Argonne NATIONAL LABORATORY

Exploiting structures in multiobjective simulation optimization problems

Tyler Chang[a] and Stefan Wild[a→b]

[a]Mathematics and Computer Science Division,
Argonne National Laboratory

[b]Applied Mathematics and Computational Research Division,
Lawrence Berkeley National Laboratory

SIAM OP 23

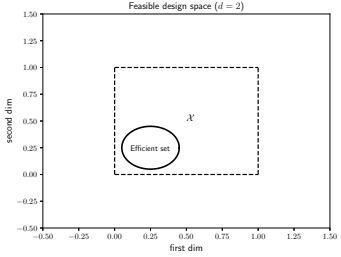
Outlines

Intro, ParMOO, and Problem Types

Early Results

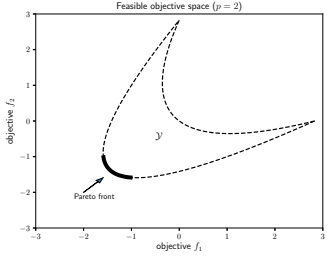
Multiobjective Optimization Problems

$$\min_{x \in \mathcal{X}} F(x)$$



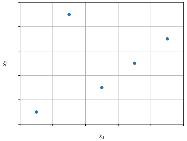
$$F : \mathcal{X} \to \mathcal{Y}$$

expensive
blackbox process

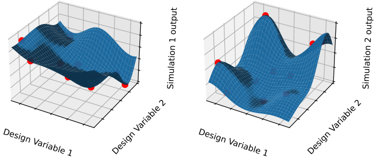


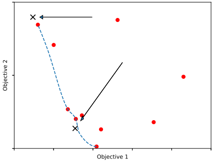
Multiobjective Response Surface Methodology

or Model-Based Optimization or Active Learning



$\mathcal{C}$



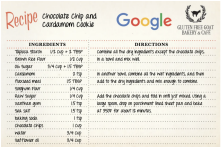


Challenge 1:

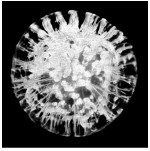
Mixed vars & problem types
+
Unusual computing environments

Commercial solutions

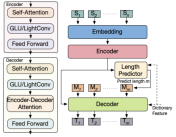
"Using Bayesian optimization for balancing metrics in recommendation systems" by Yunbao Ouyang et al. on LinkedIn Engineering Blog.



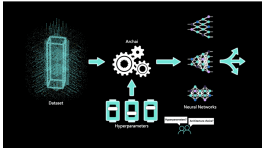
"The makings of a smart cookie" by Daniel Golovin on Google Research Blog.



"Accelerating molecular optimization with AI" by Payel Das et al. on IBM Research Blog.



"Optimizing model accuracy and latency using Bayesian multi-objective NAS" by David Eriksson et al. on Meta AI Research Blog.



"Archai can design your neural network with state-of-the-art NAS" by Shital Shah et al. on Microsoft Research Blog.

Commercial solvers

General purpose: (solver + backend)

Google – OSS Vizier + Pythia backend

[5] Song et al. OSS Vizier: distributed infrastructure and API for reliable and flexible black-box optimization. In Proc. 2022 AutoML-Conf.

Meta – BoTorch + Ax backend

[6] Balandat et al. BoTorch: a framework for efficient monte-carlo Bayesian optimization. In NeurIPS 2020.

Special purpose: (solver + special purpose deployment)

IBM – Querry-based Molecular Optimization (QMO)

[7] Hoffman et al. Optimizing molecules using efficient queries from property evaluations. Nature Machine Intelligence 4:21–31 (2022).

Microsoft – Archai for NAS

[8] Shah et al. Archai: platform for neural architecture search. Microsoft Research (Jul, 2022).

Challenge 2:

SOA blackbox optimization
+
Exploiting problem structure

SOA in blackbox optimization

"Optimization and root finding (scipy.optimize)" in SciPy v1.10.0 [9].

Stochastic dimension reduction explained in this context by Stefan [10].

SOS structure can be exploited by DFO solver POUNDERS in TAO [11].

[9] Virtanen et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods 17:261–272 (2020).

[10] Wild. Optimization and learning with zeroth-order stochastic oracles. SIAM News 56(1):1,3 (2023).

[11] Wild. Solving derivative-free nonlinear least squares problems with POUNDERS. In Advances and Trends in Optimization with Engineering Applications (2017).

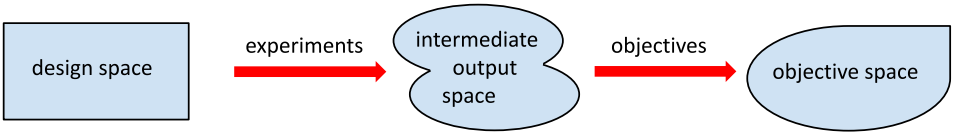
ParMOO Design Criteria

Design goals:

1. Highly customizable framework for multiobjective RSM
2. Flexible problem types (mixed-variables, constraints, etc.)
3. Easy to use, deploy, and extend (unforeseen use-cases and environments)
4. Solve large-scale problems + exploit structure and domain knowledge

[12] Chang and Wild. Designing a framework for solving multiobjective simulation optimization problems. ArXiv preprint 2304.06881 (2023).

Problem structures



Sum-of-squares structure:

$$h_i(x, S(x)) = \sum_{j \in N_i} \left(S_j(x)\right)^2$$

where each $N_1, \ldots, N_o$ is an index set.

Increases order of approximation $\Rightarrow$ increases order of convergence

Heterogeneous MOOPs:

$$h_1(x, S(x)) = S_1(x)$$
$$h_2(x, S(x)) = \|x\|^2$$

Use expensive surrogate models for h_1 (i.e., S_1) but not for h_2

Sample code

```
from parmoo import MOOP
from parmoo.optimizers import LocalGPS as gps
from parmoo.searches import LatinHypercube as lhs
from parmoo.surrogates import GaussRBF as rbf
from parmoo.acquisitions import UniformWeights as wsum
# Create MOOP object with GPS optimizer
moop = MOOP(gps)
# Add a continuous + categorical design variable
moop.addDesign({'name': "x1", 'lb': 0.0, 'ub': 1.0})
moop.addDesign({'name': "x2", 'des_type': "cat", 'levels': 3})
# Define and add a simulation function (with surrogates and search)
def s(x): return [(x["x1"]-.2)**2, (x["x1"]-.8)**2] if x["x2"]==0 else [9,9]
moop.addSimulation({'name': "sim", 'm': 2, 'sim_func': s,
                    'search': lhs, 'surrogate': rbf})
# Add 2 objectives
moop.addObjective({'name': "f1", 'obj_func': lambda x, s: s["sim"][0]})
moop.addObjective({'name': "f2", 'obj_func': lambda x, s: s["sim"][1]})
# Add 3 weighted-sum acquisition functions
for i in range(3):
    moop.addAcquisition({'acquisition': wsum})
# Solve with 5 iterations and fetch numpy struct of solutions
moop.solve(5)
results = moop.getPF()
```

ParMOO Release

Written in `Python`

Version 0.2.0 is now available on available on `pip`, `conda-forge`, and `GitHub`

`https://github.com/parmoo/parmoo`

`https://parmoo.readthedocs.io`

[15] Chang and Wild. ParMOO: A Python library for parallel multiobjective simulation optimization. *JOSS 8(82):4468 (2023).*

Example 1: Fayans EDF Model Calibration

Find params $x \in [0,1]^{13}$ to fit the Fayans model to data d_i:

$$M(\xi_i; x) \approx d_i \qquad i = 1, \ldots, 198$$

ParMOO simulation:

$$S_i(x) = M(\xi_i; x) - d_i, \qquad i = 1, \ldots, 198;$$

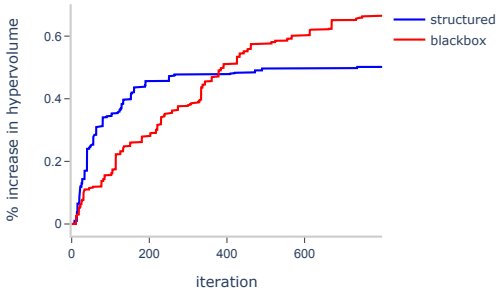
Min SOS across 3 observable classes

$$F_t = \sum_{i=1}^{m_t} \left(S_{t,i}(x)\right)^2$$

[16] Bollapragada et al. Optimization and supervised machine learning methods for fitting numerical physics models without derivatives. Journal of Physics G 48(2):024001 (2020).

Fayans Solution with ParMOO

- Approximated Fayans model using inv dist weighting on existing dataset
- Implemented parallel solver in ParMOO using libEnsemble
- Just **14-25 lines of Python code**
- Ran for **10K** sim evals
- Compared against same solver w/o exploiting SOS structure
- Structure-exploiting is better at small budgets, blackbox can be better at large budgets



Example 2: Material Manufacturing with ParMOO

Choose optimal settings for material manufacturing in a continuous flow reactor (CFR)

We know how to make a desired material, need to produce at scale:

1. **Maximize the product** (battery electrolyte: TFML)
2. Can increase temperature to **reduce reaction time**
3. Too much heat activates a side reaction; need to **minimize unwanted byproduct**

Challenges:

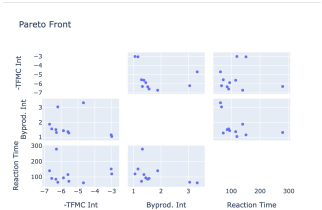
- Mixed variable types
- Heterogeneous objectives
- Must send experiments to run on CFR

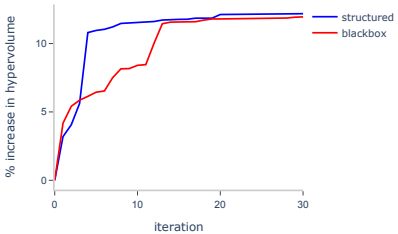
CFR Optimization with ParMOO

Extend `MOOP` class to send/receive experiment data using `MDML` library (Apache Kafka)

Used categorical variable embeddings

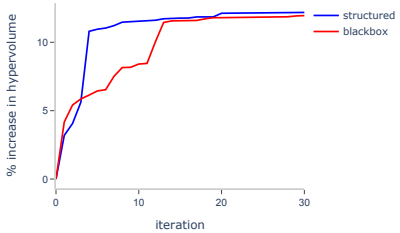
Modeled Product/Byproduct as simulations and reaction time using algebraic equation of input

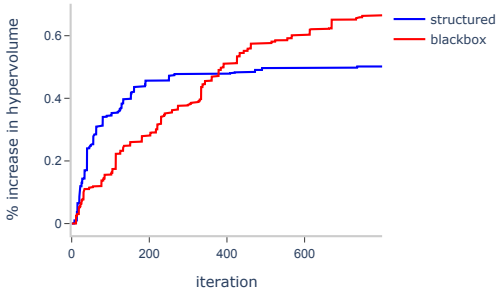




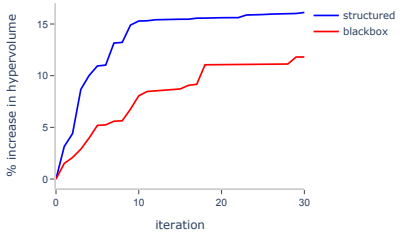
[17] Chang et al. A framework for fully autonomous design of materials via multiobjective optimization and active learning: challenges and next steps. In ICLR 2023, Workshop on ML4Materials.

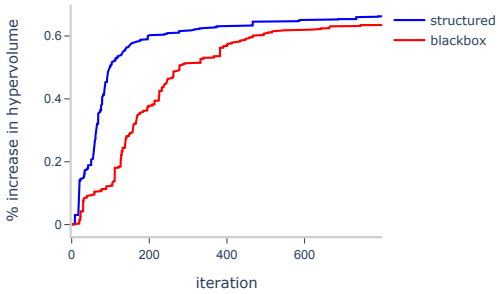
Two different problems, same issue...





Solution


% increase in hypervolume
iteration
structured
blackbox
0
5
10
15
0
10
20
30


% increase in hypervolume
iteration
structured
blackbox
0
0.2
0.4
0.6
0
200
400
600

References

[5] Song et al. OSS Vizier: distributed infrastructure and API for reliable and flexible black-box optimization. In Proc. 2022 AutoML-Conf.

[6] Balandat et al. BoTorch: a framework for efficient monte-carlo Bayesian optimization. In NeurIPS 2020.

[7] Hoffman et al. Optimizing molecules using efficient queries from property evaluations. Nature Machine Intelligence 4:21–31 (2022).

[8] Shah et al. Archai: platform for neural architecture search. Microsoft Research (Jul, 2022).

[9] Virtanen et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods 17:261–272 (2020).

[10] Wild. Optimization and learning with zeroth-order stochastic oracles. SIAM News 56(1):1,3 (2023).

[11] Wild. Solving derivative-free nonlinear least squares problems with POUNDERS. In Advances and Trends in Optimization with Engineering Applications (2017).

[12] Chang and Wild. Designing a framework for solving multiobjective simulation optimization problems. ArXiv preprint 2304.06881 (2023).

[15] Chang and Wild. ParMOO: A Python library for parallel multiobjective simulation optimization. JOSS 8(82):4468 (2023).

[16] Bollapragada et al. Optimization and supervised machine learning methods for fitting numerical physics models without derivatives. Journal of Physics G 48(2):024001 (2020).

[17] Chang et al. A framework for fully autonomous design of materials via multiobjective optimization and active learning: challenges and next steps. In ICLR 2023 Workshop on ML4Materials.

Resources

GitHub: `github.com/parmoo/parmoo`
Docs: `parmoo.readthedocs.io`
PyPI: `pip install parmoo`
Conda: `conda install --channel=conda-forge parmoo`

E-mail: `tchang@anl.gov`
E-mail: `parmoo@mcs.anl.gov`

Chang and Wild. JOSS 8(82):4468 (2023)

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, SciDAC program under contract number DE-AC02-06CH11357.