Xopt: A Simplified Framework for Optimization of Arbitrary Problems using Advanced Algorithms

SLAC NATIONAL ACCELERATOR LABORATORY

R. Roussel, C. Mayes[1]

[1] SLAC National Accelerator Laboratory, Stanford University, Menlo Park CA 94025, USA

Introduction

- Recent development of **advanced black box algorithms** has promised order of magnitude improvements in optimization speed when solving physics problems.
- Algorithms remain inaccessible to the general accelerator community, due to the expertise and infrastructure **required** to apply them towards solving optimization problems.
- We introduce the Python package, **Xopt** (`github.com/ChristopherMayes/Xopt`), which implements a simple interface for connecting arbitrarily specified optimization problems with advanced algorithms.

Xopt Input Options

- Xopt requires a **simple Python function** to evaluate the value of objectives, constraints, etc. as a function of input variables

```
evaluate(input[dict]) -> output[dict]
```

- Xopt can be **completely initialized from a YAML file**, including evaluation function, optimization algorithms, etc. (useful for limiting code, running cluster jobs)
- Inputs are validated at runtime using Pydantic

```
xopt:
  max_evaluations: 6400

generator:
  name: cnsga
  population_size: 64
  population_file: test.csv
  output_path: .

evaluator:
  function: xopt.resources.
    test_functions.tnk.evaluate_TNK
  function_kwargs:
    raise_probability: 0.1

vocs:
  variables:
    x1: [0, 3.14159]
    x2: [0, 3.14159]
  objectives: {y1: MINIMIZE, y2:
    MINIMIZE}
  constraints:
    c1: [GREATER_THAN, 0]
    c2: [LESS_THAN, 0.5]
  linked_variables: {x9: x1}
  constants: {a: dummy_constant}
```

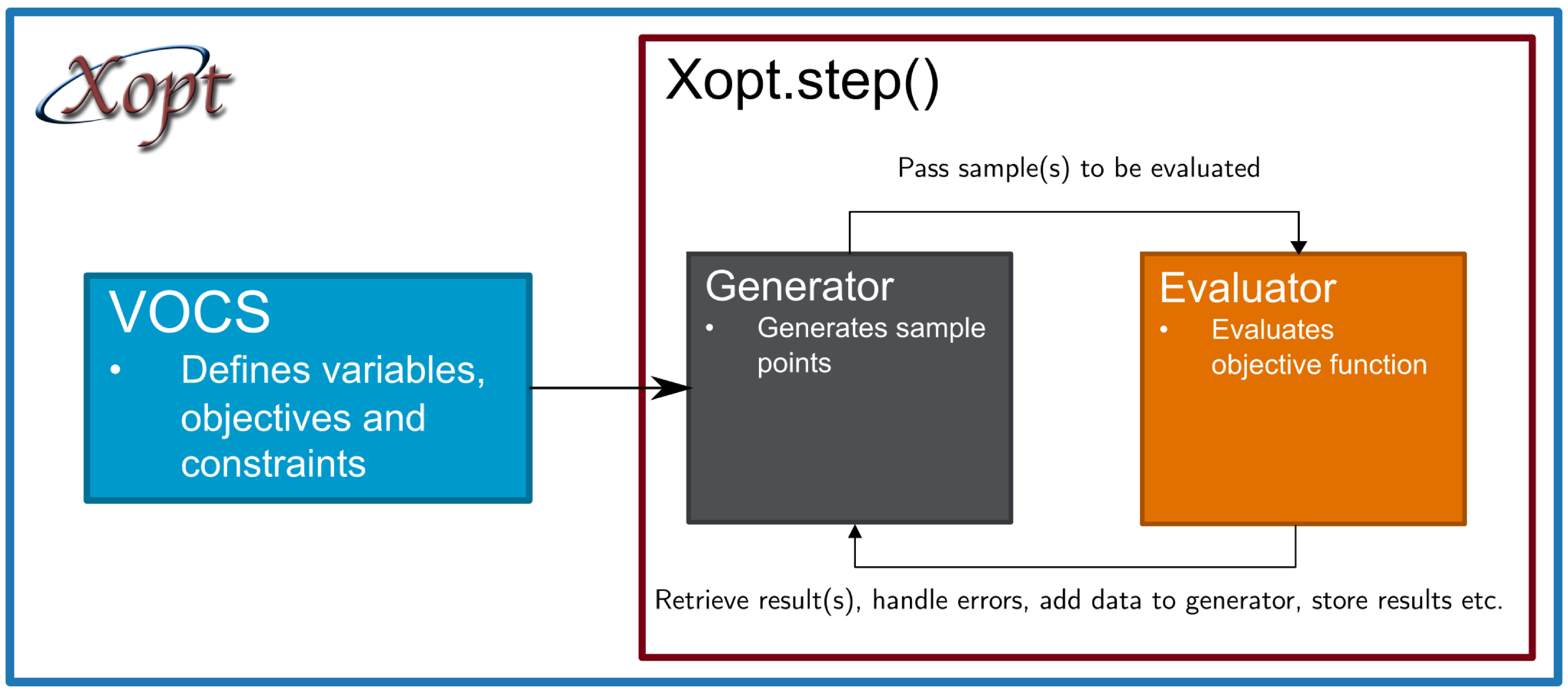
- Alternatively, Xopt objects can be created through a **Python script or interactive interface** (Jupyter notebook).

```
evaluator = Evaluator(my_function())
generator = CNSGAGenerator()
vocs = MyVOCS()

X = Xopt(
  evaluator=evaluator,
  generator=generator,
  vocs=vocs
)
```

- Finally Xopt can be initialized from data files created by **previous Xopt runs**, containing Xopt run configurations and measurements

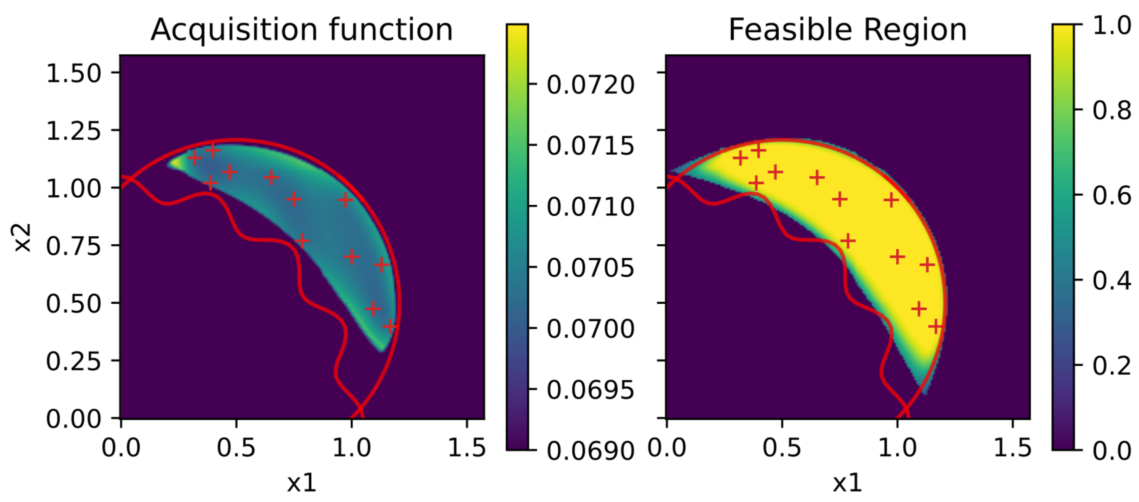
Xopt Structure



- Principal Xopt objects are **modular** and thus swappable to change algorithm type, objective evaluation, or VOCS definitions.
- Evaluators are subclasses of `concurrent.futures` Python classes, **enabling parallel evalutions** using multithreading, MPI, Dask etc. Asynchronous evaluation also available.

Generators

- **Currently availible generators**
  - Single and Multi-Objective Bayesian optimization with constraints
  - Bayesian Exploration (characterization)
  - Multi-Objective Multi-Generation BO
  - Continuous NSGA-II Genetic optimization
  - Extremum Seeking
  - Nelder-Mead (Simplex)
- **Custom generators** can be implemented by subclassing the `Generator` base class.
- Generators store objects used during optimization to allow **introspection**



Example of generator introspection with Bayesian Exploration of a constrained problem.

Example Evaluate Function

- Here we show an example evaluate function for use with an **EPICS** control system.

```
from epics import caget, caput, cainfo
import time

outputs = ["XRMS","YRMS"]
def make_epics_measurement(input_dict):
  # set inputs
  for name, val in input_dict.items():
    caput(name, val)

  # wait for inputs to settle
  time.sleep(1)

  # get output values, current time
  output_dict = caget_many(outputs)
  output_dict["time"] = time.time()

  # compute geometeric avg of beamsizes
  output_dict["RMS"] = (
    output_dict["XRMS"]*\
    output_dict["YRMS"]
  )**0.5

  return output_dict
```

Example Application - LCLS FEL Power Characterization

- **Proximal biasing** to reduce exploration step size and **constraints** to prevent charge loss.
- **Custom evaluate function** captures 80th percentile FEL power over 100 shots.
- Data stored in Pandas DataFrame objects, exported to text file with Xopt configuration
- FEL sensitivity is captured in the GP model lengthscales inside the generator object.
- Entirely executed from an interactive Jupyter notebook.