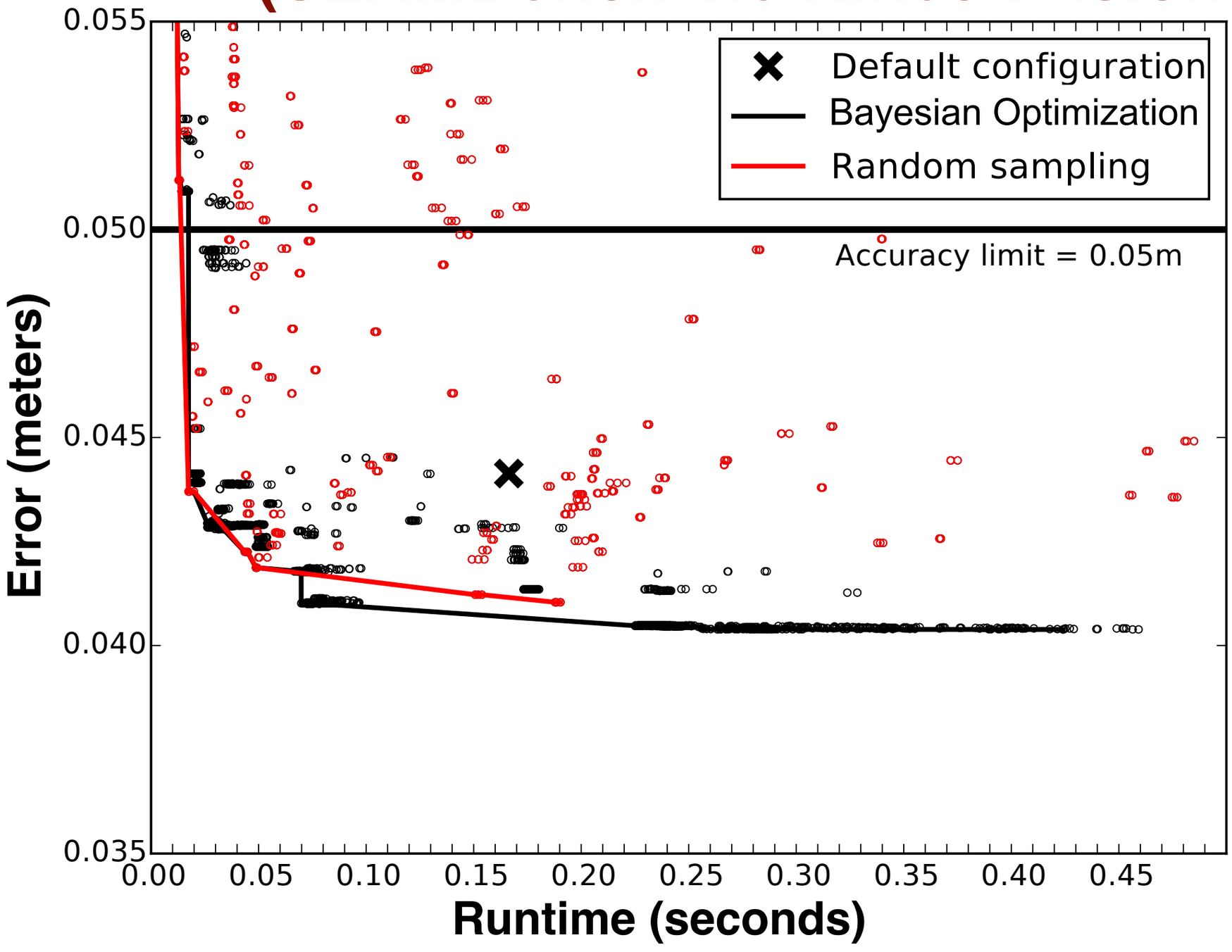




3D reconstruction

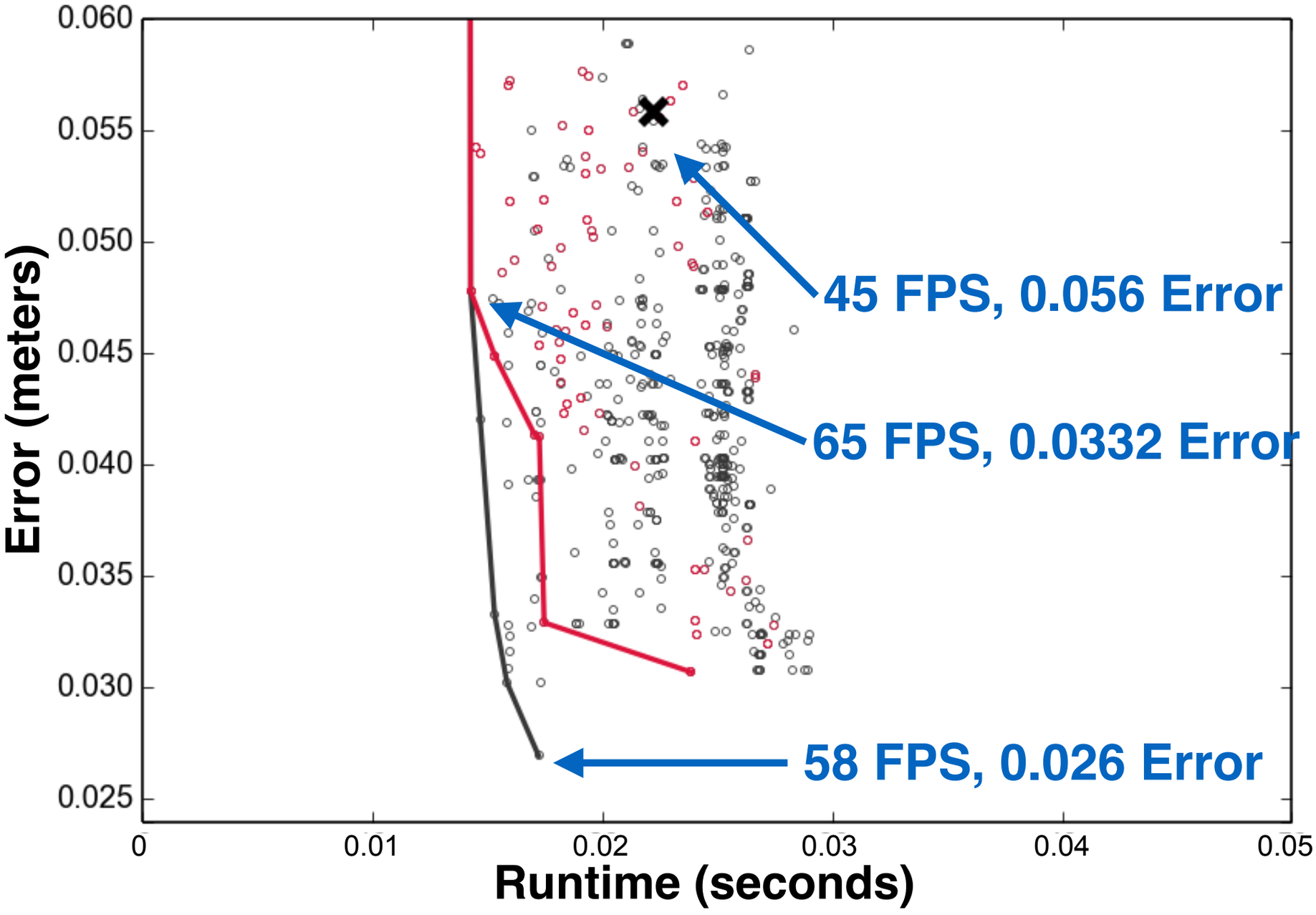
Machine	Type	CPU	CPU name	CPU cores	GPU	GPU name
Hardkernel **ODROID-XU3**	Embedded	ARM A15 + A7	Exynos 5422	4 + 4	ARM	Mali-T628

Example 3: Computer Vision Use Case
(SLAMBench 1.0 KinectFusion benchmark)



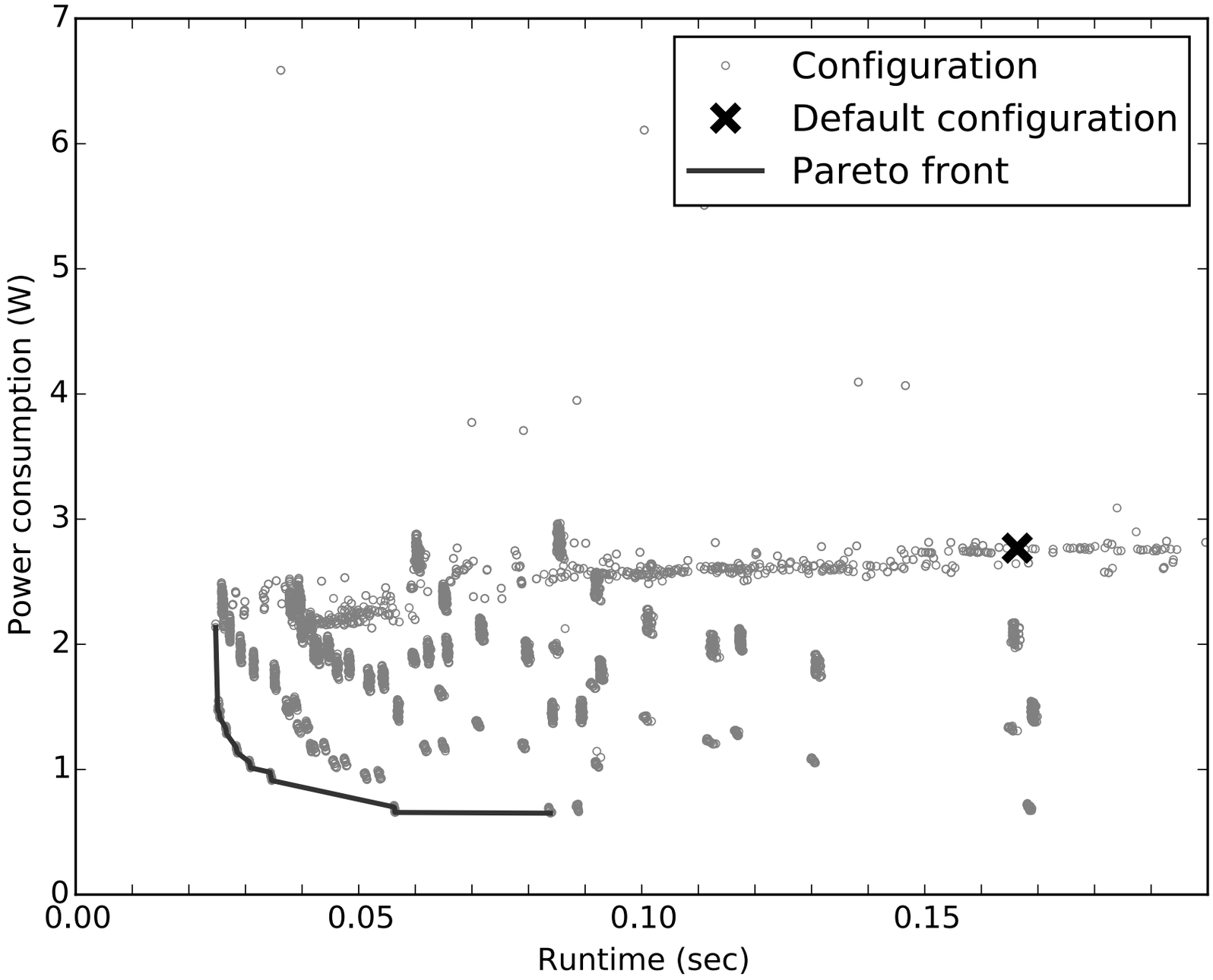
Machine	Type	CPU	CPU name	CPU cores	GPU	GPU name
ASUS **T200TA**	Detachable laptop	Intel Silvermont	Atom Z3795	4	Intel	HD Graphics

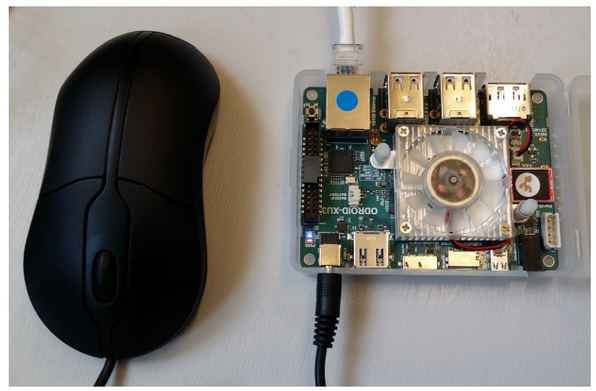
Example 3: Computer Vision Use Case
(SLAMBench 2.0 ElasticFusion benchmark)



Machine	Type	CPU	CPU name	CPU cores	GPU	GPU name
NVIDIA/Intel	Desktop	Intel Ivy Bridge	E5-1620	8	NVIDIA	GTX 780 Ti

Example 3: Computer Vision Use Case
(SLAMBench 1.0 KinectFusion benchmark)





Machine	Type	CPU	CPU name	CPU cores	GPU	GPU name
Hardkernel **ODROID-XU3**	Embedded	ARM A15 + A7	Exynos 5422	4 + 4	ARM	Mali-T628

Conclusion and Future Work

- Goal: HyperMapper is a multi-objective BBO framework
- Demonstrated on:
  hardware design, DBMS and CV applications
- Future:
  - Dynamic Black-box optimization
  - Use of prior knowledge in the multi-objective posterior
  - Asynchronous batch optimization
  - Transfer learning
  - Applications:
    - Microsoft PostgreSQL/Azure full scenario
    - Multi-fidelity in SLAM
    - Halide computer vision language

Research Team and Collaborators

- Work in collaboration with many people from academia and industry
- **We are hiring!** WASP-AI is funding positions for my research group:
  - 2 Ph.D. students (https://lu.varbi.com/en/what:job/jobID:280143/)
  - 2 Postdocs (https://lu.varbi.com/en/what:job/jobID:280619/)

Luigi Nardi, Ph.D.
WASP-AI Assistant Professor
in Machine Learning

Artur Luis
Ph.D. Student
UFMG

Info on HyperMapper

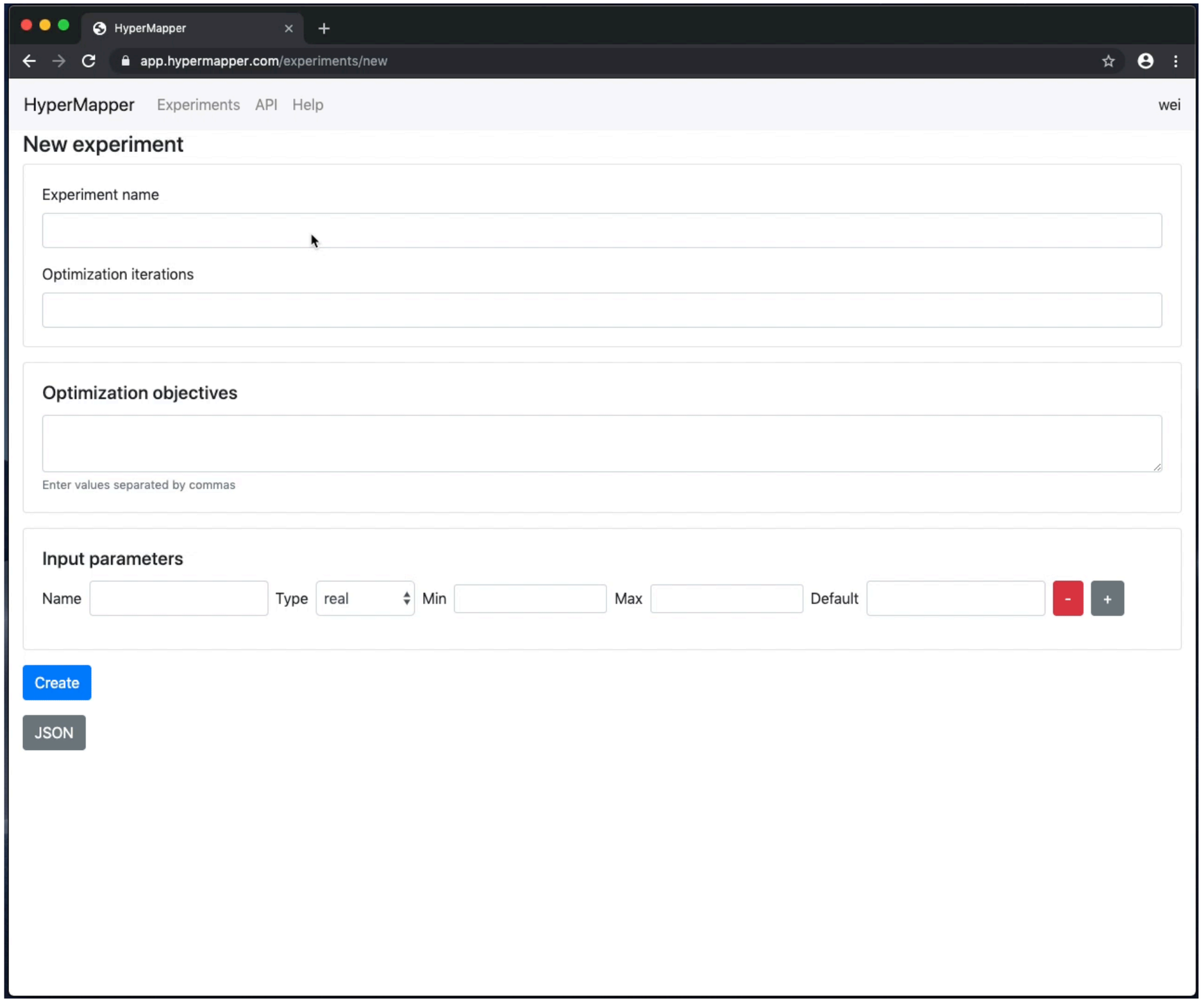
- Join HyperMapper on **Slack**: hypermapper.slack.com
- **Repo**: https://github.com/luinardi/hypermapper
- **Wiki**: https://github.com/luinardi/hypermapper/wiki
- HyperMapper **SaaS**: www.hypermapper.com

Adopters

Microsoft	Microsoft - Database Management Systems
teradata.	Teradata - Database Management Systems
Stanford University	Stanford University - Hardware Design
UC San Diego	UCSD - Spector Benchmarks (FPGAs)
TEXAS The University of Texas at Austin	UT Austin - Capri (Approximate Computing)
Imperial College London	ICL - Computer Vision and Robotics
	Etc.

Backup slides

HyperMapper SaaS



Demo HyperMapper

https://github.com/luinardi/hypermapper/wiki/Quick-Start-Guide