This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

Machine Learning Enabled Design Automation and Multi-Objective Optimization for Electric Transportation Power Systems

Derek Jackson[1](*), *Student Member*, *IEEE*, Syrine Belakaria[2](*), Yue Cao[1], *Member*, *IEEE*, Janardhan Rao Doppa[2], Xiaonan Lu[3], *Member*, *IEEE*

Abstract—**This paper presents an automated design and optimization framework for electric transportation power systems (ETPS), enabled by machine learning (ML). The use of physical models, simulations, and optimization methods can greatly aid the engineering design process. However, when considering the optimal co-design of multiple inter-dependent subsystems that span multiple physical domains, such model-based simulations can be computationally expensive, and traditional metaheuristic optimization methods can be unreliable. Bayesian optimization (BO), a ML framework, paves one feasible pathway to realize an efficient design process practically. However, current state-of-the-art BO algorithms are non-compatible or perform poorly when applied to system-level ETPS design with multiple objectives and constraints. This paper proposes a novel BO algorithm referred to as Max-value Entropy Search for Multi-objective Optimization with Constraints (MESMOC) to solve multi-objective optimization (MOO) problems with black-box constraints that can only be evaluated through design simulations. After full presentation of the algorithm, MESMOC is applied to a realistic ETPS design case using a heavy-duty electric vertical-takeoff-landing (eVTOL) urban aerial vehicle (UAV) power system. Two MOO experimental trials show a drastic reduction in the number of design simulations to discover a high-quality Pareto front. In Trial 1, MESMOC uncovered the entire Pareto front while only requiring to explore ~4% of the design space. With expanded design parameters and a larger design space in Trial 2, a near-complete but high-quality Pareto front was uncovered. Both trials compared MESMOC to the popular genetic algorithm NSGA-II and another BO algorithm PESMOC, showing superior performance.**

Index Terms—**Electric Transportation, Power System Design, Model-based Design, Multi-Objective Optimization, Design Automation, Machine Learning, Bayesian Optimization, Aviation, UAV**

[1]Derek Jackson and Yue Cao are with School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331, USA. (email: jacksder@oregonstate.edu, yue.cao@oregonstate.edu)

[2]Syrine Belakaria and Janardhan Rao Doppa are with School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99163, USA.

[3]Xiaonan Lu is with Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122, USA.

(*) Derek Jackson and Syrine Belakaria are co-first authors with equal contributions to this work.

This work was supported in part by the Oregon State University Foundation and in part by the U.S. National Science Foundation (NSF) CAREER award IIS-1845922 and OAC-1910213.

A preliminary version of this paper was presented in an oral session at the 2020 IEEE Energy Conversion Congress & Expo (ECCE), Detroit, Michigan, USA [28].

I. Introduction

A. Background

Electrification of vehicles introduces significant complexity to electrical system design. Electric transportation power systems (ETPS) include many interacting subsystems that span across multiple disciplines, requiring collaboration among various domain experts. Model-based design, with mathematical models and simulations representing real-physical systems, evaluates the performance of a design without the need of first building expensive hardware prototypes [1]. Model-based design complements the multidisciplinary design optimization (MDO) used in the automotive and aerospace industries, both known for their large and complex systems. While simulations are more efficient than hardware prototyping, design space exploration remains a time-consuming procedure. Autonomous and efficient methods to quickly find optimal designs are desired.

Model-based design begins at the system-level, where high-level models of a proposed architecture are developed. The selected optimal architecture and system parameters then serve as the design specifications for each subsystem, such as the battery pack, DC-AC inverter, or electric motor. An emphasis on system-level functionality is beneficial, as the overall performance of a large ETPS outweighs the performance of an individual component. For example, selecting a state-of-the-art power electronics converter may not be feasible due to size, weight, or temperature constraints when integrated with the rest of the system. Effective system-level design considers these constraints during the exploration stage to ensure the optimal solution is feasible.

System-level design relies on expensive simulations to explore feasibility and performance, especially in the early development process. Simulations require models with multi-physics domains (e.g., electrical, mechanical, thermal) that run on timescales of micro-seconds, milli-seconds, seconds, and minutes depending on the required fidelity and can be computationally slow. For example, a 300-minute mission in a more electric aircraft thermally integrated power system simulates one design candidate in ~15 minutes [2]. Even if simulation times were reduced by an order of magnitude, the high number of design candidates, typically in the order of thousands, can still slow down the evaluation of all designs. Additionally, modeling uncertainties using Monte Carlo simulations further increases computation time [3]. Such simulations can take a few hours to several days to explore the entire design space and are usually tailored for a particular drive cycle or mission profile. When another mission profile is replaced, such computations must start over, creating lengthy engineering processes.

This paper focuses on formulating the design automation problem and increasing optimization efficiency at the system-level. Optimization methods capable of efficiently searching through a large design space reduce the need for human interference. Engineers make decisions that limit the design space size, or otherwise it is computationally unreasonable to exhaustively search a complete design space. By shifting the decision-making process to an optimization algorithm, design automation can be achieved while also enabling the efficient search of large design spaces. Hence, effective design automation relies on the performance and efficiency of the optimization algorithm.

B. Related Works

When considering the multidisciplinary design of ETPS, it is often impossible to optimize all objectives at once due to their conflicting nature, such as minimizing energy consumption, total weight, and cost. This leads to a set of Pareto optimal solutions where an objective cannot be improved without degrading another – the outcome of multi-objective optimization (MOO). It is important to distinguish between single-objective optimization (SOO) and MOO. The goal of SOO is to find the global optimum based on one criterion, whereas, with MOO, a global optimum may no longer exist [4]. The two approaches to MOO are to either linearly combine multiple SOO problems with weights or apply metaheuristic methods [5].

The former combined SOO approach is less favorable as it can lead to aggressive exploitation behavior resulting in sub-optimal solutions [6]. Indeed MOO can be solved by using scalarization to reduce it to SOO that will optimize the weighted objective function $\sum_{i=1}^{K} w_i \cdot f_i$, where w_i stands for the weights of each individual objective function f_i. The performance of this SOO approach critically depends on the scalarization weights. However, there is no algorithmic or automated procedure in the literature to specify the scalarization weights. Reducing a problem into a single objective also requires more human intervention through prioritizing each objective. Prior work [7][8] typically employs random scalarization weights. ParEGO [8] is a prototypical example algorithm, whereas MESMO [6] has showed that solving the MOO directly does significantly better than SOO algorithms such as the ParEGO. Other SOO algorithms such as TuRBO [9], BOHB [10], and HesBO [11] can be used with scalarization to solve MOO problems, but they suffer from the same drawback of scalarization (i.e., no method specifies scalarization weights and random scalarization performs poorly), and none of these algorithms can handle black-box constraints.

For the latter metaheuristic approach, algorithms capable of true Pareto front optimization primarily use gradient-free methods, such as genetic algorithms (GAs), particle swarm optimization (PSO), ant colony optimization (ACO), and divided rectangle (DIRECT) [5][7][12], to address the nonlinear, nonconvex, and combinatorial nature of ETPS system-level design. Such metaheuristic methods have been used for the design of a solar-powered hybrid airship [13], an all-electric vehicle [14], and hybrid electric vehicles [15][16], to name a few. While these metaheuristic methods have shown to be somewhat effective optimizers, they suffer from the following limitations: 1) requiring a large number of design evaluations, which may not be practical when design simulations are computationally expensive; 2) suffering from convergence related challenges; 3) not always able to uncover the optimal Pareto front [17].

Bayesian optimization (BO) [42] is a machine learning (ML) based framework that has the potential to overcome the drawbacks of GA, especially in reducing the number of expensive design simulations to discover (approximate) optimal Pareto solutions [18]. While ML models are typically trained with previously generated data, BO instead builds statistical surrogate models throughout the optimization process, using the knowledge of prior design evaluations to improve these models continuously. These models are employed to intelligently select the sequence of designs for evaluation by maximizing a utility function defined in terms of the learned statistical model's prediction and uncertainty. Optimizing the utility function is a cheaper alternative because evaluating the surrogate models is often less time-consuming than evaluating the physical models. Details of several ML terminologies are thoroughly described in Sections II and IV.

Engineers face a myriad of physical constraints that prohibit the realization of a design. An optimization algorithm that does not address a highly constrained design will perform poorly. For example, device thermal limits set an upper bound to power flow. High-power systems thus require larger and more costly devices and cooling to avoid these thermal limits. If minimizing system weight and cost are design objectives, then optimization without constraints would select lightweight and cheap converters that could not handle the required power demand. Therefore, adding constraints to the MOO is essential. There is a large body of literature on single-objective BO algorithms with and without constraints [9]-[11][18][19][43][44]. However, there is less work on the more challenging problem of BO for *multiple* objectives [6][8][20][21][45][46] and only one prior work addressing *constrained* MOO problems, named PESMOC [22]. PESMOC is an information-theoretic approach that relies on the principle of input space entropy search. However, it is computationally expensive to optimize the acquisition function behind PESMOC. A series of approximations are performed to improve the efficiency potentially at the expense of accuracy. Additionally, PESMOC addresses the constraints satisfaction only in the design of the acquisition function, while this paper addresses the constraints satisfaction in both acquisition function design and optimization.

In electrical engineering, BO algorithms have been used to optimize a converter level design of a multi-output switched-capacitor voltage regulator and achieved a 90% reduction in the number of simulations required to optimize design parameters [23]. However, there has been little research in developing power electronic system-subsystemoriented design automation tools using ML-based optimization.

C. Major Contributions

In this paper, a novel ML-based optimization algorithm is proposed, namely **M**ax-value **E**ntropy **S**earch for **M**ulti-objective **O**ptimization with **C**onstraints (MESMOC), capable of searching through a constraint-heavy design space to efficiently discover Pareto optimal designs. This paper also serves as the first-of-its-kind that such an ML-based multi-objective BO under constraints is successfully demonstrated for an ETPS design. Experiments, although not of ETPS design, using a prior version of the ML algorithm without constraints, i.e., MESMO, consistently outperform state-of-the-art algorithms at providing an accurate, computationally-efficient, and robust optimization solver [6]. This work builds upon MESMO and extends it to handle black-box constraints that cannot be evaluated without expensive design simulations, an open problem in the BO literature, and provides a successful demonstration on an ETPS design application. While the entire optimization procedure is called MESMOC, it specifically refers to the acquisition function used in the more general BO procedure. The key idea is to build statistical models of both design objectives and constraints and use these models to select the sequence of designs for evaluation based on the information-theoretic principle of output space entropy search: maximize the information gain about the optimal *constrained* Pareto front. This ML algorithm enables automating the system-level design process by intelligently searching through large design spaces.

The ML algorithm must be applied to a physical model, i.e., the domain knowledge. For this paper, a static or averaged model is developed for each component in the power system, including multiple physical domains. Plenty of literature exists on the development and experimental validation of ETPS physical models, hence not a focus of this paper. For example, a multi-timescale parametric electrical battery model is described in [24], and [2][25][26] demonstrate the integration of multiple subsystems for ETPS. Once the physical models are combined to form the desired system architecture, MESMOC can treat the simulation as a black-box function where the outputs are optimization objectives and constraint evaluations, and the inputs are design parameters. MESMOC then evaluates the input design parameters which maximize the information gain about the optimal Pareto front in each iteration until an optimal Pareto set is found.

Without loss of generality, this paper details a case design of an all-electric vertical-takeoff-landing (eVTOL) heavy-duty, freight carrying, urban aerial vehicle (UAV) power system. However, the proposed ML-based power system design framework is generic and can be used for various complex applications, such as more electric aircraft, on/off-road vehicles, ships, green buildings, renewable energy systems, etc. Using a UAV system, this paper effectively demonstrates the drastic reduction of the number of simulation iterations towards converging to Pareto optimal designs. Experimental results demonstrate MESMOC's consistent performance over GA and PESMOC, in both Pareto front quality and convergence rate, where in one trial the optimal Pareto front is discovered by MESMOC after exploring only ~4% of the design space.

D. Paper Organization

For the rest of the paper, an overview of the ML-enabled design optimization process is presented in Section II. The details of the ETPS modeling in Section III will help comprehension of the proposed ML framework. Then a complete description of the MESMOC algorithm is in Section IV. Section V will discuss the practical issues when integrating physical models with the ML algorithm. The experimental results of the Pareto front search using MESMOC, accompanied by a performance comparison to other popular algorithms in the literature, will be presented in Section VI. Section VII concludes the paper and recommends future work.

II. High-Level ETPS Design Process Using ML

An overview of the proposed design process is presented in Fig. 1, which will be referred to throughout this section. The design process begins by constructing subsystem models of ETPS to be used in a mission-based multi-physics power system simulation. This physical modeling part is reflected in the upper box "multi-physics power system simulator" in Fig. 1. A mixed-fidelity modeling (e.g., static, dynamic, quasi-dynamic) approach may be chosen, depending on the design objectives and level of details. Regardless, the ML algorithm suits a variety of modeling methods, since they are treated as black-box functions. The power system simulator outputs the design objectives and constraints based on a set of selected design parameters $\boldsymbol{x}$ and a mission profile discussed in Section III. Given the simulator outputs, the optimization problem solved by the ML algorithm can be generically defined as

$$\begin{aligned} &\min_{\boldsymbol{x}} f_1(\boldsymbol{x}) \\ &\min_{\boldsymbol{x}} f_2(\boldsymbol{x}) \\ &\quad\vdots \\ &\min_{\boldsymbol{x}} f_K(\boldsymbol{x}) \\ s.t. \quad & c_1(\boldsymbol{x}) \leq 0 \\ & c_2(\boldsymbol{x}) \leq 0 \\ &\quad\vdots \\ & c_L(\boldsymbol{x}) \leq 0 \end{aligned} \tag{1}$$

where there are K minimization objective functions $f_{(\cdot)}(\boldsymbol{x})$ and L inequality constraint functions $c_{(\cdot)}(\boldsymbol{x})$ in the negative null form. Equality constraints are omitted from (1) for simplicity.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

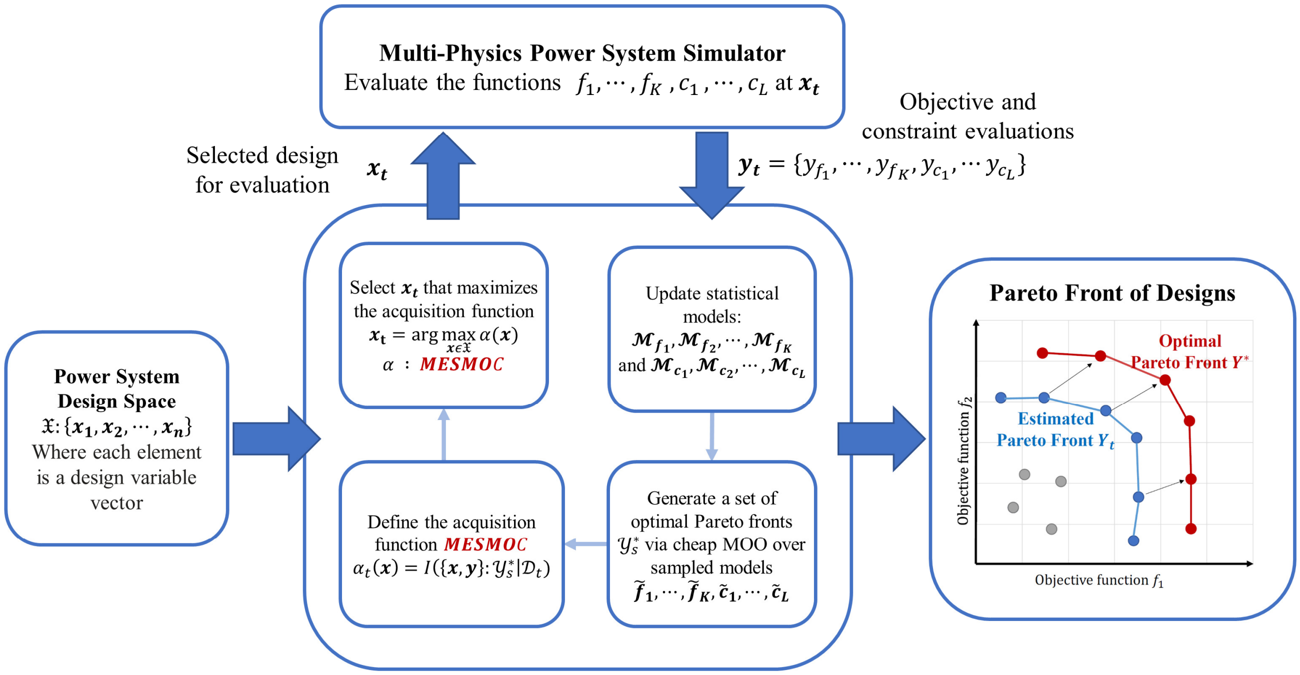


Fig. 1. An overview of the proposed ML design process for an ETPS.

Next comes the design parameter selection and evaluation process. A design parameter vector $\boldsymbol{x_t}$, where t denotes the iteration number, represents the set of parameters used in each power subsystem model, such as battery pack voltage and capacity, and motor quantity and size. The pre-defined design space $\mathfrak{X}$, is the set of all possible design parameter vectors $\mathfrak{X}: \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}\}$, where n is the design space size, which means all possible combinations of individual physical parameter choices. The vital role of MESMOC is to intelligently select the proper $\boldsymbol{x_t}$ for the next iteration without the need to go through all n design vectors, which will be discussed in the next two paragraphs and in detail in Section IV.

Referring to the upper side of Fig. 1, assuming an $\boldsymbol{x_t}$ is chosen, then the power system simulation yields an output vector, $\boldsymbol{y_t} = \boldsymbol{F}(\boldsymbol{x_t})$, where $\boldsymbol{F}$ is the black-box function defined by the physical models. To be specific, given K design objectives and L constraints that the system must satisfy, the power system simulation output is $\boldsymbol{y_t} = \{y_{f_1}, \dots, y_{f_K}, y_{c_1}, \dots, y_{c_L}\}$. An objective function f_k ($k \in \{1, \cdots, K\}$) can be any design evaluation metric to be optimized, such as energy consumption, system weight, cost, or reliability. Likewise, a constraint function c_l ($l \in \{1, \cdots, L\}$) is a metric that limits the physical realization, such as component temperatures, battery state, or spatial limits. The latest input $\boldsymbol{x_t}$ and output $\boldsymbol{y_t}$ join a pool of all previous evaluated inputs and outputs $\mathcal{D}_t$ (i.e., $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{\boldsymbol{x_t}, \boldsymbol{y_t}\}$). This entire pool of data, known as prior experimental data, is used by the ML algorithm to intelligently select new design parameters $\boldsymbol{x_t}$ for the next iteration.

The proposed MESMOC utilizes statistical models, represented as $\mathcal{M}_{f_k}$ and $\mathcal{M}_{c_l}$, to learn the true mapping function from the input parameters to the design objectives and constraints, respectively. The statistical models provide Gaussian probability distributions, based on the prior information, of the objective and constraint values for each design candidate in $\mathfrak{X}$. Approximations of the objectives ($\tilde{f}_1, \dots, \tilde{f}_K$) and constraints ($\tilde{c}_1, \dots, \tilde{c}_L$) are generated by sampling from these distributions via random Fourier features [27]. Each objective and constraint is often approximated multiple times to create a set of sampled functions. Further details of function sampling will be presented in Section IV.

Table I. Nomenclature reference for the ML optimization process.

Symbol	***Description***
K	Number of objectives
k	$\in \{1, \cdots, K\}$, indexing term
L	Number of constraints
l	$\in \{1, \cdots, L\}$, indexing term
$\mathfrak{X}$	Design space
$\mathcal{X}^*$	Optimal Pareto set
$\mathcal{Y}^*$	Optimal Pareto front
$\boldsymbol{x}$	$\in \mathfrak{X}$, design parameter vector
$\boldsymbol{y}$	Power system simulator output vector
$f_k(\boldsymbol{x})$	Objective function
$c_l(\boldsymbol{x})$	Constraint function
$\mathcal{M}$	Surrogate statistical model
$\mathcal{D}$	Training data set
$\alpha(\boldsymbol{x})$	Acquisition function

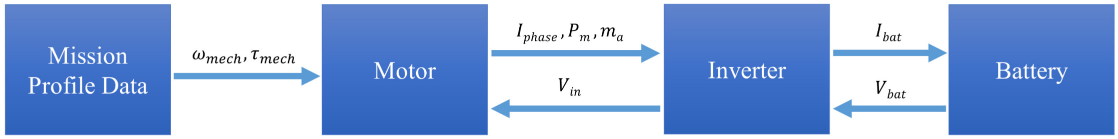


Fig. 2. Block diagram of UAV power system showing the interfaces between each component model [28].

With these sampled functions, a set of *approximate* Pareto fronts $\mathcal{Y}_s^*$ are generated by solving multiple *cheap* and *fast* MOO problems with a GA based algorithm. This procedure is considered cheap because the sampled functions provide quick design evaluations compared to the power system simulation. The approximated Pareto fronts $\mathcal{Y}_s^*$ and previously evaluated designs $\mathcal{D}_t$ are then used to construct the acquisition function $\alpha(\boldsymbol{x})$, which infers the potential information gain about the *optimal* constrained Pareto front $\mathcal{Y}^*$ for a given design $\boldsymbol{x}$. MESMOC utilizes a max-value entropy search-based acquisition function, which is covered in more detail in Section IV. The next simulated design $\boldsymbol{x}_t$ is selected by maximizing $\alpha(\boldsymbol{x})$ that ensures the maximum information gain about the optimal constrained Pareto front $\mathcal{Y}^*$ is achieved in the next iteration.

Then this $\boldsymbol{x}_t$ feeds to the next iteration of the multi-physics power system simulator as discussed previously. After the design $\boldsymbol{x}_t$ is evaluated, the results $\boldsymbol{y}_t$ are used to update statistical models $\mathcal{M}_{f_k}$ and $\mathcal{M}_{c_l}$. The process is then repeated. Throughout the optimization process, the estimated Pareto front is continuously updated according to the new information and will approach the optimal constrained Pareto front $\mathcal{Y}^*$. After a specified number of iterations, the final Pareto set is ready to review. Table I summarizes a nomenclature reference.

III. Electrical Power System Modeling

This section serves to understand the physical domain of the ML design automation and optimization framework. As this paper intends to explore the use of ML to reduce the number of iterations for ETPS optimization, technical details and validations of the ETPS models will not be the focus but are discussed in separate literature [2][25][26]. In fact, the type of power system modeling is of minor importance as the optimization algorithm treats it as a black-box function. Still, some level of the physical understanding allows to comprehend the application of the ML algorithm better. In this paper, a time-based quasi-static simulation capturing averaged power calculations is used. While the quasi-static simulation does not capture the full dynamics of the system, the accuracy and usefulness of the approach for UAV power system design has been demonstrated in [25].

A. System Overview

The UAV system architecture consists of a central Li-ion battery pack, hex-bridge DC-AC inverters, permanent magnet synchronous machines (PMSM), and necessary wiring, as shown in Fig. 2. A set of variable design parameters, such as the battery pack configuration and motor size, are included in the system models. This set, known as the design space, will be

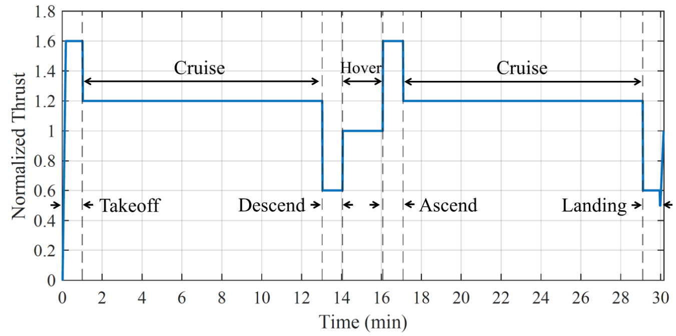


Fig. 3. Mission-profile used for UAV simulations [28].

searched by the ML algorithm to find the optimal designs. The sweeping ranges of the design parameters will be discussed in Section VI.

The mass of the aircraft frame and its cargo is held constant for all designs. The total craft mass m_{UAV} varies depending on the number of cells in the battery pack, motor sizing, and the number of motors. Note that power electronics mass is assumed constant for this study since the semiconductor weight variation is relatively small. Other design details may be included, such as heat sinks or filters. However, this paper focuses on the development of the ML-physical integrated framework rather than a high-fidelity model.

The system follows a pre-configured mission-profile during a simulation, such as the 30-minute mission shown in Fig. 3 that represents a roundtrip flight. The mission-profile is structured as a normalized thrust vs. time array with increments of 1 second in the case of static modeling. Thrust values are scaled by the UAV's total mass and converted into propeller mechanical speed ω_{mech} and torque τ_{mech}, which serve as inputs to the motor model. Thrust to speed and torque conversion is achieved in the same manner as [25], where a propeller-dependent relation is developed. The required motor power characteristics to achieve this mission profile are back-propagated through the motor drive to the battery. Total thrust output power and system power loss are used to determine the total energy consumed throughout the flight, denoted as E_{total}.

B. Motor

The motors considered in this design illustration are fixed-phase fixed-pole (e.g., 3ϕ 8-pole) PMSMs. A chosen reference motor provides initial electrical and mechanical parameters. All potential motor designs are scaled from this reference using two design parameters: 1) number of coil winding turns N_t, and 2) height of the stator structure h_s. These two physical parameters are selected because of their influence over the electrical parameters. For example, the formula for a single coil's inductance,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

$$L_{coil} = \frac{\mu N_t^2}{l_{coil}} \cdot \pi \left(\frac{h_s}{2}\right)^2 \quad (2)$$

is dependent on N_t and h_s, where l_{coil} is the length of the coil, governing motor diameter, and μ is the magnetic permeability constant. For a given set of N_t and h_s, the thickest wire gauge in AWG for the stator coil is selected while still satisfying the winding fill factor limit. In general, motor design is accomplished by perturbing reference values of stator resistance, synchronous inductance, back EMF constant, and mass using these two design parameters. The model utilizes the per-phase equivalent circuit of a PMSM motor for all electrical calculations.

The inputs and outputs of the motor subsystem is given in

$$[I_{phase}, m_a, P_m, P_{m,loss}, T_{m,n+1}] = Motor(\omega_{mech}, \tau_{mech}, V_{in}, T_{m,n} \mid N_m, N_t, h_s) \quad (3)$$

where the inputs are mechanical speed ω_{mech}, mechanical torque τ_{mech}, input voltage from the inverter V_{in}, and motor winding temperature $T_{m,n}$, all under various combinations of N_m number of motors, N_t, h_s. Not all the necessary math will be covered in this paper; however, an overview is provided to aid comprehension. For a detailed derivation of this model, see [25][26].

For the model outputs, the RMS phase current can be found from the mechanical output power and loss using

$$I_{phase} = \frac{\omega_{mech}\tau_{mech} + P_{m,loss}}{3V_{EMF}} \quad (4)$$

The modulation index m_a is given in

$$m_a = \sqrt{6}V_t/V_{in} \quad (5)$$

but requires knowledge of the terminal voltage V_t given the back EMF voltage V_{EMF} and voltage drop due to motor impedance. While m_a is calculated in the motor model, it is utilized in the inverter subsystem, to be described in the following subsection. P_m represents the input power to the motor and is found with (6), where $P_{m,loss}$, in (7), is the combined mechanical and electrical losses.

$$P_m = P_{out} + P_{m,loss} \quad (6)$$

$$P_{m,loss} = 3R_s I_{phase}^2 + P_{mech,loss} \quad (7)$$

The differential equation of the motor winding's temperature gradient is represented by

$$m_m c_p \frac{dT_{m,n+1}}{dt} = P_{m,loss} + h_{air} A_m (T_a - T_{m,n+1}) \quad (8)$$

where $m_m c_p$ is the motor's thermal capacitance, h_{air} is the heat transfer coefficient based on the Nusselt number, A_m is the surface area of the motor coils, T_a is the ambient temperature, and $T_{m,n+1}$ is the updated motor temperature. As this is a static model, only the steady-state temperature is required. Therefore, (8) can be reduced to

$$T_{m,n+1} = T_a + \frac{P_{m,loss}}{A_m \cdot h_{air}} \quad (9)$$

C. *Motor Drive*

The motor drive is a 3ϕ hex-bridge DC-AC inverter consisting of MOSFET/diode pairs. For this given topology and use of static modeling, two practical design parameters are the MOSFET/diode selection M_{sw}, and the switching frequency f_{sw}. Inverter modeling is based on [25], which uses an averaged switching approach to calculate inverter losses and basic control requirements. Similar to the motor subsystem, (10) shows the inputs and outputs of the inverter subsystem, where V_{DC} is the DC bus voltage, I_{phase} is a phase RMS current of the motor stator, m_a is the amplitude modulation index of the SPWM switching scheme, P_m is the motor input power from (6), and $T_{i,n}$ is the temperature of the MOSFET/diode.

$$[I_{bat}, P_{i,loss}, T_{i,n+1}] = Inverter(V_{DC}, I_{phase}, m_a, P_m, T_{i,n} \mid f_{sw}, M_{sw}) \quad (10)$$

In order to determine the left-hand-side outputs in (10), some intermediate loss calculations are required. The conduction loss of a single MOSFET is found with

$$P_{c,m} = 2 \cdot I_{phase}^2 \cdot R_{on} \cdot \left[\frac{1}{8} + \frac{m_a \cos\phi}{3\pi}\right] \quad (11)$$

which averages the time-varying duty cycle [29] to approximate the losses with a drain-source on-resistance R_{on} and I_{phase}. R_{on} comes from the selected MOSFET datasheet where curve-fitting is used to adjust the on-resistance given at $T_{i,n}$. The MOSFET switching loss is given in

$$P_{sw,m} = f_{sw} \cdot (E_{on} + E_{off}) \quad (12)$$

which is broken up into a turn-on energy (13) and turn-off energy (14) [30]. Approximating E_{on} and E_{off} requires the reverse recovery charge Q_{rr} and rising/falling current and voltage times obtained from the datasheet and [31].

$$E_{on} = V_{DC} \cdot \frac{\sqrt{2}I_{phase}}{\pi} \cdot \frac{t_{ri}+t_{fu}}{2} + Q_{rr} \cdot V_{DC} \quad (13)$$

$$E_{off} = V_{DC} \cdot \frac{\sqrt{2}I_{phase}}{\pi} \cdot \frac{t_{ru}+t_{fi}}{2} \quad (14)$$

Similar to the MOSFET conduction loss, diode conduction loss is given in (15) but uses the forward voltage drop rather than the on-resistance. Q_{rr} of the diode is required to solve for the switching power loss using (16) and can be found on the datasheet. It is worth noting these approximate loss calculations result in worst case scenario values.

$$P_{c,d} = V_{on} \cdot \sqrt{2}I_{phase} \cdot \left[\frac{1}{2\pi} - \frac{m_a \cos\phi}{8}\right] \quad (15)$$

$$P_{sw,d} = \frac{Q_{rr} \cdot V_{DC} \cdot f_{sw}}{4} \quad (16)$$

Total power loss $P_{i,loss}$ of the inverter is found by summing up (11)-(16) and scaled with the number of switches. The required input current from the battery I_{bat} is derived from the total power into the inverter and V_{DC}.

An aluminum heatsink is assumed to attach all switching devices. Given a heatsink thermal capacitance $m_s c_p$, the differential equation for the heatsink temperature T_s gradient is

$$m_s\, c_p \frac{dT_s}{dt} = \frac{T_j - T_s}{R_{th,js}} + \frac{T_a - T_s}{R_{th,sa}} \quad (17)$$

where T_j is the MOSFET/diode junction temperatures, T_a is the ambient temperature. $R_{th,js}$ and $R_{th,sa}$ are the semiconductor junction to heatsink and heatsink to ambient air thermal resistances, respectively. Note that T_j is equivalent to the inverter temperature $T_{i,n}$ in (10). As this model assumes steady-state behavior, thermal capacitances are ignored, and (17) can be simplified to (18) for a one-second interval. Equation (19) is then used to find the semiconductor junction temperatures using $P_{i,loss}$ and junction to sink thermal resistance.

$$T_{s,n+1} = \frac{T_{j,n}R_{th,sa} + T_a R_{th,js}}{R_{th,js}+R_{th,sa}} \tag{18}$$

$$T_{j,n+1} = T_{s,n+1} + P_{i,loss} \cdot R_{th,js} \tag{19}$$

D. Battery

The Li-ion battery pack configuration consists of three design parameters: 1) number of cells in series N_s, 2) number of cells in parallel N_p, and 3) the battery cell model dataset, M_{bat}. The number of cells in series determines the battery pack voltage, whereas the number of cells in parallel indicates the pack's Ah capacity. As individual battery cell current can be determined using N_p, it is only necessary to model a single battery cell. It is assumed that each cell is identical and discharges at the same rate.

Battery cell modeling follows the work [24], where a Randle's equivalent circuit is used. Battery constants of the equivalent circuit such as the open-circuit voltage V_{oc}, and the RC-network's resistance $R_{(\cdot)}$ and capacitance $C_{(\cdot)}$ are parametrically computed as a function of the state of charge (SOC). Using experimentally gathered parameters (denoted by a_k), V_{oc} is calculated by

$$\ln(V_{oc}) = a_0 + a_1 \ln^1(SOC) + \cdots + a_6 \ln^6(SOC) \tag{20}$$

or

$$V_{oc} = \exp\left[\sum_{k=0}^{6} a_k \ln^k(SOC)\right] \tag{21}$$

Resistances $R_{(\cdot)}$ and capacitances $C_{(\cdot)}$, one for each time-constant RC-network, is calculated the same way as V_{oc} using (21) but with a different set of a_k parameters. In [24], the a_k parameters are found over different time scales of seconds, minutes, and hours. However, as this paper focuses on static modeling, capacitive elements can be ignored, allowing simplification of resistances into a single constant R_{int}.

A function notation of the battery pack model is stated as

$$\begin{gathered}[V_{bat}, SOC_{n+1}, P_{b,loss}] = \\ Battery(SOC_n, I_{bat} \mid N_s, N_p, M_{bat})\end{gathered} \tag{22}$$

showing the subsystem interface. The three outputs are calculated based on the present SOC_n, current demand I_{bat}, and the battery pack design parameters. With known V_{oc} and R_{int} and single-cell current demand of the system I_{cell}, battery terminal voltage V_{bat} and power loss $P_{b,loss}$ are found using Ohm's law and I^2R losses, respectively. After every time step of the simulation, the SOC_{n+1} is adjusted given the energy consumption during the last interval. Battery temperature change is not considered here.

IV. MESMOC: Max-value Entropy Search for Multi-objective Optimization with Constraints

In this section, we first provide the ML theory background and formal problem setup. Subsequently, we describe the novel ML-based optimization algorithm referred to as MESMOC to accelerate design automation of ETPS by saving a significant amount of resources.

A. ML Background and Problem Setup

Bayesian Optimization Framework is a very efficient framework to solve global optimization problems using *black-box evaluations of expensive objective functions*. Let $\mathfrak{X} \subseteq \mathfrak{R}^d$ be an input design space with *d* design parameters. In a single-objective problem setting, we are given an unknown real-valued objective function $f: \mathfrak{X} \mapsto \mathfrak{R}$, which can evaluate each design $\boldsymbol{x} \in \mathfrak{X}$ to produce an evaluation $y = f(\boldsymbol{x})$. Each evaluation $f(\boldsymbol{x})$ is expensive in terms of the consumed resources. For example, in the case of the ETPS design problem, there involves performing an expensive simulation to evaluate the design quality. The main goal is to find a design $\boldsymbol{x}^* \in \mathfrak{X}$ that approximately optimizes f by performing a limited number of function evaluations. BO algorithms learn a *cheap* surrogate statistical model from the training data obtained from past function evaluations. This statistical model can predict the design output that is not evaluated yet and can quantify uncertainty in prediction. The prediction/uncertainty estimates from this model are used to intelligently select the next input design for evaluation by trading-off exploration (selecting designs with high uncertainties) and exploitation (selecting designs with better predicted objective values) to quickly direct the search towards optimal inputs. The three key elements of a BO framework are as follows:

1) Statistical Model of the true function $f(\boldsymbol{x})$. *Gaussian Process (GP)* [32] is the most commonly used model. A GP over a space $\mathfrak{X}$ is a random process from $\mathfrak{X}$ to $\mathfrak{R}$. It is characterized by a mean function $\mu: \mathfrak{X} \mapsto \mathfrak{R}$ and a covariance or kernel function $\kappa: \mathfrak{X} \times \mathfrak{X} \mapsto \mathfrak{R}$. If a function f is sampled from GP(μ, κ), then $f(\boldsymbol{x})$ is distributed normally $\mathcal{N}(\mu(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}))$ for a finite set of inputs from $\boldsymbol{x} \in \mathfrak{X}$.

2) Acquisition Function (α) to score the utility of evaluating a candidate input $\mathbf{x} \in \mathfrak{X}$ based on the statistical model. Some popular acquisition functions in the single-objective literature include expected improvement (EI), upper confidence bound (UCB), predictive entropy search (PES), and max-value entropy search (MES) [33].

3) Optimization Procedure to select the best scoring candidate input according to α, depending on the statistical model. DIRECT [34] is a very popular approach for acquisition function optimization.

Problem Setup of Multi-Objective Optimization with Constraints: Our goal is to minimize real-valued objective functions $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \cdots, f_K(\boldsymbol{x})$, with $K \geq 2$, while satisfying L black-box constraints of the form $c_1(\boldsymbol{x}) \geq 0, c_2(\boldsymbol{x}) \geq 0, \cdots, c_L(\boldsymbol{x}) \geq 0$ over continuous space $\mathfrak{X} \subseteq \mathfrak{R}^d$ (e.g., battery cell voltage, motor winding temperature, etc.). Each evaluation of an input $\boldsymbol{x} \in \mathfrak{X}$ produces a vector of objective values and constraint values $\boldsymbol{y} = (y_{f_1}, \cdots, y_{f_K}, y_{c_1}, \cdots, y_{c_L})$ where $y_{f_k} =$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

$f_k(\boldsymbol{x})$ for all $k \in \{1,2,\cdots,K\}$ and $y_{c_l} = c_l(\boldsymbol{x})$ for all $l \in \{1,2,\cdots,L\}$. We say that a valid design $\boldsymbol{x}$, satisfying all constraints, *Pareto-dominates* another design $\boldsymbol{x}'$ if $f_k(\boldsymbol{x}) \leq f_k(\boldsymbol{x}'), \forall k$ and there exists some $k \in \{1,2,\cdots,K\}$ such that $f_k(\boldsymbol{x}) < f_k(\boldsymbol{x}')$. The optimal solution of MOO problem with constraints is a set of designs $\mathcal{X}^* \subset \mathfrak{X}$ such that no design $\boldsymbol{x}' \in \mathfrak{X}\backslash\mathcal{X}^*$ Pareto-dominates a design $\boldsymbol{x} \in \mathcal{X}^*$ and all designs in $\mathcal{X}^*$ satisfy the L problem constraints. The solution set $\mathcal{X}^*$ is called the optimal *Pareto set*, and the corresponding set of function values $\mathcal{Y}^*$ is called the optimal *Pareto front*. Our goal is to approximate $\mathcal{X}^*$ by minimizing the number of function evaluations.

B. *MESMOC for Multi-Objective Optimization with Constraints*

In this subsection, we first provide an overview of the proposed MESMOC algorithm, then explain the technical details of the algorithm by mathematically describing the output space entropy-based acquisition function.

Overview of MESMOC algorithm. We propose an acquisition function referred to as MESMOC that computes the information gain of a new design $\boldsymbol{x}$ with respect to the optimal Pareto front $\mathcal{Y}^*$. In each iteration of MESMOC, we select the design that provides the maximum information gain for evaluation. The algorithm has four major steps as depicted by Fig. 1.

Step 1: Learning surrogate models (Posterior estimation). Gaussian processes are shown to be effective surrogate statistical models. We model the objective functions and black-box constraints by independent GP models $\mathcal{M}_{f_1}, \mathcal{M}_{f_2}, \cdots, \mathcal{M}_{f_K}$ and $\mathcal{M}_{c_1}, \mathcal{M}_{c_2}, \cdots, \mathcal{M}_{c_L}$ with a zero mean and i.i.d. (independent and identically distributed) observation noise. Let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{t-1}$ be the training data from the past $t-1$ iterations, where $\boldsymbol{x}_i \in \mathfrak{X}$ is a design and $\boldsymbol{y}_i = \{y_{f_1}^i, \cdots, y_{f_K}^i, y_{c_1}^i, \cdots y_{c_L}^i\}$ is the output vector resulting from evaluating the objective functions and constraints at $\boldsymbol{x}_i$. We learn surrogate models from $\mathcal{D}$.

Step 2: Pareto front sampling. We sample black-box functions and constraints from the model. We employ the sampled functions to generate a set of sampled Pareto fronts via a *cheap* constrained multi-objective optimization solver.

Step 3: Acquisition function optimization. We optimize the acquisition function based on the sampled pareto fronts to select the design with the highest information gain while satisfying the constraints for the next evaluation.

Step 4: Design evaluation: we evaluate the objective functions and constraints for the selected design, and add them to the aggregate training data to update the statistical models based on this new information.

Output space entropy-based acquisition function. Input space entropy-based methods such as PESMO and PESMOC [21][22] select the next candidate input $\boldsymbol{x}_t$ by maximizing the information gain about the optimal Pareto set $\mathcal{X}^*$. For ease of notation, we drop the subscript in below discussion. The acquisition function based on input space entropy is given as

$$\begin{aligned} \alpha(\boldsymbol{x}) &= I(\{\boldsymbol{x},\boldsymbol{y}\}, \mathcal{X}^* \mid \mathcal{D}) \\ &= H(\mathcal{X}^* \mid \mathcal{D}) - \mathbb{E}_y[H(\mathcal{X}^* \mid \mathcal{D} \cup \{\boldsymbol{x},\boldsymbol{y}\})] \\ &= H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}) - \mathbb{E}_{\mathcal{X}^*}[H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{X}^*)] \end{aligned} \quad (23)$$

Information gain is defined as the expected reduction in entropy $H(\cdot)$ of the posterior distribution $P(\mathcal{X}^* \mid \mathcal{D})$ over the optimal Pareto set $\mathcal{X}^*$ as given in (23). This mathematical formulation relies on a very expensive and high-dimensional ($m \cdot d$ dimensions) distribution $P(\mathcal{X}^* \mid \mathcal{D})$, where m is the size of the optimal Pareto set $\mathcal{X}^*$, and d is the dimension of the input space (number of design parameters). Furthermore, optimizing the expected entropy, $\mathbb{E}_{\mathcal{X}^*}[H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{X}^*)]$, in (23) poses significant challenges: a) a series of approximations are required, which can be potentially sub-optimal [21]; and b) optimization, even after approximations, is expensive; and c) performance is strongly dependent on the number of Monte-Carlo samples.

To overcome the above challenges of computing the input space entropy-based acquisition function, [6] proposed to maximize the information gain about the optimal **Pareto front** $\mathcal{Y}^*$. However, the MESMO algorithm in [6] did not address the challenge of a *constrained* Pareto front. We propose an extension of MESMO's acquisition function to maximize the information gain between the next candidate input for evaluation $\boldsymbol{x}$ and the optimal constrained Pareto front $\mathcal{Y}^*$, given as

$$\begin{aligned} \alpha(\mathbf{x}) &= I(\{\boldsymbol{x},\boldsymbol{y}\}, \mathcal{Y}^* \mid \mathcal{D}) \\ &= H(\mathcal{Y}^* \mid \mathcal{D}) - \mathbb{E}_y[H(\mathcal{Y}^* \mid \mathcal{D} \cup \{\boldsymbol{x},\boldsymbol{y}\})] \\ &= H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}) - \mathbb{E}_{\mathcal{Y}^*}[H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{Y}^*)] \end{aligned} \quad (24)$$

In this case, the output vector $\boldsymbol{y}$ is $K+L$ dimensional: $\boldsymbol{y} = (y_{f_1}, y_{f_2}, \cdots, y_{f_K}, y_{c_1} \cdots y_{c_L})$ where $y_{f_k} = f_k(\boldsymbol{x})$ for all $k \in \{1,2,\cdots,K\}$ and $y_{c_l} = c_l(\boldsymbol{x})$ for all $l \in \{1,2,\cdots,L\}$. Consequently, the first term in the r.h.s. of (24), entropy of a factorizable $(K+L)$-dimensional Gaussian distribution $P(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x})$, can be computed in a closed form as

$$H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}) = \frac{(K+L)(1+\ln(2\pi))}{2} + \sum_{k=1}^{K} \ln(\sigma_{f_k}(\boldsymbol{x})) + \sum_{l=1}^{L} \ln(\sigma_{c_l}(\boldsymbol{x})) \quad (25)$$

where $\sigma_{f_k}^2(\boldsymbol{x})$ and $\sigma_{c_l}^2(\boldsymbol{x})$ are the predictive variances of k^{th} function and l^{th} constraint GPs at input $\boldsymbol{x}$, respectively. The second term in the r.h.s. of (24) is an expectation over the Pareto front $\mathcal{Y}^*$. We can approximately compute this term via Monte-Carlo sampling as

$$\mathbb{E}_{\mathcal{Y}^*}[H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{Y}^*)] \simeq \frac{1}{S}\sum_{s=1}^{S}[H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{Y}_s^*)] \quad (26)$$

where S is the number of samples, and $\mathcal{Y}_s^*$ denotes a sample Pareto front. The main advantages of our acquisition function are computational efficiency and robustness to the number of samples [6].

There are two key algorithmic steps to compute (26): 1) compute Pareto front samples $\mathcal{Y}_s^*$; and 2) compute the entropy with respect to a given Pareto front sample $\mathcal{Y}_s^*$.

1) Computing Pareto front samples via cheap multi-objective optimization. To compute a Pareto front sample $\mathcal{Y}_s^*$, we first sample functions and constraints from the posterior GP

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

models via random Fourier features [31] and then solve a cheap multi-objective optimization over the K sampled functions and L sampled constraints.

Cheap MO solver: We sample $\tilde{f}_k$ from GP model $\mathcal{M}_{f_k}$ for each of the K functions and $\tilde{c}_l$ from GP model $\mathcal{M}_{c_l}$ for each of the L constraints. A *cheap* constrained multi-objective optimization problem over the K sampled functions $\tilde{f}_1, \tilde{f}_2, \cdots, \tilde{f}_K$ and the L sampled constraints $\tilde{c}_1, \tilde{c}_2, \cdots, \tilde{c}_L$ is solved to compute the sample Pareto front $\mathcal{Y}_s^*$. This cheap multi-objective optimization also allows us to capture the interactions between different objectives while satisfying the constraints. We employ the popular constrained NSGA-II algorithm [35] to solve the constrained MO problem with cheap objective functions noting that other MOO algorithms (GAs, PSO, etc.) can be used for similar effects.

2) Entropy computation with a sample Pareto front. Let $\mathcal{Y}_s^* = \{\mathbf{z}^1, \cdots, \mathbf{z}^m\}$ be the sample Pareto front, where m is the size of the Pareto front and each $\mathbf{z}^i$ is a $(K+L)$-vector evaluated at the K sampled functions and L sampled constraints $\mathbf{z}^i = \{z_{f_1}^i, \cdots, z_{f_K}^i, z_{c_1}^i, \cdots, z_{c_L}^i\}$, and $i \in \{1, \cdots, m\}$. The following inequality holds for each component y_j of the $(K+L)$-vector $\boldsymbol{y} = \{y_{f_1}, \cdots, y_{f_K}, y_{c_1}, \cdots, y_{c_L}\}$ in the entropy term $H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{Y}_s^*)$:

$$y_j \le \max\{z_j^1, \cdots z_j^m\} \quad \forall j \in \{f_1, \cdots, f_K, c_1, \cdots, c_L\} \tag{27}$$

The inequality essentially means that the j^{th} component of $\boldsymbol{y}$ (i.e., y_j) is upper-bounded by a value obtained by taking the maximum of j^{th} components of all m $(K+L)$-vectors in the Pareto front $\mathcal{Y}_s^*$. This inequality has been proven by a contradiction in [21] for $j \in \{f_1, \cdots, f_K\}$. We assume the same for $j \in \{c_1, \cdots, c_L\}$.

By combining the inequality (27) and the fact that each function is modeled as an independent GP, we can model each component y_j as a truncated Gaussian distribution since the distribution of y_j must satisfy $y_j \le \max\{z_j^1, \cdots, z_j^m\}$. Furthermore, a common property of entropy measure allows us to decompose the entropy of a set of independent variables into a sum over entropies of individual variables [36]:

$$\begin{aligned}(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{Y}_s^*) \simeq \textstyle\sum_{k=1}^{K} H\,(y_{f_k} | \mathcal{D}, \boldsymbol{x}, \max\{z_{f_k}^1, \cdots z_{f_k}^m\}) \\ + \textstyle\sum_{l=1}^{L} H\,(y_{c_l} | \mathcal{D}, \boldsymbol{x}, \max\{z_{c_i}^1, \cdots z_{c_i}^m\})\end{aligned} \tag{28}$$

The r.h.s. is a summation over entropies of $(K+L)$-variables $\boldsymbol{y} = \{y_{f_1}, \cdots, y_{f_K}, y_{c_1}, \cdots y_{c_L}\}$. The differential entropy for each y_j is the entropy of a truncated Gaussian distribution [37] and given by

$$H(y_{f_k} | \mathcal{D}, \boldsymbol{x}, y_s^{f_k *}) \simeq \left[\frac{(1+\ln(2\pi))}{2} + \ln(\sigma_{f_k}(\boldsymbol{x})) + \ln\Phi(\gamma_s^{f_k}(\boldsymbol{x})) - \frac{\gamma_s^{f_k}(x)\phi(\gamma_s^{f_k}(x))}{2\Phi(\gamma_s^{f_j}(x))}\right] \tag{29}$$

$$H(y_{c_l} | \mathcal{D}, \boldsymbol{x}, y_s^{c_l *}) \simeq \left[\frac{(1+\ln(2\pi))}{2} + \ln(\sigma_{c_l}(\boldsymbol{x})) + \ln\Phi(\gamma_s^{c_l}(\boldsymbol{x})) - \frac{\gamma_s^{c_l}(x)\phi(\gamma_s^{c_l}(x))}{2\Phi(\gamma_s^{c_l}(x))}\right] \tag{30}$$

Consequently, we have:

$$\begin{aligned}H(\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{x}, \mathcal{Y}_s^*) \simeq \\ \textstyle\sum_{k=1}^{K}\left[\frac{(1+\ln(2\pi))}{2} + \ln(\sigma_{f_k}(\boldsymbol{x})) + \ln\Phi(\gamma_s^{f_k}(\boldsymbol{x})) - \frac{\gamma_s^{f_k}(x)\phi(\gamma_s^{f_k}(x))}{2\Phi(\gamma_s^{f_k}(x))}\right] \\ + \textstyle\sum_{l=1}^{L}\left[\frac{(1+\ln(2\pi))}{2} + \ln(\sigma_{c_l}(\boldsymbol{x})) + \ln\Phi(\gamma_s^{c_l}(\boldsymbol{x})) - \frac{\gamma_s^{c_l}(x)\phi(\gamma_s^{c_l}(x))}{2\Phi(\gamma_s^{c_l}(x))}\right]\end{aligned} \tag{31}$$

where $\gamma_s^{c_l}(\boldsymbol{x}) = \frac{y_s^{c_l *} - \mu_{c_l}(x)}{\sigma_{c_l}(x)}$, $\gamma_s^{f_k}(\boldsymbol{x}) = \frac{y_s^{f_k *} - \mu_{f_k}(x)}{\sigma_{f_k}(x)}$; $y_s^{c_l *}$ and $y_s^{f_k *}$ are the maximum values of constraint $\tilde{c}_l$ and function $\tilde{f}_k$ reached after the cheap multi-objective optimization over sampled functions and constraints: $y_s^{c_l *} = \max\{z_{c_l}^1, \cdots z_{c_l}^m\}$, $y_s^{f_k *} = \max\{z_{f_k}^1, \cdots z_{f_k}^m\}$; ϕ and Φ are the p.d.f. (probability distribution function) and c.d.f. (cumulative distribution function) of a standard normal distribution, respectively. By combining (24), (25), and (31), we obtain the final form of our acquisition function as

$$\begin{aligned}\alpha(\boldsymbol{x}) \simeq \frac{1}{S}\textstyle\sum_{s=1}^{S}\Big[\textstyle\sum_{k=1}^{K} \frac{\gamma_s^{f_k}(x)\phi(\gamma_s^{f_k}(x))}{2\Phi(\gamma_s^{f_k}(x))} - \ln\Phi(\gamma_s^{f_k}(\boldsymbol{x})) \\ + \textstyle\sum_{l=1}^{L} \frac{\gamma_s^{c_l}(x)\phi(\gamma_s^{c_l}(x))}{2\Phi(\gamma_s^{c_l}(x))} - \ln\Phi(\gamma_s^{c_l}(\boldsymbol{x}))\Big]\end{aligned} \tag{32}$$

A complete description of the MESMOC algorithm is summarized in Algorithm 1. The blue colored steps correspond to the computation of our output space entropy-based acquisition function via sampling.

V. ML-Physical Model Integration and System Design Framework

Now that an understanding of the MESMOC ML algorithm and the ETPS models has been established, the integration of the two can be discussed.

A. Practical Issues in Model-ML Integration

The physical modeling approach described in Section III requires a proper interface with the ML algorithm. A major challenge of model-ML integration is how to handle failed design cases, as the failed designs still provide useful learning information although the simulations face non-preferred conditions. In order to maintain consistent results, simulations must continue running regardless of whether the design succeeds the mission under constraints or not. Consistency in results is essential, since MESMOC learns from every simulation. However, simulations are subject to instability when the vehicle is operating at extreme limits. For example, the motor winding temperature is especially susceptible to positive feedback and eventual simulation instability.

Simulation instability can be prevented by suppressing model critical variables, which are often constraint variables (e.g., temperature, voltage, current, etc.), when a specified limit is exceeded. A soft limit approach still allows differentiation between healthy and ill designs by preserving information about the extremity of constraint violations. A hyperbolic function, such as the $tanh$ trigonometric function, realizes this soft limit by asymptotically approaching a value. When a variable's

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

Algorithm 1 MESMOC Algorithm

Input: input space $\mathfrak{X}$; $\boldsymbol{K}$ black-box functions $\boldsymbol{f_1(x), f_2(x), \cdots, f_K(x)}$; $\boldsymbol{L}$ black-box constraints $\boldsymbol{c_1(x), c_2(x), \cdots, c_L(x)}$; and maximum number of iterations $\boldsymbol{T_{max}}$

Initialize Gaussian process models $\boldsymbol{\mathcal{M}_{f_1}, \mathcal{M}_{f_2}, \cdots, \mathcal{M}_{f_K}}$ and $\boldsymbol{\mathcal{M}_{c_1}, \mathcal{M}_{c_2}, \cdots, \mathcal{M}_{c_L}}$ by evaluating at $\boldsymbol{N_0}$ initial points

for each $\boldsymbol{t = N_0 + 1}$ to $\boldsymbol{T_{max}}$ **do**:

Select $\mathbf{x}_t \leftarrow \arg\boldsymbol{max}_{\mathbf{x}\in\mathfrak{X}}\, \boldsymbol{\alpha_t}(\mathbf{x})$

s.t. $(\boldsymbol{\mu_{c_1} \geq 0, \cdots, \mu_{c_L} \geq 0})$

$\boldsymbol{\alpha_t}(.)$ is computed as:

for each sample $\boldsymbol{s \in 1, \cdots, S}$:

Sample $\boldsymbol{\tilde{f}_k \sim \mathcal{M}_{f_k}}$, $\quad \forall \boldsymbol{k} \in \{\boldsymbol{1, \cdots, K}\}$

Sample $\tilde{\mathbf{c}}_l \sim \boldsymbol{\mathcal{M}_{c_l}}$, $\quad \forall \boldsymbol{l} \in \{\boldsymbol{1, \cdots, L}\}$

// Solve *cheap* MOO over $(\boldsymbol{\tilde{f}_1, \cdots, \tilde{f}_K})$ constrained by $(\boldsymbol{\tilde{c}_1, \cdots, \tilde{c}_L})$

$\boldsymbol{\mathcal{Y}_s^*} \leftarrow \arg\boldsymbol{max}_{x\in\mathcal{X}}(\boldsymbol{\tilde{f}_1, \cdots, \tilde{f}_K})$

s.t. $(\boldsymbol{\tilde{c}_1 \geq 0, \cdots, \tilde{c}_L \geq 0})$

Compute $\boldsymbol{\alpha_t}(.)$ based on the $\boldsymbol{S}$ samples of $\boldsymbol{\mathcal{Y}_s^*}$ as given in equation (32)

Evaluate $\boldsymbol{x_t}$; $\mathbf{y}_t \leftarrow (\boldsymbol{f_1(x_t), \cdots, f_K(x_t), c_1(x_t), \cdots, c_L(x_t)})$

Aggregate data: $\boldsymbol{\mathcal{D}} \leftarrow \boldsymbol{\mathcal{D}} \cup \{(\boldsymbol{x_t, y_t})\}$

Update models $\boldsymbol{\mathcal{M}_{f_1}, \mathcal{M}_{f_2}, \cdots, \mathcal{M}_{f_K}}$ and $\boldsymbol{\mathcal{M}_{c_1}, \mathcal{M}_{c_2}, \cdots, \mathcal{M}_{c_L}}$

$\boldsymbol{t \leftarrow t + 1}$ return Pareto front of $\boldsymbol{f_1(x), f_2(x), \cdots, f_K(x)}$ based on $\boldsymbol{\mathcal{D}}$

end for

return Pareto front of $\boldsymbol{f_1(x), f_2(x), \cdots, f_K(x)}$ based on $\boldsymbol{\mathcal{D}}$

constraint is to be violated, a hyperbolic function suppresses the output through variable saturation. For the example of motor winding temperature, applying a soft limit when the temperature exceeds its physical limit reduces further positive feedback by saturating the calculated temperature. This provides more information to the ML algorithm than a hard ceiling limit, where various failed system designs would report the same motor temperature that would be indistinguishable by the ML.

In the case of battery energy depletion, the SOC is reset to the simulation's initial SOC. This is preferred over terminating a simulation at the minimum allowed SOC, as the reported total energy consumed will likely be much lower than a design that completes the mission. A depth of discharge (DOD) variable keeps track of the total amount of battery discharged throughout multiple SOC resets. By resetting the SOC, the total energy consumed will more accurately represent the necessary energy to complete the mission for a specific design.

B. MESMOC Parallel Evaluation

To accelerate the overall design optimization process, multiple instances of MESMOC can be run in parallel. For each individual run, the surrogate statistical models are trained only using the designs selected by that MESMOC instance. To avoid redundant simulations, if a MESMOC instance selects a design which has been previously evaluated by another instance, the results will be shared. This parallel computing approach diversifies the exploration across the runs to quickly uncover the approximate Pareto design sets.

C. MESMOC Scalability

In the unconstrained version of MESMOC, namely MESMO, paper [6] shows that MESMO maintains its performance on problems with up to ten input space dimensions and also evaluates experiments with up to six objective functions. MESMOC is expected to perform the same and will be demonstrated in Section VI. Additionally, there is existing literature for a wrapper over any BO method that allows for scaling to high dimensional input spaces [38]. Very-high-dimensional design space, objectives, and constraints enabled MESMOC or a hierarchical optimization process for ETPS design will be of interest for future work.

D. Software Tools and Interface Requirement

The MESMOC algorithm is developed in Python while the UAV modeling is developed in MATLAB. MESMOC utilizes an open-source BO library called *Spearmint*, and the *PyGMO* library is used for the NSGA-II algorithm in the cheap MO solver. The UAV models are developed with the basic MATLAB setup without special toolboxes. The *matlab.engine* library enables MATLAB function and script calls in the Python environment. After software compatibility verification, MESMOC, given the full parameter ranges of the design space to search through, calls on MATLAB simulation for design evaluations. With the described set of tools and their integration, the process of finding an optimal design is automated.

VI. Experiments & Results

A. Experiment Overview

As discussed in Section I, the MOO is a model-based design methodology. It is used to run numerous simulation-based experiments, as a standard industry practice, especially common in the aviation sector. The design objective here is not to validate dynamic control laws or model accuracy, hence demonstration via hardware-in-the-loop or hardware is beyond the scope. The MOO builds upon accurate models, on the other

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

hand, which are verified elsewhere in literature. Therefore, this section details various simulation-based experiments to showcase the efficacy of the proposed ML-based MOO process.

Case studies must be given concrete design parameters, objectives, and constraints. First, the optimization problem for this eVTOL UAV case design is formally defined by

$$\begin{aligned} &\min_{x} E_{total} \\ &\min_{x} m_{UAV} \\ s.t.\quad & DOD \leq 0.75 \\ & V_{bat} \geq 3.0\,V \\ & T_m \leq 130°C \\ & T_i \leq 125°C \\ & m_a \leq 1.155 \end{aligned} \tag{33}$$

This case study chooses two optimization objectives: minimizing the energy consumption throughout the mission, E_{total}, and minimizing the UAV's total mass, m_{UAV}. The two selected objectives contain mutual tradeoffs, since heavier components, such as motors, tend to be more efficient; however, the extra weight consumes additional energy during flight. These two objectives are common practice in industrial designs; however, users may choose others such as reliability and cost. Five representative constraints serve for demonstration purposes in this paper. In particular, Li-ion batteries have a typical operating range between 20% and 95% SOC, thus a maximum DOD of 75%. A minimum battery cell terminal voltage, V_{bat}, is imposed to prevent cell damages [28]. The maximum motor winding temperature, T_m, limits thermal degradation of wire lamination and is based on the NEMA insulation class B rating [39]. Semiconductor device failure is avoided with a maximum inverter temperature, T_i, that is set marginally lower than the datasheet information [28]. A maximum modulation index, m_a, is set to avoid excessive unwanted distortion that can occur under an SPWM switching scheme [40]. At the end of a simulation, the constraint and objective variables are returned to MESMOC, as described in Fig. 1 in Section II.

On the ML algorithm side, MESMOC is compared to the known constrained GA, namely NSGA-II, and to the BO method PESMOC. We evaluate the performance of MESMOC and the baselines using the Pareto hypervolume (PHV) metric, which is a commonly employed metric to measure the quality of a given Pareto front [41]. The *Platypus* library was used for the NSGA-II trials and configured for the same total number of design evaluations as MESMOC and PESMOC. Given the randomness in the NSGA-II process, which is a known issue, we run the algorithm several times and report the Pareto front of the best performing run (Figs. 4 and 6) and the hypervolume curves of three different runs (Figs. 5 and 7), to be further discussed in Subsections B and C.

Each optimization algorithm is applied to the design process of a UAV power system in two trials to evaluate performance. The first trial approaches the power system design with a five-dimensional design space, while the second trial adds an additional dimension to explore the scalability of the ML algorithm *under the same computing assumption*. The design space for both trials, comprised of each parameter and the range, is summarized in Table II. The bounds on the design space were found through a broad search using the brute force method, similar to Latin hypercube sampling [7]. This method of boundary selection is sufficient for demonstration purpose in this paper, while real-life selections also consider the specific application's requirement, rule of thumb, hardware's absolute limits, and the designer's confidence interval.

Table II. UAV Design Space Ranges for Trial 1 and Trial 2.

Design Parameter	***Range***
Battery cells in series, N_s (#)	[10:18]
Battery cells in parallel, N_p (#)	[16:70]
Quantity of motors, N_m (#)	[6:10]
Height of stator structure, h_s (mm)	[8:26]
Motor stator winding turns, N_t (#)	[10:55]
Inverter switching frequency, f_{sw} (kHz)	[10:40]

B. Trial 1: 5-Dimensional Design Space

The design space for this trial consists of 42,000 design combinations using five parameters: N_s, N_p, N_m, N_t, and h_s, indicated in Table II. Based on the experimental setup discussed in Subsection A and the constrained optimization problem given in (33), brute force, MESMOC, PESMOC, and NSGA-II methods were run. Fig. 4 shows the most meaningful points evaluated by the four algorithms and their respective Pareto fronts. The figure indicates each point as a potential design with respect to the two objective functions, omitting points that do not pass all the constraints. Note that the sporadic empty spaces and discrete boundaries are due to a few factors: 1) the battery parameters, N_s and N_p, cause discrete changes to the objectives; 2) the inherent difference in the energy requirement and vehicle mass between *n*-motored UAVs creates clusters of points within the objective space; and 3) many designs in the design space are not valid due to constraint violations and are thus not shown.

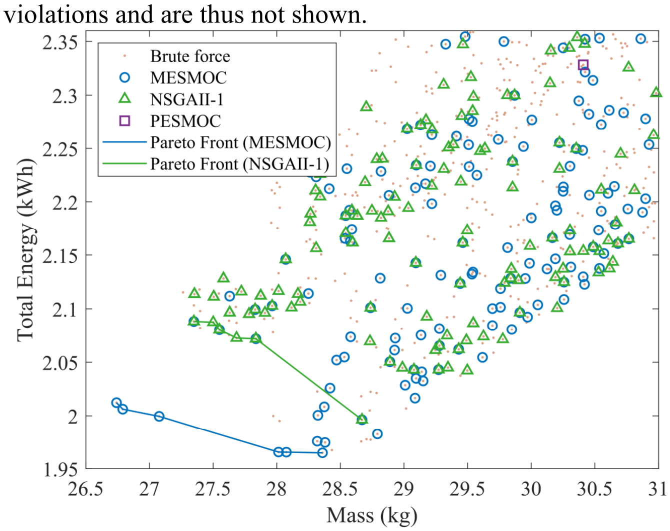


Fig. 4. Trial 1: Pareto fronts and design space evaluated by brute force, MESMOC, PESMOC and NSGA-II. Points beyond the upper right corner are not shown as they are far away from the Pareto front and affect the figure resolution if shown otherwise.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

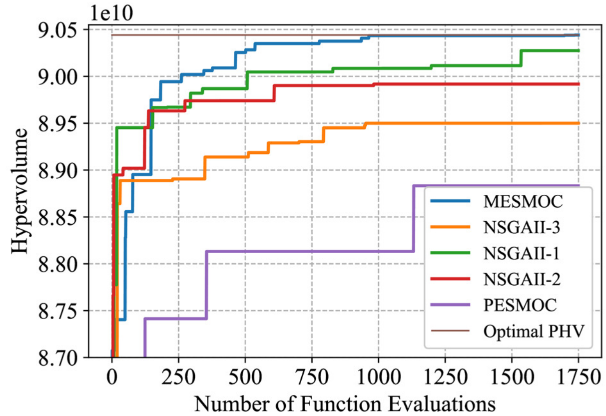


Fig. 5. Trial 1: PHV through the design search for MESMOC, PESMOC and three runs of NSGA-II.

MESMOC uncovered all six points in the optimal Pareto front; this was achieved with 1,736 design evaluations, equating to ~4% of the entire design space searched. In comparison, PESMOC and NSGA-II did not find any of the optimal Pareto optimal points under the same number of design evaluations. Additionally, given that the constraints are defined as a black-box, it is important to evaluate each algorithm's ability to select inputs that satisfy the constraints. PESMOC and NSGA-II experiments show poor performance with percentages of valid points selection of 4% and 39%, respectively. For MESMOC, 95% of the selected inputs are valid.

While Fig. 4 provides a visual aid to compare the Pareto fronts to the entire design space, it does not show the convergence behavior and progress of the algorithms. Hence the PHV metric shown in Fig. 5 is used, measuring the hypervolume of the best Pareto front throughout the search. The graph depicts how each algorithm's hypervolume approaches the volume of the optimal PHV, which is calculated from the brute force search results. Note that a single generation for NSGA-II consists of many iterations based on the population size, and an iteration is equivalent to a design evaluation. From Fig. 5, not only does MESMOC successfully discover the optimal Pareto front, it also converges faster than PESMOC and NSGA-II do to their best front. This experiment highlights MESMOC's ability to reduce design evaluations while also maintaining search accuracy.

C. *Trial 2: 6-Dimensional Design Space*

The terminating point for MOO algorithms such as GA or BO is a continuing challenge. When the optimal Pareto front is unknown, there is no deterministic method to generate a termination point. As such, it is common to interpret the given Pareto front after a set number of iterations as the most optimal; the approximate number depends on the application and complexity. Algorithms that provide an accurate and fast convergence rate are then highly favorable, as the quality of optimization is dependent on the Pareto front at the end of a search. Different from Subsection B, this subsection compares the non-optimal Pareto front quality of the proposed MESMOC

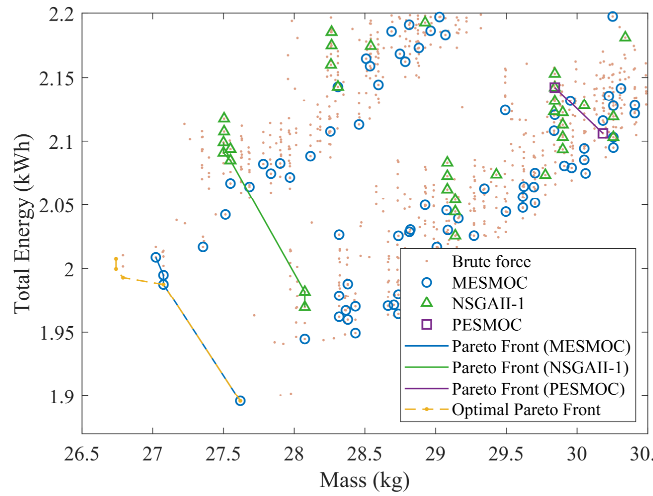


Fig. 6. Trial 2: Pareto fronts and design space evaluated by brute force, MESMOC, PESMOC, and NSGA-II.

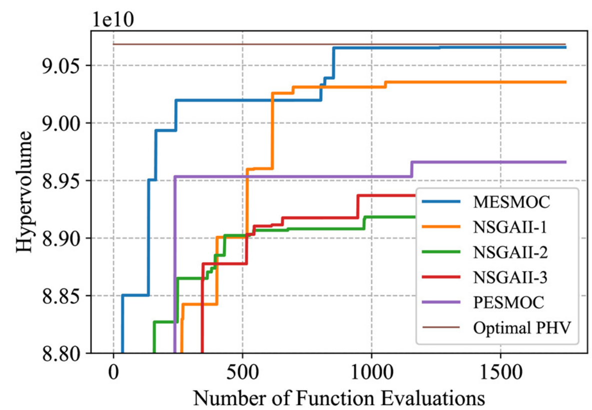


Fig. 7. Trial 2: PHV through the design search for both MESMOC, PESMOC, and three runs of NSGA-II.

to that of PESMOC and NSGA-II, by quadrupling the design space size while retaining the same number of design evaluations. To achieve this increase in design space, the inverter switching frequency, f_{sw}, becomes a design parameter, indicated in Table II. All other parameters and their ranges remain the same as in Trial 1. Thus, the total number of design combinations increases to 168,000. Under the same constrained optimization problem as Trial 1, Fig. 6 shows the Pareto fronts of all algorithms, along with the optimal Pareto front from brute force. MESMOC uncovered two of the five points on the Pareto front while PESMOC and NSGA-II did not discover any Pareto optimal points. Although MESMOC does not find the entire optimal Pareto front, it offers a strong final Pareto front. Moreover, MESMOC maintains its high performance at selecting 80% of valid designs while the performance of PESMOC and NSGA-II degrades further to selecting only 1% and 15% of valid points, respectively.

A PHV plot throughout the search with MESMOC, PESMOC, and NSGA-II is provided in Fig. 7, showing that the Pareto front found by MESMOC has a PHV quantitatively close to the optimal PHV. Compared to the PHV in Fig. 5 from Trial 1, MESMOC's PHV curve in Trial 2 grows much slower and contains extended periods of zero improvement. During the perceived periods of no improvement, MESMOC is still selecting designs to simulate that will maximize the information gain with respect to the optimal Pareto front. The intelligent component of MESMOC to continually improve the statistical model of the design space with respect to the objective functions guarantees continual Pareto front improvement. However, the increased design space size requires more design evaluations to build a robust statistical model of the objective functions. Before MESMOC was terminated at 1,750 designs, the PHV was still increasing. On the other hand, NSGA-II depends on random 'mutations' of its current set of designs for PHV improvement. Because of the GA's inherent randomness, there is no guarantee it will converge on the optimal Pareto front. The enhanced performance of the output-space entropy search used in MESMOC is made clear when compared to PESMOC, another BO algorithm.

While MESMOC can drastically reduce the number of design evaluations compared to brute force, PESMOC, and NSGA-II, it comes at a cost, admittedly. The MESMOC acquisition function relies on the previously evaluated design parameters and objective values. Thus, the computational cost of selecting the next design to simulate increases with respect to the iteration number. To put it into context, MESMOC consumes an average of 60 seconds per iteration throughout the optimization process (excluding physical model's simulation time) for Trial 1 and an average of 68 seconds per iteration for Trial 2. In terms of scalability, the time per iteration should increase linearly with the increase of number of constraints and objectives, experimentally shown in [6] with MESMO. This computational cost increase is part of the motivation behind the parallel implementation of MESMOC, as discussed in Section V-B. Regardless, the performance of MESMOC is highly favorable, as there is a tradeoff between computation time and optimization accuracy for systems with a large design space. In the end, the drastic reduction in the number of design evaluations outweighs the computation gain in parameter selection per iteration. The benefit is especially obvious when simulating complex physical models treated as a black-box, where each physical simulation iteration takes much time.

D. *One Optimal UAV Power System Design*

The UAV power system simulation is demonstrated with the Pareto optimal design of Trial 1 that had the greatest system mass but lowest energy consumption among the few Pareto points. This design, with parameters $N_s = 12$, $N_p = 26$, $N_m = 10$, $h_s = 12\,mm$, $N_t = 55$, and $f_{sw} = 30\,kHz$ resulted in a total UAV mass of $28.36\,kg$ and consumed $1.965\,kWh$ of energy. The system behavior throughout the 30-minute flight is given in Fig. 8. Fig. 8a provides the total system power consumption and loss, along with the battery SOC throughout the simulation. Fig. 8b-c contains the remaining design constraints of (33), specifically the motor and inverter temperatures, battery cell voltage, and inverter modulation index. While the temperatures are calculated as steady-state, they remain non-constant due to the feedback from temperature-dependent resistance.

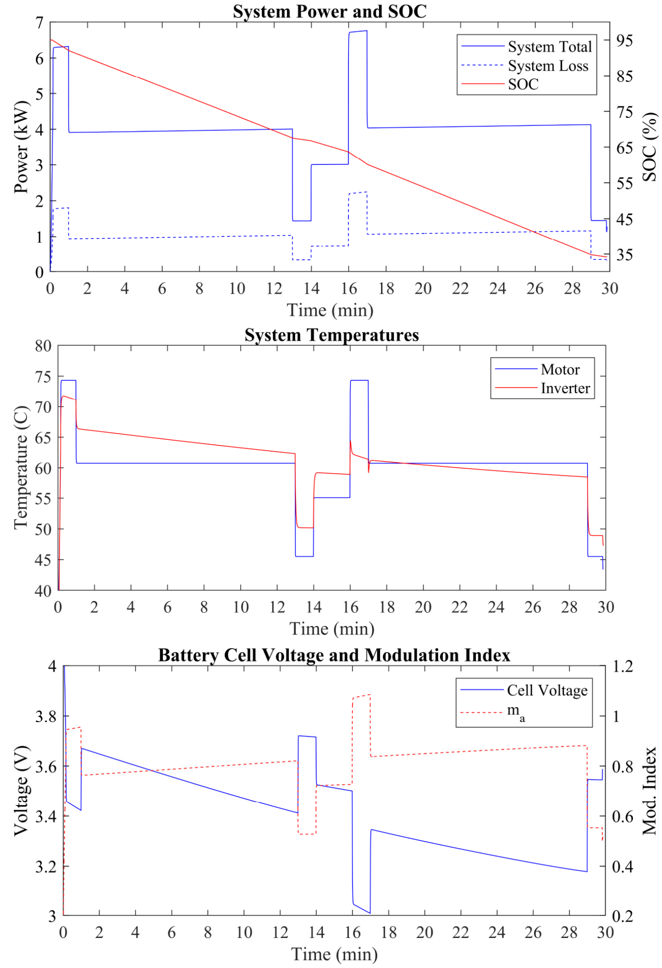


Fig. 8. 30-minute VTOL UAV simulation using the mission profile shown in Fig. 3 and Pareto optimal design parameters.

VII. Conclusion and Future Work

This paper presented a ML based design automation and multi-objective optimization approach for ETPS – an enhancesd model-based design methodology. A novel MESMOC ML algorithm is proposed, capable of searching through a constraint-heavy design space to efficiently discover Pareto optimal designs. This paper also serves as the first-of-its-kind that such an ML-based multi-objective BO under constraints is successfully demonstrated for an ETPS design. The proposed MESMOC algorithm is a specific instance of the general Bayesian optimization framework to optimize multiple objectives by handling constraints through expensive design simulations. By treating the physical simulation of ETPS as a black-box function, MESMOC builds statistical models of the objective functions and constraints to intelligently search through the design space to efficiently discover Pareto optimal designs. MESMOC was shown to minimize the number of design simulations, exploring only 4% of the design space to uncover the full optimal Pareto front in Trial 1 experiment, using an eVTOL UAV case study. In contrast, the popular GA NSGA-II and another BO algorithm PESMOC, were unsuccessful in their search. MESMOC consistently provided a better PHV than the baselines throughout all search iterations. The increased design space size in Trial 2 demonstrated MESMOC's superior Pareto front accuracy when the design space was extremely large to sufficiently explore, where two of the five Pareto optimal points were uncovered. Although there are some outstanding challenges, the ML based design automation offers a promising solution to significantly reduce engineering effort and time to determine an optimal power system design.

Future work will investigate the improvement of Pareto front discovery regardless of the size of the design space. This will be approached from the side of design problem setup by narrowing the breadth of search along with improvements in algorithm development. An investigation into the reliability of the search is also underway. Additional work will explore and verify multi-fidelity physical ETPS models and their interaction with ML.

ACKNOWLEDGEMENT

The authors would like to thank Alastair Thurlbeck of Oregon State University for his efforts in developing the UAV base models.

REFERENCES

[1] R. Aarenstrup, *Managing Model-Based Design*, Natick, MA: The MathWorks, Inc., 2015.

[2] Y. Cao, M. A. Williams, B. J. Kearbey, A. T. Smith, P. T. Krein, and A. G. Alleyne, "20x-Real Time Modeling and Simulation of More Electric Aircraft Thermally Integrated Electrical Power Systems," in *Proc. IEEE International Conf. Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles (ESARS-ITEC)*, 2016, pp. 1-6.

[3] A. Papageorgiou, M. Tarkian, K. Amadori, J. Ölvander, "Multidisciplinary Design Optimization of Aerial Vehicles: A Review of Recent Advancements", *International Journal of Aerospace Engineering*, vol. 2018, pp. 1-21, 2018.

[4] A. Ghosh and S. Dehuri, "Evolutionary Algorithms for Multi-criterion Optimization: A Survey," *International Journal of Computing & Information Sciences*, vol. 2, pp. 38-57, 2004.

[5] S. Zhao, F. Blaabjerg and H. Wang, "An Overview of Artificial Intelligence Applications for Power Electronics," *IEEE Transactions on Power Electronics*, vol. 36, no. 4, pp. 4633-4658, 2021.

[6] S. Belakaria, A. Deshwal, and J. R. Doppa, "Max-value Entropy Search for Multi-Objective Bayesian Optimization," in *Proc. Advances in Neural Information Processing Systems (NeurIPS) Conf.*, 2019, pp. 7823-7833.

[7] E. Silvas, T. Hofman, N. Murgovski, L. F. P. Etman and M. Steinbuch, "Review of Optimization Strategies for System-Level Design in Hybrid Electric Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 57-70, 2017.

[8] J. Knowles, "ParEGO: A Hybrid Algorithm with On-line Landscape Approximation for Expensive Multiobjective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50-66, 2006.

[9] D. Eriksson, M. Pearce, J. Gardner, R. D Turner, and M. Poloczek, "Scalable Global Optimization via Local Bayesian Optimization," in *Proc. Advances in Neural Information Processing Systems (NeurIPS) Conf.*, 2019, pp. 5497–5508.

[10] S. Falkner, A. Klein, and F. Hutter. "BOHB: Robust and Efficient Hyperparameter Optimization at Scale," in *Proc. of International Conference on Machine Learning (ICML)*, 2018, pp. 1-19.

[11] A. Nayebi, A. Munteanu, and M. Poloczek, "A Framework for Bayesian Optimization in Embedded Subspaces," in *Proc. of International Conference on Machine Learning (ICML)*, 2019, pp. 4752–4761.

[12] X. Hu, J. Han, X. Tang and X. Lin, "Powertrain Design and Control in Electrified Vehicles: A Critical Review," *IEEE Transactions on Transportation Electrification*, pp. 1-19, 2021.

[13] L. Zhang, M. Lv, W. Zhu, H. Du, J. Meng, J. Li, "Mission-based Multidisciplinary Optimization of Solar-powered Hybrid Airship," *Journal of Energy Conversion and Management*, vol. 185, pp. 44-54, 2019.

[14] N. B. Hadj, J. K. Kammoun and R. Neji, "Application of Evolutionary Algorithm for Triobjective Optimization: Electric Vehicle," *Int. Journal of Energy Optimization and Engineering (IJEOE)*, vol. 3, no. 3, pp. 1-19, 2014.

[15] C. Desai and S. S. Williamson, "Optimal Design of a Parallel Hybrid Electric Vehicle Using Multi-objective Genetic Algorithms," in *Proc. IEEE Vehicle Power and Propulsion Conf.*, 2009, pp. 871-876.

[16] X. Wu, B. Cao, J. W. and Z. Wang, "Application of Particle Swarm Optimization for Component Sizes in Parallel Hybrid Electric Vehicles," in *Proc. IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)* , 2008, pp. 2874-2878.

[17] M. Hauschild, M. Pelikan, "An Introduction and Survey of Estimation of Distribution Algorithms," *Journal of Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111-128, 2011.

[18] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148-175, 2016.

[19] J. M. Hernández-Lobato, et al., "Predictive Entropy Search for Bayesian Optimization with Unknown Constraints," in *Proc. International conference on machine learning*, 2015, pp. 1699-1707.

[20] M. Emmerich and J.W. Klinkenberg, "The Computation of the Expected Improvement in Dominated Hypervolume of Pareto Front Approximations," Technical Report, Leiden University, pp. 1-8, 2008.

[21] D. Hernandez-Lobato, J. Hernandez-Lobato, A. Shah, and R. Adams, "Predictive Entropy Search for Multi-objective Bayesian Optimization," in *Proc. of International Conference on Machine Learning (ICML)*, 2016, pp. 1492–1501.

[22] E. C. Garrido-Merchán and D. Hernández-Lobato, "Predictive Entropy Search for Multi-objective Bayesian Optimization with Constraints," *Neurocomputing*, vol. 361, pp. 50–68, 2019.

[23] Z. Zhou, S. Belakaria, A. Deshwal, W. Hong, J. R. Doppa, P. Pande, and D. Heo, "Design of Multi-Output Switched-Capacitor Voltage Regulator via Machine Learning," in *Proc. IEEE/ACM Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020, pp. 502-507.

[24] Y. Cao, R. C. Kroeze, and P. T. Krein, "Multi-timescale Parametric Electrical Battery Model for Use in Dynamic Electric Vehicle Simulations," *IEEE Trans. Transportation Electrification*, vol. 2, no. 4, pp. 432-442, 2016.

[25] A. Thurlbeck and Y. Cao, "Analysis and Modeling of UAV Power System Architectures," in *Proc. IEEE Transportation Electrification Conf. (ITEC)*, 2019, pp. 1-8.

[26] Y. Cao and A. Thurlbeck, "Heavy-duty UAV Electric Propulsion Architectures and Multi-timescale Multi-physics Modeling," in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

Proc. AIAA/IEEE Electric Aircraft Technologies Symposium (EATS), 2019, pp. 1-13.
[27] A. Rahimi and B. Recht, "Random Features for Large-scale Kernel Machines," in *Proc. Advances in Neural Information Processing Systems (NeurIPS) Conf.*, 2008, pp. 1177-1184.
[28] D. Jackson, S. Belakaria, Y. Cao, J.R. Doppa, X. Lu, "Machine Learning Enabled Fast Multi-Objective Optimization for Electrified Aviation Power System Design," in *Proc. IEEE Energy Conversion Congress & Expo (ECCE)*, 2020, pp. 6385-6390.
[29] John H., G. A. Goodarzi, *Electric Powertrain: Energy Systems, Power Electronics and Drives for Hybrid, Electric and Fuel Cell Vehicles*, Hoboken, NJ: John Wiley & Sons, 2018.
[30] D. Graovac and M. Purschel, "MOSFET Power Losses Calculation Using the Data-Sheet Parameters," Infineon Application Note, 2006.
[31] T. A. Lipo, *Introduction to AC Machine Design*, Hoboken, NJ: John Wiley & Sons, 2017.
[32] C. Williams and C. Rasmussen, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.
[33] Z. Wang and S. Jegelka, "Max-value Entropy Search for Efficient Bayesian Optimization," in *Proc. International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017, pp. 1-12.
[34] D. Jones, C. Perttunen, and B. Stuckman, "Lipschitzian Optimization without the Lipschitz Constant," *Journal of Optimization Theory and Applications*, vol. 79, no. 1, pp. 157-181, 1993.
[35] K. Deb, A. Pratap, S. Agarwal, and TAMT Meyarivan, "A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, 2002.
[36] T. Cover and J. Thomas, *Elements of Information Theory*, 2nd ed. Hoboken, New Jersey, USA: John Wiley & Sons, 2006.
[37] J. Michalowicz, J. Nichols, and F. Bucholtz, *Handbook of Differential Entropy*. Beaverton, OR, USA: Chapman and Hall/CRC, 2014.
[38] C. Y. Oh, E. Gavves and M. Welling, "BOCK: Bayesian Optimization with Cylindrical Kernels," in *Proc. International Conference on Machine Learning*, pp. 3868-3877, 2018.
[39] Engineering ToolBox, "Nema Insulation Classes," 2004. www.engineeringtoolbox.com/nema-insulation-classes-d_734.html
[40] P. T. Krein, *Elements of Power Electronics,* 2nd ed. New York: Oxford University Press, 2015.
[41] E. Zitzler, "Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications," PhD thesis, ETH Zurich, Switzerland, 1999.
[42] J.R. Doppa, "Adaptive Experimental Design for Optimizing Combinatorial Structures", in *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4940-4945, 2021.
[43] A. Deshwal, S. Belakaria, J.R. Doppa, "Mercer Features for Efficient Combinatorial Bayesian Optimization", in *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pp. 7210-7218, 2021.
[44] A. Deshwal, S. Belakaria, J.R. Doppa, "Bayesian Optimization over Hybrid Spaces", in *Proc. International Conference on Machine Learning (ICML),* 2632-2643, 2021.
[45] S. Belakaria, A. Deshwal, N.K. Jayakodi, J.R. Doppa, "Uncertainty-aware Search Framework for Multi-Objective Bayesian Optimization", in *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pp. 10044-10052, 2020.
[46] S. Belakaria, A. Deshwal, J.R. Doppa, "Multi-Fidelity Multi-Objective Bayesian Optimization: An Output Space Entropy Search Approach", in *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pp. 10035-10043, 2020.

Derek Jackson (Student Member, IEEE) received the B.S. and M.S. degrees in electrical and computer engineering with a minor in computer science from Oregon State University (OSU), Corvallis, OR, USA, in 2019 and 2021, respectively. Derek is currently pursuing a Ph.D. in electrical and computer engineering as part of the Energy Systems Group at OSU.

Derek was an electrical engineering intern with the Product Validation Department at Daimler Trucks NA in Portland, OR, USA; and an electrical engineering intern with the Product Design Group at Blount International (now Oregon Tool) in Portland, OR, USA. His research interests include power electronics, motor drives, and power system design and optimization for transportation electrification and microgrids. Derek received the IEEE ECCE Conference Student Travel Award in 2020. He is vice-president of the IEEE Power & Energy Society (PES) and Power Electronics Society (PELS) Chapter at OSU.

Syrine Belakaria received the engineering degree in telecommunication from the Higher School of Communication of Tunis (SupCom), Tunisia, in 2016 and the master's degree in electrical engineering from university of Idaho, USA. She is currently pursuing her PhD in computer science at Washington State University. Her research focuses on AI-driven adaptive experiment design for science and engineering applications including hardware design, materials design, and electric transportation systems. Her awards and honors include winning the IBM PhD Fellowship (2021-2023), being a finalist for both Microsoft Research Fellowship (2021) and Open Philanthropy AI Fellowship (2020), the Outstanding Research Assistant in EECS (2021) and the Outstanding Teaching Assistant in Computer Science from the College of Engineering, WSU. She has published in top-tier AI venues including AAAI, NeurIPS, ICML, and JAIR.

Yue Cao (Member, IEEE) received the B.S. degree (Hons.) in electrical engineering with a second major in mathematics from the University of Tennessee, Knoxville, TN, USA, in 2011, and the M.S. and Ph.D. degrees in electrical engineering from the University of Illinois at Urbana–Champaign (UIUC), Champaign, IL, USA, in 2013 and 2017, respectively.

Dr. Cao is currently an Assistant Professor with the Energy Systems Group at Oregon State University (OSU), Corvallis, OR, USA. Before joining OSU, he was a Research Scientist with the Propulsions Team at Amazon Prime Air in Seattle, WA, USA. He was a Power Electronics Engineer Intern with Special Projects Group at Apple Inc., Cupertino, CA, USA; Halliburton Company, Houston, TX, USA; Flanders Electric, Evansville, IN, USA; and Oak Ridge National Laboratory, TN, USA. He was a Sundaram Seshu Fellow in 2016 at UIUC, where he was a James M. Henderson Fellow in 2012. His research interests include power electronics, motor drives, and energy storage with applications in renewable energy integration and transportation electrification.

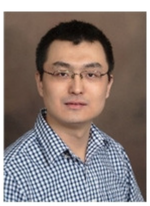
Dr. Cao was a national finalist of the USA Mathematical Olympiad (USAMO) in 2006 and 2007. He received the Myron Zucker Award from the IEEE Industry Applications Society (IAS) in 2010. He was a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TTE.2021.3113958, IEEE Transactions on Transportation Electrification

recipient of Oregon State Learning Innovation Award for transformative education in 2020. He is currently the Tutorials Chair of the 2021 IEEE Energy Conversion Congress Expo (ECCE) and the Special Sessions Chair of 2022 ECCE. He is a board member and Award Chair of IEEE Power Electronics Society (PELS) TC11 – Aerospace Power. In 2020, he helped establish an IEEE PELS Chapter at OSU. He is currently an Associate Editor for IEEE Transactions on Transportation Electrification.

Jana Doppa is the George and Joan Berry Distinguished Associate Professor in the School of Electrical Engineering and Computer Science at Washington State University, Pullman. He received his Ph.D. degree in Computer Science from Oregon State University and his M.Tech. degree from Indian Institute of Technology (IIT), Kanpur. His research focuses on both foundational and applied aspects of artificial intelligence to accelerate scientific discovery and engineering design for domains including electronic design automation, nanoporous materials design, biological sequence design, and electric transportation systems design.

His research has been recognized with a number of awards, honors, and distinctions including the 2019 National Science Foundation CAREER Award; the 2021 Early Career Award in AI by the International Joint Conference on Artificial Intelligence; the 2021 Best Paper Award from ACM Transactions on Design Automation of Electronic Systems; the 2013 Outstanding Paper Award from the AAAI Conference on Artificial Intelligence; the 2018 Best Student Abstract Award from the AAAI Conference on Artificial Intelligence; the 2015 Outstanding PhD Dissertation Award from Oregon State University and was nominated for ACM Doctoral Dissertation Award; a 2015 Google Faculty Research Award; the 2020 Outstanding Junior Faculty Research Award and the 2018 Reid-Miller Teaching Excellence Award from the College of Engineering, Washington State University. He is on the editorial board of Journal of Artificial Intelligence Research and Machine Learning Journal.



Xiaonan Lu (Member, IEEE) received his B.E. and Ph.D. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2008 and 2013, respectively. In 2010-2011, he was a guest Ph.D. student at the Department of Energy Technology, Aalborg University, Denmark. From 2013 to 2014, he was a Postdoc Research Associate at the Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, USA. During 2015-2018, he was with Argonne National Laboratory, USA, first as a Postdoc Appointee and then as an Energy Systems Scientist. In 2018, he joined the College of Engineering at Temple University as an Assistant Professor. His research interests include modeling and control of power electronic inverters, hybrid AC and DC microgrids, and real-time hardware-in-the-loop simulation. Dr. Lu is an Associate Editor of IEEE Transactions on Industrial Electronics, an Associate Editor of IEEE Transactions on Industry Applications, and an Editor of IEEE Transactions on Smart Grid. He serves as the Vice Chair of the Industrial Power Converters Committee (IPCC) in the IEEE Industry Applications Society (IAS). He received the 2020 Young Engineer of the Year Award in the IEEE Philadelphia Section.