A FRAMEWORK FOR FULLY AUTONOMOUS DESIGN OF MATERIALS VIA MULTIOBJECTIVE OPTIMIZATION AND ACTIVE LEARNING: CHALLENGES AND NEXT STEPS

Tyler H. Chang[1]*, **Jakob R. Elias**[2], **Stefan M. Wild**[1,3]†, **Santanu Chaudhuri**[2,4]
and Joseph A. Libera[2]
[1]Mathematics and Computer Science Division, Argonne National Laboratory
[2]Applied Materials Division, Argonne National Laboratory
[3]Applied Mathematics and Computational Research Division,
Lawrence Berkeley National Laboratory
[4]Department of Civil, Materials, and Environmental Engineering, University of Illinois Chicago
{tchang,jelias,schaudhuri,jlibera}@anl.gov, wild@lbl.gov

ABSTRACT

In order to deploy machine learning in a real-world self-driving laboratory where data acquisition is costly and there are multiple competing design criteria, systems need to be able to intelligently sample while balancing performance trade-offs and constraints. For these reasons, we present an active learning process based on multiobjective black-box optimization with continuously updated machine learning models. This workflow is built on open-source technologies for real-time data streaming and modular multiobjective optimization software development. We demonstrate a proof of concept for this workflow through the autonomous operation of a continuous-flow chemistry laboratory, which identifies ideal manufacturing conditions for the electrolyte 2,2,2-trifluoroethyl methyl carbonate.

1 INTRODUCTION

Data-driven automated laboratories, also called self-driving laboratories, can significantly accelerate molecular synthesis and materials discovery. A key technical challenge of fully autonomous and artificial intelligence-assisted laboratory design is to effectively collect and utilize data from multiple complex processes in order to inform future experimentation. One long-standing template for utilizing experimental and simulation data is the multiresponse surface methodology (RSM) (Myers et al., 2016), whereby initial data sets are gathered through design of experiments and then statistical models are built for each quantity of interest, analyzed, and hypothesis tested iteratively.

In modern scientific and engineering settings, several common paradigms could be considered as specific implementations of this discovery framework, the most common of which are active learning and model-based optimization techniques such as Bayesian optimization. To account for multiple competing criteria, we utilize an active learning framework based in multiobjective optimization, which utilizes surrogates (such as Gaussian processes), optimization solvers, and multicriteria data acquisition in a closed feedback loop. Several definitions of active learning exist, including both adaptive sampling to model complex processes with uniform accuracy (Sapsis & Blanchard, 2022) and iterative selection of candidate designs to improve a global machine learning model that drives optimization convergence (Yuan et al., 2023). In this paper we use the latter definition, since global model accuracy is not a reasonable goal when using an extremely limited budget for experiments, as in our case.

We present our approach to integrate streaming experimental data from a real-world, fully automated continuous-flow chemistry setup into a customizable machine learning (ML) and multiobjective optimization framework for steering the design of battery electrolytes. In Section 2 we briefly summarize the problem background, focusing on relevant techniques and challenges. Section 3 introduces

*Correspondence to tchang@anl.gov

†Work was primarily performed at Argonne, prior to changing institutions

our framework for addressing these challenges, which is based on integration between the multiobjective optimization library ParMOO (Chang & Wild, 2023) and the data streaming platform for Manufacturing Data and Machine Learning (MDML) (Elias et al., 2020). In Section 4 we present our results from the synthesis of the battery electrolyte 2,2,2-trifluoroetheyl methyl carbonate (TFMC) with an integrated automated feed, continuous-flow reactor (CFR), and nuclear magnetic resonance (NMR) setup. This experiment does not constitute novel material discovery, but in Section 5 we discuss how our framework will allow for extension to the discovery domain.

2 Background and Key Challenges

There is no shortage of options when it comes to multiobjective optimization solvers and libraries. Notable techniques include genetic algorithms (Blank & Deb, 2020), search-based methods (Le Digabel, 2011), and Bayesian optimization (Balandat et al., 2020). We note that we are not the first paper to take a multiobjective approach to chemical synthesis (Häse et al., 2018; Shields et al., 2021). However, setting up these solvers to integrate with scientific computing and laboratory environments is a nontrivial task, where each optimization software must be integrated as an optimization service (Raghunath et al., 2017) or function (Chang et al., 2020) in the broader library. This approach is appropriate for integrating with computer simulation environments; but in order to successfully integrate with heterogeneous data sources, a more flexible approach is needed (Barba-González et al., 2018).

Another challenge in applying optimization solvers for autonomous material discovery is exploiting structures to address problem complexity. Optimization solvers tend to perform well over continuous input spaces; but when some of the design variables are complex (such as chemical networks), the problem becomes combinatorial for classical solvers unless they are carefully tailored to the problem at hand. In the context of material discovery, the molecular structure can be embedded by using either its molecular descriptors (Shields et al., 2021) or problem-specific latent-space representations (Hoffman et al., 2022). Similarly, after performing experiments, the raw outputs of these chemical processes may consist of large volumes of time series data or spectral measurements, and the majority of this information is lost when the data is postprocessed to derive meaningful objectives. However, modern simulation optimization solvers can take advantage of these raw simulation outputs as a method for exploiting the physical structure in the problem (Wild, 2017).

Thus, our challenge is to exploit problem-specific structures and embeddings in a way that generalizes to a variety of applications while modeling data from multiple chemical processes. This requires a multiobjective optimization framework that is flexible enough to utilize customized problem definitions, while still leveraging state-of-the-art modeling techniques, and that can be coupled with a flexible data-streaming service.

3 A Framework for Autonomous Experimentation

To address our challenges, we use the multiobjective optimization library ParMOO (Chang & Wild, 2023) to manage multiple surrogate models, exploit domain knowledge, and utilize problem embeddings through its customizable modular framework. The data-streaming service MDML (Elias et al., 2020) provides an event-driven architecture using an Apache Kafka instance with publishers and subscribers to pass data between laboratory workstations, servers, supercomputers, and more. Once an MDML instance has been created—typically on a server separate from experiments and machine learning codes—any number of experiments or clients can begin streaming data. Additional Kafka instances can be added to provide horizontal scaling if needed. By utilizing MDML as ParMOO's simulation distribution backend, we are able to stream scientific data from multiple sources, process that data, and issue requests for new experiments in a closed loop.

Additionally, both of these tools are open-source, lightweight Python libraries, which can be installed on a typical laboratory workstation. To integrate ParMOO with MDML, we have extended ParMOO's `MOOP` class, overwriting the `solve` method to produce Kafka requests for experimental data via MDML. This means that the MDML platform handles data streaming and recording through its usual methods and is responsible for issuing experiments or simulations, logging data, and storing results. In this work we used two servers running Kafka brokers as our MDML instance, providing FuncX (Chard et al., 2020) endpoints for our analysis tasks. We then used a lab workstation running

LabVIEW (which directly connects to the CFR and NMR) and a compute node running a ParMOO solver instance as analysis client processes. Figure 5 in Appendix A.2 further illustrates this setup.

For more information on how ParMOO facilitates exploiting problem structure, the specific structure and components of its modeling and candidate selection, and the techniques used in this paper, see Appendix A.1. For more information on how MDML handles data streaming from heterogeneous sources, see Appendix A.2.

4 Chemical Synthesis with ParMOO and MDML

In this section we explore electrolyte production in an autonomously operated CFR using the framework from Section 3. Compared with traditional batch processing, CFRs can provide a means to rapidly test reagent combinations, experiment with high reaction temperatures, and measure yield and specificity when equipped with online characterization. However, manual operation can be very expensive, time-consuming, and inefficient. Thus, automating all aspects of the experimentation process is attractive in order to maximize equipment utilization and minimize labor. Furthermore, when coupled with ML algorithms, the total number of experiments required to reach optimal parameters is also minimized, in comparison with a traditional exhaustive search and experimental design techniques.

For this example we are interested in identifying ideal manufacturing conditions for the battery electrolyte TFMC in a CFR, based on the reaction illustrated in Figure 1. Note that in this example we are using a predetermined pair of reagents, solvent, and base and a limited range of values for the reaction conditions. Specifically, we must select an optimal flow rate, reaction temperature, and equivalence ratio for the two reagents. To reduce the reaction time and increase production speed, we would like the reaction to be carried out at high temperatures. However, high temperatures can also activate a side reaction that produces an unwanted byproduct and reduces the purity of the product. Therefore, we are seeking chemical mixtures and conditions that produce large amounts of TFMC but small amounts of the byproduct trifluoroethanol (TFE).

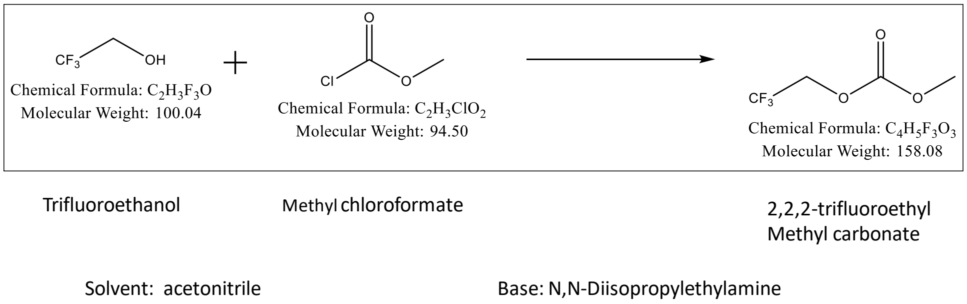
Chemical Formula: $C_2H_3F_3O$
Molecular Weight: 100.04
Chemical Formula: $C_2H_3ClO_2$
Molecular Weight: 94.50
Chemical Formula: $C_4H_5F_3O_3$
Molecular Weight: 158.08
Trifluoroethanol
Methyl chloroformate
2,2,2-trifluoroethyl Methyl carbonate
Solvent: acetonitrile
Base: N,N-Diisopropylethylamine

Figure 1: Diagram of the basic reaction optimized for the production of the electrolyte 2,2,2-trifluoroethyl methyl carbonate (TFMC).

To summarize our self-driving laboratory setup, we created an MDML client process running a modified ParMOO solver, as described in Section 3, which produced requests for experiments and consumed experimental results. Specific ParMOO solver settings are given in Appendix A.1, and specific MDML settings are given in Appendix A.2. Experiment requests were consumed by a laboratory workstation running LabVIEW, which operates a multiport VICI valve controlling the flow of reagents into a Vaportec CFR. Outputs from the CFR were shunted into a Magritek NMR, which recorded peak areas for both the product (TFMC) and byproduct (TFE). The operation of this setup and collection of NMR data were controlled by a workstation running a control program written in LabVIEW and returned to the Kafka brokers via an MDML producer. Additional details on this lab setup are given in Appendix B.

In our case the correspondence of each peak to TFMC or TFE is determined by the ordering of the peaks, which had been identified as a suitable strategy for this experiment via prior experimentation and domain knowledge. Ultimately, only the peak areas are used in our objective calculations.

However, this reduction sacrifices valuable information about the shape and location of each peak, which could potentially be used to improve surrogate model accuracy (Wild, 2017). Additionally, we acknowledge that this might not be a suitable strategy in the general case.

Experiment: Optimal Manufacturing Conditions

Although one can experiment with different options, we have fixed the solvent and base for this experiment to a predetermined pairing that has shown promising results. We focus on identifying optimal manufacturing conditions, such as flow rates, reaction times, and reaction temperatures. In order to ensure safe operations of the CFR, upper and lower bound constraints were given for each of these values, as shown in Table 1.

Parameter	Lower bound	Upper bound
Temperature (degrees C)	40	150
Reaction time (seconds)	60	300
Equivalence ratio (no units)	0.9	2

Table 1: Design variables and bound constraints for initial experiment.

One of the key challenges for this problem is to achieve convergence on an extremely limited budget. Since we are running real experiments on a CFR, each experiment has significant costs in terms of financial cost, resources, and time. As examples, the raw materials (reagents, solvents, and bases) used in these experiments must have high purity, which comes at a significant financial cost; the machinery (CFR and NMR) have operating costs; and the experiments themselves are time-consuming, requiring up to 10 minutes each. Therefore, we cannot afford to perform hundreds of experiments in this setting, as would be typical in a Bayesian optimization feedback loop.

To directly control the balance between exploration and exploitation and encourage significant exploitation on a limited budget (while uncertainties are still high in some regions of the space), we favor an active learning implementation that is more similar to a traditional statistical RSM, where an initial design of experiments of predetermined size is allocated to drive exploration and initial model accuracy and then all future budget is allocated to exploiting our model, similarly as in Chang et al. (2022). We note that, in this strategy, global convergence is determined by the density of the initial design of experiments and is difficult to guarantee for an unknown function. However, this approach is also more flexible for extension to domain-specific surrogate modeling techniques, which will aid in the transition to chemical discovery (Yuan et al., 2023).

ParMOO was instructed to maximize the product and minimize the byproduct subject to the constraints in Table 1. After a 15-point initial exploratory design of experiments, ParMOO was run for 9 additional model-exploiting iterations with a batch size of three experiments per iteration (i.e., 27 additional iterative experiments driven by the optimization framework). Because of lack of continued improvement over several batches of experiments, not all experiments allocated in the budget were carried to completion. In total, 41 experiments were completed; the corresponding data is shown in Figure 2. Since we are attempting to maximize the product (TFMC) and minimize the byproduct (TFE), the ideal solution would be located in the bottom-right corner. Note that the clustering of late (high index) runs in the bottom-right corner indicates convergence to optimal manufacturing conditions with an extremely limited budget.

5 Path Forward

In Section 4 we were successful in optimizing the manufacturing conditions for a predetermined material, but additional work is needed in order to broaden our scope to the original goal of material *discovery*. We fully anticipate that the code will continue to find more applications in both synthesis and discovery of materials. The fundamental method is fully extendable to multireactor, multistep reactions, with more complicated hardware configurations.

As an example, we can use a similar CFR setup to synthesize more complex materials such as metal-organic frameworks (MOFs), with the nucleation and connection of organic linkers to metal nodes being fully visible to the in-line NMR. However, the detection limits and reaction conditions for this

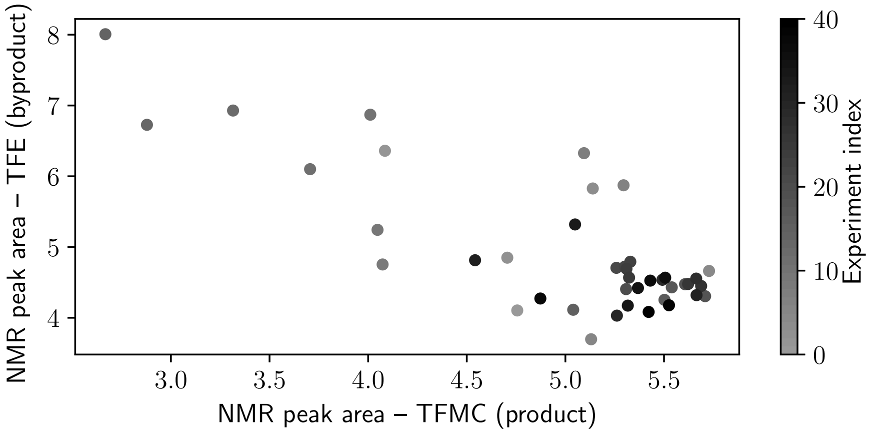

Figure 2: Product (TFMC) vs. byproduct (TFE) areas recorded by NMR for 41 continuous-flow chemistry experiments steered by ParMOO using MDML. Point colors are coded by experiment index, with larger indices indicating later runs.

problem generally take months to fully explore and optimize. The size of MOF libraries and linkers is increasing, but the convergence rates of high-throughput synthesis optimization tools are lacking. By utilizing a similar autonomous framework, we can extend these methods to reactors for making variations of linkers and MOFs and exploring their synthesis.

This is a much larger problem than the one that has been described in the paper. To extend to this problem, we will need to utilize domain-specific latent space embeddings, structure-exploiting optimizers that consider the full spectrum of NMR data during surrogate modeling, and heterogeneous data sources including simulation (for early exploration) followed by experimental data (for late-stage testing). In order to keep costs reasonable and extend our limited budget, it could also be interesting to consider the cost of each experiment in the acquisition function. However, we must be careful when doing this in order to still allow for expensive but insightful experiments to be evaluated.

Acknowledgments

This material was based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics and SciDAC programs under Contract Nos. DE-AC02-05CH11231 and DE-AC02-06CH11357 and by the Argonne LDRD program.

References

Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21524–21538. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/f5b1b89d98b7286673128a5fb112cb9a-Paper.pdf`.

Cristóbal Barba-González, José García-Nieto, Antonio J. Nebro, José A. Cordero, Juan J. Durillo, Ismael Navas-Delgado, and José F. Aldana-Montes. jMetalSP: A framework for dynamic multi-objective big data optimization. *Applied Soft Computing*, 69:737–748, 2018. doi: https://doi.org/10.1016/j.asoc.2017.05.004.

Julian Blank and Kalyanmoy Deb. pymoo: Multi-objective optimization in Python. *IEEE Access*, 8:89497–89509, 2020. doi: 10.1109/ACCESS.2020.2990567.

Tyler H. Chang and Stefan M. Wild. ParMOO: Python library for parallel multiobjective simulation optimization. Technical Report Version 0.1.0, Argonne National Laboratory, Lemont, IL, USA, 2022. URL `https://parmoo.readthedocs.io/en/latest`.

Tyler H. Chang and Stefan M. Wild. ParMOO: A Python library for parallel multiobjective simulation optimization. *Journal of Open Source Software*, 8(82):4468, 2023. doi: 10.21105/joss.04468.

Tyler H. Chang, Jeffrey Larson, Layne T. Watson, and Thomas C. H. Lux. Managing computationally expensive blackbox multiobjective optimization problems using libEnsemble. In *Proc. 2020 Spring Simulation Conference (SpringSim 2020), the 28th High Performance Computing Symposium (HPC '20)*, pp. Article No. 31. SCS, 2020. doi: 10.22360/SpringSim.2020.HPC.001.

Tyler H. Chang, Layne T. Watson, Jeffrey Larson, Nicole Neveu, William I. Thacker, Shubhangi Deshpande, and Thomas C. H. Lux. Algorithm 1028: VTMOP: Solver for blackbox multiobjective optimization problems. *ACM Transactions on Mathematical Software*, 48(3):Article No. 36, 2022. doi: 10.1145/3529258.

Ryan Chard, Yadu Babuji, Zhuozhao Li, Tyler Skluzacek, Anna Woodard, Ben Blaiszik, Ian Foster, and Kyle Chard. funcX: A federated function serving fabric for science. In *Proc. 29th International Symposium on High-Performance Parallel and Distributed Computing (HPDC '20)*, pp. 65–76. ACM, 2020. doi: 10.1145/3369583.3392683.

Jakob R. Elias, Ryan Chard, Joseph A. Libera, Ian T. Foster, and Santanu Chaudhuri. The manufacturing data and machine learning platform: Enabling real-time monitoring and control of scientific experiments via IoT. *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, pp. 1–2, 2020. doi: 10.1109/WF-IoT48130.2020.9221078.

Florian Häse, Loïc M Roch, and Alán Aspuru-Guzik. Chimera: enabling hierarchy based multiobjective optimization for self-driving laboratories. *Chemical science*, 9(39):7642–7655, 2018. doi: 10.1039/C8SC02239A.

Samuel C. Hoffman, Vijil Chenthamarakshan, Kahini Wadhawan, Pin-Yu Chen, and Payel Das. Optimizing molecules using efficient queries from property evaluations. *Nature Machine Intelligence*, 4(1):21–31, 2022. doi: 10.1038/s42256-021-00422-y.

Sébastien Le Digabel. Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. *ACM Transactions on Mathematical Software*, 37(4):Article No. 44, 2011. doi: 10.1145/1916461.1916468.

Raymond H. Myers, Douglas C. Montgomery, and Christine M. Anderson-Cook. *Response Surface Methodology: Process and Design Optimization Using Designed Experiments*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 4 edition, 2016. ISBN 9781118916032.

Chaitra Raghunath, Tyler H. Chang, Layne T. Watson, Mohamad Jrad, Rakesh K. Kapania, and Raymond M. Kolonay. Global deterministic and stochastic optimization in a service oriented architecture. In *Proc. 2017 Spring Simulation Conference (SpringSim 2017), the 25th High Performance Computing Symposium (HPC '17)*, pp. Article No. 7, Virginia Beach, VA, USA, 2017. SCS. doi: 10.22360/springsim.2017.hpc.023.

Themistoklis P. Sapsis and Antoine Blanchard. Optimal criteria and their asymptotic form for data selection in data-driven reduced-order modelling with Gaussian process regression. *Philosophical Transactions of the Royal Society A*, 380(2229):20210197, 2022. doi: 10.1098/rsta.2021.0197.

Benjamin J. Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I. M. Alvarado, Jacob M. Janey, Rryan P. Adams, and Abigail G. Doyle. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590(7844):89–96, 2021. doi: 10.1038/s41586-021-03213-y.

Stefan M. Wild. Solving derivative-free nonlinear least squares problems with POUNDERS. In Tamas Terlaky, Miguel F. Anjos, and Shabbir Ahmed (eds.), *Advances and Trends in Optimization with Engineering Applications*, pp. 529–540. SIAM, 2017. doi: 10.1137/1.9781611974683.ch40.

Xiaoze Yuan, Yuwei Zhou, Qing Peng, Yong Yang, Yongwang Li, and Xiaodong Wen. Active learning to overcome exponential-wall problem for effective structure prediction of chemical-disordered materials. *Nature Computational Materials*, 9(1):12, 2023.

A Software Details

A.1 ParMOO

ParMOO is a framework for customizing and deploying multiobjective simulation optimization solvers (Chang & Wild, 2023). The key distinction in ParMOO is the difference between a *simulation* and an *objective*, as depicted in Figure A.1. This abstraction and the utilization of custom latent-space embedding tools are the primary mechanisms by which ParMOO exploits problem structure.

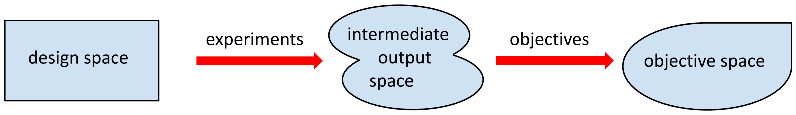

Figure 3: ParMOO models complex processes separately from objectives, in order to exploit physical structure in how the objectives are defined.

To create a ParMOO instance, users must first specify all design variables, data sources (e.g., experiments and simulations), problem constraints, and problem objectives. For complex design variable types, ParMOO is capable of automatically generating latent space embeddings or utilizing customized embedding tools. Next, users provide

- any pre-existing data or a technique for generating an initial design of experiments;
- surrogate models for learning each of the simulation outputs;
- optimization solvers for the surrogate problem; and
- one or more data acquisition functions for determining how ParMOO will blend objectives.

When run, ParMOO will evaluate the initial experimental design, then iteratively generate batches of new experiments to be distributed for evaluation across all data sources, by following the standard response surface methodology. This feedback loop is illustrated in Figure 4. For further information on this process, see ParMOO's online documentation (Chang & Wild, 2022).

For the demonstration described in Section 4, ParMOO was configured to use a 15-point Latin hypercube design of experiments and a Gaussian RBF (the mean function for a Gaussian process) surrogate, and the surrogate problem was solved via generalized pattern search. Three acquisition functions were provided (resulting in candidate batches of size three), two of which used the epsilon-constraint method to scalarize the problem and the third of which used a fixed 50-50 weighting of the two objectives (i.e., maximize the product and minimize the byproduct).

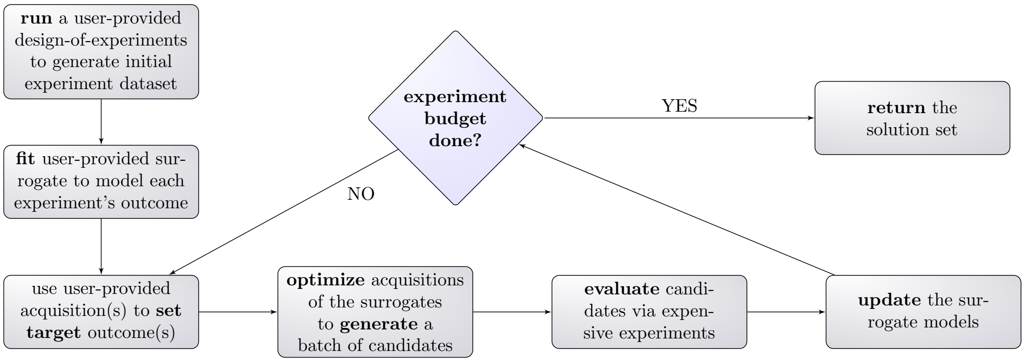
run a user-provided design-of-experiments to generate initial experiment dataset
fit user-provided surrogate to model each experiment's outcome
use user-provided acquisition(s) to set target outcome(s)
optimize acquisitions of the surrogates to generate a batch of candidates
evaluate candidates via expensive experiments
update the surrogate models
experiment budget done?
YES
NO
return the solution set

Figure 4: Program control flowchart, depicting how ParMOO combines user-provided techniques to optimize complex processes via active learning.

A.2 MDML

The MDML platform provides cyberinfrastructure to standardize the research and operational environment to act on scientific data streams and integrate ML in the loop to steer or optimize experiments. MDML has been designed to meet the needs of in situ measurements for accelerating scalable materials manufacturing while providing capabilities that can be easily adapted to any scientific domain. MDML enables users to construct rich, data-oriented analysis pipelines that span disparate computational environments.

Underlying MDML is the widely used, distributed event streaming platform named Apache Kafka. This allows scientists using MDML to follow an event-driven architecture. Clients connected to the MDML's Kafka service are able to publish and subscribe to one or many data channels that they have created. Clients here include researchers themselves, software programs controlling/monitoring experiments, and ML models. Given this architecture, any relevant experiment data can be streamed anywhere and fed into client processes running ML models (such as ParMOO) in a unified format. Additionally, when the experimental data collection process can be fully automated (as with our integrated feed, CFR, and NMR setup), this enables a user-free feedback loop.

The architecture of our MDML instance utilizes two servers to run two Kafka brokers and their various components, databases for long-term storage of streaming data, and FuncX's (Chard et al., 2020) serverless compute endpoints for spawning analysis jobs that act on streamed data. A diagram of this architecture is shown in Figure 5.

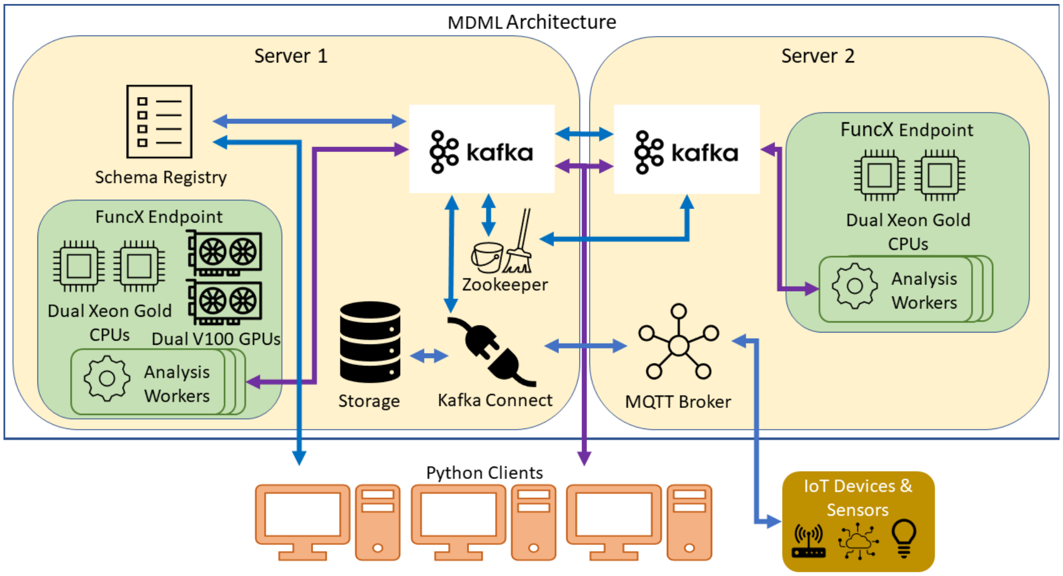

Figure 5: Diagram of the MDML architecture spread across two servers. Purple arrows represent data streams from producers and consumers.

In the diagram, the CFR is illustrated as a Python client since CFR data is collected locally via LabVIEW and streamed out via a Python script. The CFR also receives suggested experiments from ParMOO via a Python client and loads this information into LabVIEW. Here ParMOO is represented as an analysis worker. It receives results published from the CFR client, decides which experiment should be performed next, and publishes its own data, which is received by the CFR, completing the loop. The last important component to note here is Kafka, which is responsible for ushering produced data messages to the appropriate data consumers (i.e., CFR results to ParMOO and ParMOO's suggested experiments to CFR). Several MDML features shown in this diagram were not used for the automation of the CFR. These include the MQTT broker for collecting data from IoT devices and sensors; databases for storage; and Kafka Connect, which automatically converts data from the latter two sources to Kafka topics and vice versa.

B Experimental Methods

TO perform autonomous experimentation on the CFR using ParMOO and the MDML in a closed feedback loop, we created a MDML client process running a modified ParMOO solver, as described in Section 3. Next, we created an additional experiment-running client on a laboratory workstation that directly controlled the CFR. Then, the MDML host was run on a remote laboratory mainframe to broker requests and collect data.

As described in Section A.1, the ParMOO client consists of a ParMOO solver that was configured to use a 15-point Latin hypercube design of experiments, Gaussian RBF surrogate model, generalized pattern search optimizer, and three acquisition functions (two using the epsilon-constraint method for Pareto front exploration and the third using a fixed weighting of the two objectives). Following the active learning framework, the ParMOO client generated an initial design of experiments and then posted Kafka producer requests for these designs, listened for results data through a Kafka consumer, used those results to update its Gaussian models, optimized those models to target new experimental designs, and then posted new producer requests in an iterative feedback loop. In order to reduce experiment times on the CFR, experiment requests within a single batch were sorted by the reaction temperature. This process was beneficial since it takes additional time to stabilize temperatures when adjusting reaction temperatures by large amounts.

For the experiment client, a lab computer running the LabVIEW control program (LVCP) was configured to consume Kafka requests from the ParMOO client. The LVCP consumer polled for a producer request in the form of a valid experiment parameter test set. Whenever a valid experiment parameter test set was received from the ParMOO client, an experiment was set in motion by automated deployment of the provided set on the CFR and NMR setup, described in the next paragraph. After completion of a test, the LVCP posted the result as a Kafka producer and awaited the next data point as a consumer.

In order to perform the physical experiments, a CFR apparatus manufactured by Vaportec was used to carry out continuous-flow reactions. Inputs to the A and B channels of the Vaportec unit were provided by multiport VICI valves connecting up to 15 possible reagents to the CFR. The output from the CFR was shunted through a Magritek NMR equipped with a flow-through cell. The selector valves, Vaportec reactor, and Magritek NMR were fully controlled by using a Labview control program. A photograph illustrating this setup is given in Figure 6.

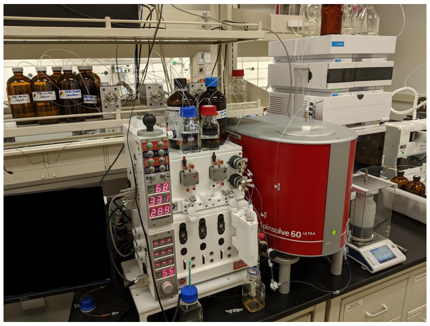

Figure 6: Our laboratory setup for performing the physical experiments. From left to right on the workbench are the lab workstation running the LVCP, the Vaportec CFR unit with automated feed, and the Magritek NMR recording results.

In order to calibrate the equipment, the reactor was manually set up and NMR spectra tests were run on pure solvents. Operating parameter ranges were also set. Whenever a valid experiment parameter set was received, the experiment was initiated from LVCP. Steady state was determined by monitoring the integrated NMR peak areas provided by the Magritek software, and test completion was determined by the integrated peak area falling below a standard deviation criterion set in advance. To conclude a single experiment, data acquisition by the NMR required stopping of flow though its flow-through cell, whereas it was not possible to stop the flow in the CFR. Therefore a bypass loop was employed on the CFR in order to terminate each experiment.

C Open Access Statement

We acknowledge that it will not be possible for researchers to reproduce our results without access to highly specialized laboratory equipment. However, in order to facilitate open access and reproducibility of results as much as possible, the source code for using ParMOO to create MDML consumer/producer services and a variation of our script used for Section 4 (demonstrating solver settings, but not running the experiments) is publicly available at `https://github.com/parmoo/cfr-materials`. We have also included a copy of the raw MDML-recorded outputs (a JSON file) from our experiment, as well as a Python script for postprocessing these results to produce the plot in Figure 2.