Structural and Multidisciplinary Optimization (2024) 67:36
https://doi.org/10.1007/s00158-023-03731-5

RESEARCH PAPER

Multi-fidelity error-estimate-based model management

Tucker Babcock[1] · **Dustin Hall**[2] · **Justin S. Gray**[2] · **Jason E. Hicken**[1]

Received: 22 May 2023 / Revised: 28 November 2023 / Accepted: 19 December 2023 / Published online: 28 February 2024

Abstract
This paper presents a novel multi-fidelity model-management framework based on the estimated error between the low-fidelity and high-fidelity models. The optimization algorithm is similar to classical multi-fidelity trust-region model-management approaches, but it replaces the trust-radius constraint with a bound on the estimated error between the low- and high-fidelity models. This enables globalization without requiring the user to specify non-intuitive parameters such as the initial trust radius, which have a significant impact on the cost of the optimization yet can be hard to determine a priori. We demonstrate the framework on a simple one-dimensional optimization problem, a series of analytical benchmark problems, and a realistic electric-motor optimization. We show that for low-fidelity models that accurately capture the trends of the high-fidelity model, the developed framework can significantly improve the efficiency of obtaining high-fidelity optima compared to state-of-the-art multi-fidelity optimization methods and a direct high-fidelity optimization.

Keywords Multi-fidelity optimization · Electric-motor optimization · Error estimates

1 Introduction

High-fidelity numerical modeling and optimization tools have become an integral part of the detailed design process for systems such as aircraft and their components. However, the large computational cost of high-fidelity analysis and optimization often precludes its use at earlier stages in the design process, ultimately limiting the effectiveness of such analyses. This is unfortunate, as it is during the conceptual design phase that engineers have the most flexibility to explore the design space and consider novel concepts. Currently, decisions at the conceptual design phase are largely dominated by experience, engineering intuition, and low-fidelity analyses. These decision-making processes can be quite effective when designing systems that iterate on past designs. However, they can fail when the designers are engaging in clean-sheet design and exploring truly novel concepts. It is in these cases where the use of high-fidelity analysis and optimization would be most effective at supporting the design process.

Multi-fidelity analysis and optimization methods provide an effective framework for combining the computational efficiency of low-fidelity methods with the accuracy of high-fidelity tools. When dealing with models of multiple fidelities, it is essential to have a methodology to effectively distribute work between the models to balance speed and accuracy. At the 2010 National Science Foundation workshop on Multidisciplinary Design Optimization for Complex Engineered Systems, Boeing Technical Fellow Dr. Evin Cramer highlighted this need for a model-management strategy by identifying several aspects of multi-fidelity modeling that need to be addressed (Simpson and Martins 2011). Namely, she identified the difficulty in choosing the correct level of fidelity for the right application, the ability to effectively use multiple levels of fidelity at once, and a lack of maturity in multi-fidelity tools that preclude their industrial adoption.

In a review of multi-fidelity methods, Peherstorfer et al. (2018) differentiate between three types of multi-fidelity model-management strategies: adaptation, fusion, and filtering. They define adaptation strategies as those that adapt the low-fidelity model based on information from the high-fidelity model, fusion strategies as those that combine low- and

Responsible Editor: Yoojeong Noh

✉ Tucker Babcock
tuckerbabcock1@gmail.com

[1] Department of Mechanical, Aerospace, and Nuclear Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

[2] Propulsion Systems Analysis Branch, MS 5-11, NASA Glenn Research Center, Cleveland, OH 44135, USA

high-fidelity outputs, and filtering strategies as those that use the high-fidelity model only when indicated by a low-fidelity model. For completeness, the following paragraphs highlight some relevant literature related to each of these multi-fidelity approaches.

Multi-fidelity optimization approaches that employ fusion strategies typically follow the Efficient Global Optimization (EGO) framework described by Jones et al. (1998). In these approaches, one constructs a multi-fidelity surrogate (or emulator) that *fuses* data from the low- and high-fidelity models. Various fusion strategies have emerged through the years. Kennedy and O'Hagan (2001) developed a Gaussian-process-based multi-fidelity method to learn the discrepancy between a low- and high-fidelity model in a Bayesian framework. Forrester et al. (2007) developed a co-Kriging-based multi-fidelity surrogate that eases some of the computational burden associated with estimating the Gaussian-process (GP) hyperparameters. More recently, Eweis-Labolle et al. (2022) developed a generalized multi-fidelity surrogate based on latent map GPs that can efficiently fuse arbitrary numbers of models, and can support discrete inputs.

There have been numerous studies developed that optimize these GP-based multi-fidelity surrogates; we refer the reader to the literature (Keane 2003; Forrester et al. 2007; Jo and Choi 2014; Foumani et al. 2023) for examples. Despite their broad usage and effectiveness on many problems, EGO-type multi-fidelity optimization formulations have difficulties handling large numbers of design variables and general nonlinear constraints (Shi et al. 2021). Despite recent efforts to ameliorate the difficulty of GPs to handle large design spaces (Shan and Wang 2010; Eriksson and Jankowiak 2021), the curse of dimensionality is still a problem (Viana et al. 2014; Shi et al. 2021).

Multi-fidelity filtering strategies are perhaps the least studied in the literature, though there has been a renewed interest lately. Réthoré et al. (2014) used a filtering-based optimization strategy to optimize a wind farm layout. Further, Wu et al. (2022b) developed a sequential multi-fidelity approach specifically designed for multi-disciplinary problems that can consider arbitrary levels of fidelity for each discipline. The authors subsequently used this sequential multi-fidelity approach to perform a multi-fidelity aerostructural optimization of a large-scale transport aircraft (Wu et al. 2022a). Despite the great promise shown by filtering methods, the implementations typically require modifications to the underlying software, making it harder for general practitioners to take advantage of them.

Adaptation model-management strategies have largely been built upon the Trust-Region Model-Management (TRMM) approach introduced by Lewis (1996) and Alexandrov et al. (1998). Originally limited to unconstrained optimization, Alexandrov et al. (2001) later extended the TRMM framework to a general Approximation and Model-Management Optimization (AMMO) design-optimization framework, supporting augmented Lagrangian optimization, multilevel algorithms for large-scale constrained optimization (MAESTRO), and sequential quadratic programming (SQP). TRMM methods generalize the classic trust-region SQP optimization method by replacing the high-fidelity model's quadratic approximation with a low-fidelity model. By calibrating the low-fidelity model at each optimization iteration—such that the objectives and constraints and their gradients are equal to the high-fidelity model—the sequence of low-fidelity sub-optimizations will provably converge to the high-fidelity optimum (Alexandrov et al. 2001).

These TRMM-based methods have been the subject of continued research interest. Gratton et al. (2008) developed a method to recursively apply the TRMM framework to arbitrary levels of fidelity. Their method is similar to the class of multigrid methods used for solving partial differential equations, as it switches between levels of fidelity to accelerate convergence to the optimum. Olivanti et al. (2019) extended this approach with a new gradient-based criterion to determine when to switch between fidelity levels. March and Willcox (2012b) further developed a variation of the TRMM framework that satisfies the high-fidelity first-order optimality conditions without needing to evaluate high-fidelity gradients. Subsequently, they extended this method to support constrained optimization (March and Willcox 2012a). Elham and van Tooren (2017) replaced the trust region merit function-based step acceptance criteria with a filter method that considers decreases in the objective function and infeasibility separately when deciding to accept or reject a step. Nagawkar et al. (2021) developed a method that achieves the required first-order consistency between the low- and high-fidelity models by using a manifold mapping to ensure that the low-fidelity model is a reliable representation of the high-fidelity model during the low-fidelity sub-optimization process.

Outside of the TRMM-based approaches, Bryson and Rumpfkeil (2018) developed a multi-fidelity quasi-Newton framework that uses the low-fidelity model in its line-search procedure. Compared to conventional TRMM methods that only calibrate the low-fidelity model at each design iteration, this quasi-Newton method also builds and maintains a high-fidelity Hessian approximation. By maintaining this high-fidelity Hessian approximation, the multi-fidelity method is able to pick more effective descent directions than would be possible using the low-fidelity Hessian. Further, the algorithm can efficiently switch to a direct high-fidelity optimization should the low-fidelity model be deemed too inaccurate. Finally, Hart and van Bloemen Waanders (2023) developed an approach that uses post-optimality sensitivities with respect to

model discrepancy at the end of the low-fidelity optimization to update the high-fidelity optimization solution.

While the multi-fidelity quasi-Newton approaches show great promise at reducing the cost of finding high-fidelity optima, their algorithms requires a special implementation and cannot use an off-the-shelf optimization algorithm, likely limiting its adoption. In the case of TRMM-based approaches, the trust-region constraint introduces parameters (e.g., the initial radius and radius scaling factors) that are not intuitive and may be difficult for practitioners to define. While TRMM and related algorithms will converge robustly for a wide range of parameters, computational cost can be adversely affected by poor choices (Conn et al. 2000, Chapter 17; Gould et al. 2005). In addition to the potential issues associated with parameter selection, we hypothesize that the isotropy of the trust-radius constraint can impede optimization progress as it cannot account for possible anisotropy in the error in the calibrated low-fidelity model. Thus, *the main algorithmic contribution of this work is to define the trust region in terms of the estimated error between the low- and high-fidelity models*. This definition allows the optimization to take higher-quality steps than conventional TRMM methods. Furthermore, users can then select a target error tolerance for, say, the objective function rather than needing to define non-intuitive parameters. Thus, we present a multi-fidelity model-management framework based on error estimates between the low- and high-fidelity models.

The rest of this paper is organized as follows. Section 2 details the error-estimate calculation. Section 3 describes the proposed model-management framework. Section 4 presents results from the error-estimate model-management framework, considering a simple demonstration problem, a series of analytical benchmark problems, and a realistic problem showcasing the optimization of an electric motor. Finally, Sect. 5 discusses the presented model-management framework and highlights future areas of research.

2 Error estimates

Consider a high-fidelity model $f_{\text{hi}} : \mathbb{R}^n \to \mathbb{R}$ and a low-fidelity model $f_{\text{lo}} : \mathbb{R}^n \to \mathbb{R}$. Let $\boldsymbol{x} \in \mathbb{R}^n$ be the common design variables shared by the two models. Fernández-Godino et al. (2016) identify two distinct categories used to calibrate the low-fidelity model to the high-fidelity data: additive/multiplicative corrections, and comprehensive corrections.

Additive corrections are of the form:

$$\hat{f}^{(k)}(\boldsymbol{x}) = f_{\text{lo}}(\boldsymbol{x}) + \gamma^{(k)}(\boldsymbol{x}), \tag{1}$$

where $\gamma^{(k)}(\boldsymbol{x})$ is the additive correction function defined based on the calibration point $\boldsymbol{x}^{(k)}$. Superscripts (k) indicate that a quantity has been evaluated at or is defined by the k-th calibration point $\boldsymbol{x}^{(k)}$. Multiplicative corrections are of the form:

$$\hat{f}^{(k)}(\boldsymbol{x}) = \beta^{(k)}(\boldsymbol{x}) f_{\text{lo}}(\boldsymbol{x}), \tag{2}$$

where $\beta^{(k)}(\boldsymbol{x})$ is the multiplicative correction function. The order of the calibration indicates the level of continuity between low- and high-fidelity models; zeroth-order calibration implies that $\hat{f}^{(k)}\left(\boldsymbol{x}^{(k)}\right) = f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)$, while first-order calibration requires that both $\hat{f}^{(k)}\left(\boldsymbol{x}^{(k)}\right) = f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)$ and $\nabla\hat{f}^{(k)}\left(\boldsymbol{x}^{(k)}\right) = \nabla f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)$, and so on for higher-order calibrations. Comprehensive corrections encompass all other available correction schemes.

Alexandrov et al. (1998, 2001) showed that the TRMM strategy is provably convergent to a high-fidelity optimum as long as at least first-order calibrated models are used for each response function (the objective and each constraint). To that end, we consider the first-order additive calibration models:

$$\hat{f}^{(k)}(\boldsymbol{x}) = f_{\text{lo}}(\boldsymbol{x}) + \gamma^{(k)}(\boldsymbol{x}), \tag{1}$$

calibrated about the reference point $\boldsymbol{x}^{(k)}$, where the correction term is defined as follows:

$$\begin{aligned}\gamma^{(k)}(\boldsymbol{x}) = {} & f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right) - f_{\text{lo}}\left(\boldsymbol{x}^{(k)}\right) \\ & + \left(\nabla f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right) - \nabla f_{\text{lo}}\left(\boldsymbol{x}^{(k)}\right)\right)\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right),\end{aligned} \tag{3}$$

and we follow the convention that gradients are row vectors.

Next, we define the error between the high-fidelity model and the calibrated low-fidelity model as follows:

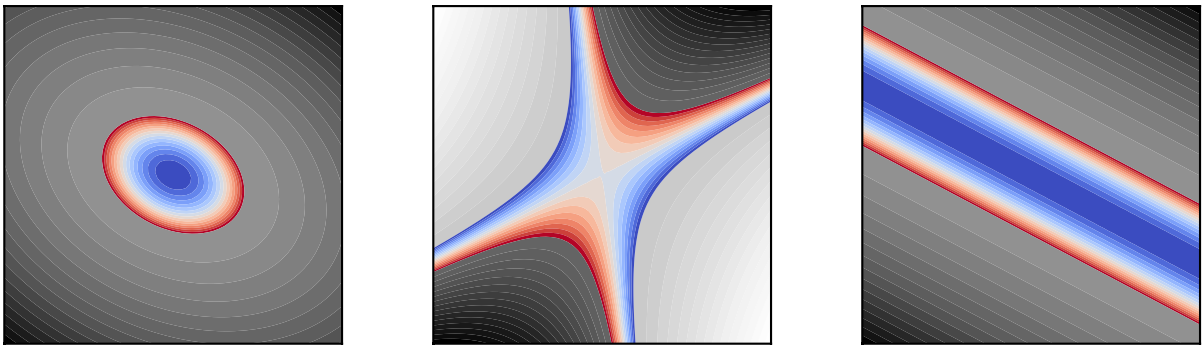
$$E^{(k)}(\boldsymbol{x}) = \hat{f}^{(k)}(\boldsymbol{x}) - f_{\text{hi}}(\boldsymbol{x}). \tag{4}$$

We take a Taylor series expansion of Eq. (4) about $\boldsymbol{x}^{(k)}$ to estimate the error in the calibrated model at an arbitrary design vector $\boldsymbol{x}$ without needing to re-evaluate the high-fidelity model. Considering the first-order calibration scheme given in Eqs. (1) and (3), we obtain the following second-order error estimate, which we distinguish from the exact error with a hat:

$$\hat{E}^{(k)}(\boldsymbol{x}) = \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right)^{\mathrm{T}} \mathsf{H}_{\Delta}^{(k)}\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right), \tag{5}$$

where $\mathsf{H}_{\Delta}^{(k)} = \nabla^2\hat{f}^{(k)}\left(\boldsymbol{x}^{(k)}\right) - \nabla^2 f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)$, the difference in the Hessians of the calibrated low-fidelity and high-fidelity models.

For many engineering problems of interest, the Hessian matrices $\nabla^2\hat{f}^{(k)}\left(\boldsymbol{x}^{(k)}\right)$ and $\nabla^2 f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)$ are not available, either due to excessive computational cost or storage requirements. To address this concern, one can approximate the Hessian difference using methods such as quasi-Newton approaches, or Arnoldi's method, which uses matrix–vector products to construct a low-rank approximation of the target matrix.



(a) Definite Hessian difference (b) Indefinite Hessian difference (c) Semidefinite Hessian difference

Fig. 1 The second-order error constraints have an unbounded feasible region when the difference in the model's Hessians is indefinite or semidefinite

When Hessian-vector products are not available explicitly, one can compute approximate Hessian-vector products by performing a directional finite-difference of the gradient.

2.1 Characterizing the error constraints

Given the ultimate goal of using the error estimate given in Eq. (5) as a constraint in an optimization, it is important to be able to characterize how the error bounds will affect the optimization. We now describe the properties of the second-order error estimate, and we propose modifications to ensure their suitability for use in an optimization.

The feasible region will take on different shapes depending on the definite-ness of $\mathsf{H}_\Delta^{(k)}$. In the case of a positive or negative definite Hessian difference, the feasible region is bounded by an ellipsoid, as illustrated for a generic two-dimensional second-order constraint in Fig. 1a. If the Hessian difference is indefinite, the feasible region becomes unbounded, in the shape of a saddle. The interface between the feasible and infeasible regions is the hyperboloid, offset from the saddle point by the constraint tolerance, as illustrated in Fig. 1b. Finally, if the Hessian difference is semidefinite, the feasible region is again unbounded, as illustrated in Fig. 1c.

While the unbounded constraints are not necessarily inaccurate, they are based on local information and will likely become inaccurate as the size of the design step grows. Thus, we wish to bound the feasible region so that we may remain in the region where the error estimates are accurate. For the case of either negative- or positive-definite Hessian differences, there is no work to be done, as the feasible region is already bounded by an ellipsoid. To remedy the unbounded constraint for the case of indefinite and semidefinite Hessian differences, we find an upper bound on the absolute value of the error estimates and use that bound as our constraint.

We start by decomposing the symmetric Hessian difference into its spectral decomposition $\mathsf{H}_\Delta^{(k)} = \mathsf{V}\Lambda\mathsf{V}^\mathrm{T}$, where V holds the eigenvectors of $\mathsf{H}_\Delta^{(k)}$ and Λ is a diagonal matrix with the eigenvalues of $\mathsf{H}_\Delta^{(k)}$ as its diagonal entries. As Λ is a diagonal matrix, we can then simplify Eq. (5) to

$$\hat{E}^{(k)}(\boldsymbol{x}) = \frac{1}{2}\sum_i \Lambda_{i,i}\left(y_i^{(k)}\right)^2, \tag{6}$$

where $\boldsymbol{y}^{(k)} = \mathsf{V}^T\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right)$, and the subscript i denotes an index into the matrix and vector. Next, we use the triangle inequality to bound the absolute value of the sum given in Eq. (6):

$$\left|\hat{E}^{(k)}(\boldsymbol{x})\right| = \left|\frac{1}{2}\sum_i \lambda_i\left(y_i^{(k)}\right)^2\right| \leq \frac{1}{2}\sum_i \left|\lambda_i\right|\left(y_i^{(k)}\right)^2, \tag{7}$$

where we have omitted the redundant absolute value around the squared $y_i^{(k)}$ term. Finally, we can expand the bounded sum and reverse the previous steps to find

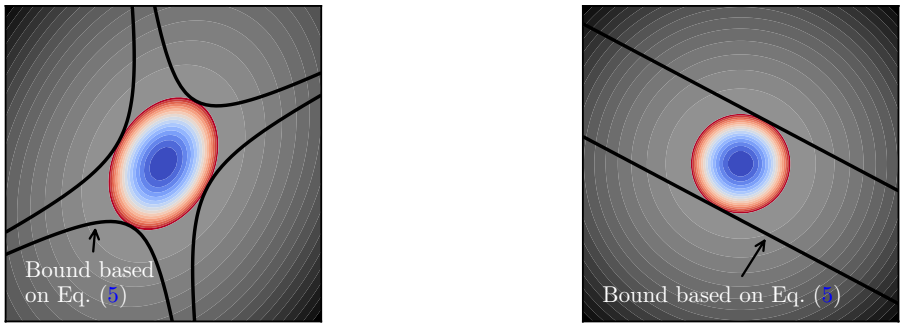
$$\left|\hat{E}^{(k)}(\boldsymbol{x})\right| \leq \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right)^\mathrm{T}\left|\mathsf{H}_\Delta^{(k)}\right|\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right), \tag{8}$$

where $\left|\mathsf{H}_\Delta^{(k)}\right|$ indicates a modification to $\mathsf{H}_\Delta^{(k)}$ such that each eigenvalue has been replaced with its absolute value. Henceforth, we will use

$$\tilde{E}^{(k)}(\boldsymbol{x}) = \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right)^\mathrm{T}\left|\mathsf{H}_\Delta^{(k)}\right|\left(\boldsymbol{x} - \boldsymbol{x}^{(k)}\right) \tag{9}$$

to denote the estimated bound that we use in the model-management framework.

This procedure works well to create bounded steps in the case of full-rank indefinite Hessian differences, as illustrated in Fig. 2a, which shows the modified error estimate for the same indefinite Hessian difference as shown in Fig. 1b. However, in the case of semidefinite or rank-deficient Hessian



(a) Bounded feasible region for an indefinite Hessian difference (b) Bounded feasible region for a semidefinite Hessian difference

Fig. 2 Redefining the second-order error estimates to use a full-rank positive-definite modification to the Hessian difference ensures that each error constraint results in a bounded feasible region, even for indefinite and semidefinite Hessian differences

differences, we can still end up with unbounded steps. To remedy this, we replace each zero eigenvalue of the Hessian difference with the smallest non-zero eigenvalue. This ensures that a step in any direction is bounded and that we are not overly conservative with step bounds in directions where the Hessian difference is small. Error-estimate constraint contours based on the new bounds given in Eq. (9) are plotted in Fig. 2b for the same semidefinite Hessian difference shown in Fig. 1c.

An inherent assumption of our error-estimate model is that the quadratic error model is sufficiently accurate within the error bounds used during the optimization. If the model is highly nonlinear, or if the error bounds are too large, this assumption may be invalid, and the constraints may fail to properly globalize the optimization.

3 Error-estimate-based model management

This section describes the details of the proposed multi-fidelity error-estimate-based model management (E^2M^2) framework. Specifically, we describe the steps of the optimization algorithm and discuss the role of the error estimates as constraints in the optimization procedure.

We start by considering general, non-linearly constrained optimization problems of the form:

$$\begin{aligned} &\min_{\boldsymbol{x}} \ f_{\text{hi}}(\boldsymbol{x}) \\ &\text{s.t.} \quad \boldsymbol{g}_{\text{hi}}(\boldsymbol{x}) = \boldsymbol{0} \\ &0 \quad \boldsymbol{h}_{\text{hi}}(\boldsymbol{x}) \leq \boldsymbol{0}, \end{aligned} \tag{10}$$

where $f_{\text{hi}} : \mathbb{R}^n \to \mathbb{R}$ is the high-fidelity objective function, $\boldsymbol{g}_{\text{hi}} : \mathbb{R}^n \to \mathbb{R}^{m_g}$ is the high-fidelity equality constraint function, and $\boldsymbol{h}_{\text{hi}} : \mathbb{R}^n \to \mathbb{R}^{m_h}$ is the high-fidelity inequality constraint function, and m_g and m_h represent the number of equality and inequality constraints, respectively.

The proposed E^2M^2 framework is an iterative procedure. We start with an initial design vector $\boldsymbol{x}^{(0)}$, optimality and feasibility tolerances ϵ_{opt} and ϵ_{feas}, and user-specified error bounds τ_{abs} and τ_{rel} for each response function.

At each iteration k, the low- and high-fidelity models and their gradients are evaluated at the current design vector $\boldsymbol{x}^{(k)}$. Then, using Eq. (3), calibration models are constructed for each response function. Next, for each response function, we use Eq. (9) to build the second-order error-estimate models. Then, we pose the error-constrained sub-problem as follows:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \hat{f}^{(k)}(\boldsymbol{x}) \\ \text{s.t.} \quad & \hat{\boldsymbol{g}}^{(k)}(\boldsymbol{x}) = \boldsymbol{0} \\ & \hat{\boldsymbol{h}}^{(k)}(\boldsymbol{x}) \leq \boldsymbol{0} \\ & \tilde{E}_f^{(k)} \leq \min\left(\tau_{\text{abs},f}, \left|f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)\right| \tau_{\text{rel},f}\right) \\ & \tilde{E}_{\boldsymbol{g},i}^{(k)} \leq \min\left(\tau_{\text{abs},\boldsymbol{g},i}, \left|g_{\text{hi},i}\left(\boldsymbol{x}^{(k)}\right)\right| \tau_{\text{rel},\boldsymbol{g},i}\right), \\ & \qquad\qquad \forall i = 1, 2, \ldots, m_{\boldsymbol{g}} \\ & \tilde{E}_{\boldsymbol{h},i}^{(k)} \leq \min\left(\tau_{\text{abs},\boldsymbol{h},i}, \left|h_{\text{hi},i}\left(\boldsymbol{x}^{(k)}\right)\right| \tau_{\text{rel},\boldsymbol{h},i}\right), \\ & \qquad\qquad \forall i = 1, 2, \ldots, m_{\boldsymbol{h}}, \end{aligned} \tag{11}$$

where the $\hat{f}^{(k)}$, $\hat{\boldsymbol{g}}^{(k)}$, and $\hat{\boldsymbol{h}}^{(k)}$ functions indicate the use of calibrated models as defined in Eq. (1).

As we want this model-management framework to be usable with off-the-shelf optimization algorithms, we cannot solve Eq. (11) as written, since there will likely be cases where the error constraints are incompatible with the "true" constraints, $\hat{\boldsymbol{g}}^{(k)}$ and $\hat{\boldsymbol{h}}^{(k)}$, creating an infeasible problem. While we could simply increase the error tolerances to the point where the constraints are feasible, that would defeat the purpose of bounding the steps based on the estimated error. Luckily, this is a known problem for trust-region methods (Nocedal and Wright 1999), and we

can rely on techniques developed for such methods. We take an approach based on the Sequential ℓ_1 Quadratic Programming (Sℓ_1QP) method described in Chapter 18.5 of Nocedal and Wright (1999). We first move the calibrated constraints into the objective, as an ℓ_1 penalty term. Then, we reformulate the non-smooth ℓ_1 penalty term as an "elastic" program by introducing the slack variables $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^{m_g}$, and $\boldsymbol{t} \in \mathbb{R}^{m_h}$. This results in the following sub-problem:

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{v},\boldsymbol{w},\boldsymbol{t}} \quad & \hat{f}^{(k)}(\boldsymbol{x}) + \mu\left(\sum_i^{m_g}\left(v_i + w_i\right) + \sum_i^{m_h} t_i\right) \\
\text{s.t.} \quad & v_i, w_i \geq 0, \quad \forall i = 1, 2, \ldots, m_{\boldsymbol{g}} \\
& t_i \geq 0, \quad \forall i = 1, 2, \ldots, m_{\boldsymbol{h}} \\
& \hat{g}_i^{(k)}(\boldsymbol{x}) = v_i - w_i, \quad \forall i = 1, 2, \ldots, m_{\boldsymbol{g}} \\
& \hat{h}_i^{(k)}(\boldsymbol{x}) \leq t_i, \quad \forall i = 1, 2, \ldots, m_{\boldsymbol{h}} \\
& \tilde{E}_f^{(k)} \leq \min\left(\tau_{\text{abs},f}, \left|f_{\text{hi}}\left(\boldsymbol{x}^{(k)}\right)\right| \tau_{\text{rel},f}\right) \\
& \tilde{E}_{\boldsymbol{g},i}^{(k)} \leq \min\left(\tau_{\text{abs},\boldsymbol{g},i}, \left|g_{\text{hi},i}\left(\boldsymbol{x}^{(k)}\right)\right| \tau_{\text{rel},\boldsymbol{g},i}\right), \\
& \qquad\qquad \forall i = 1, 2, \ldots, m_{\boldsymbol{g}} \\
& \tilde{E}_{\boldsymbol{h},i}^{(k)} \leq \min\left(\tau_{\text{abs},\boldsymbol{h},i}, \left|h_{\text{hi},i}\left(\boldsymbol{x}^{(k)}\right)\right| \tau_{\text{rel},\boldsymbol{h},i}\right), \\
& \qquad\qquad \forall i = 1, 2, \ldots, m_{\boldsymbol{h}}.
\end{aligned}
\tag{12}
$$

The constraints in the sub-problem defined in Problem (12) are always compatible (Nocedal and Wright 1999).

The penalty parameter μ must be chosen carefully to balance the competing goals of improving the objective and ensuring feasibility. We base our scheme that updates μ on Algorithm 18.5 given in Nocedal and Wright (1999) and describe it here. During each sub-problem iteration, after Problem (12) is solved, if the slack variables $\boldsymbol{v}$, $\boldsymbol{w}$, and $\boldsymbol{t}$ are all less than ϵ_{feas}, then μ is deemed acceptable and will be used again in the next iteration. If, instead, the values of the slacks are non-zero, we may need to increase the value of the penalty. We define $\boldsymbol{m}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^{m_g + m_h}$ as the constraint violation at the design specified by $\boldsymbol{x}$. To determine how much to increase μ, we solve another optimization problem that minimizes the ℓ_1 norm of $\boldsymbol{m}(\boldsymbol{x})$. The solution to this optimization problem represents the maximum reduction in infeasibility that could be achieved inside the error-estimate bounds. If the maximum achievable reduction in infeasibility is at least 1% larger than the actual reduction in infeasibility, then we increase the penalty parameter by a factor of 1.5.

The values used for each error bound, $\tau_{\text{abs}}^{(k)}$ and $\tau_{\text{rel}}^{(k)}$, are free to vary from each sub-optimization to the next as needed. We note, however, that we have not found constant maintenance of these bounds to be needed, compared to the updates needed to the trust radius in a trust-region-based optimization. As the error-estimate constraints are based on the estimated level of correlation between the low- and high-fidelity models, the actual design variable bound can be thought of as sizing itself. Still, the development of a scheme to algorithmically vary these bounds is an avenue for future research and may yield additional efficiency gains as it could allow the optimization algorithm to further exploit trends in the low-fidelity model without being overly conservative. In the results presented in Sect. 4, we adopt a scheme such that the first iteration has $\tau_{\text{abs}}^{(0)} = \tau_{\text{rel}}^{(0)} = \infty$, allowing the low-fidelity trends to be fully explored by the sub-problem optimizer. We use the user-specified values for τ_{abs} and τ_{rel} at each subsequent iteration.

Once the optimization problem defined in Problem (12) is solved, the next iteration of the optimization scheme begins again with the calibration of the low-fidelity model at the previous sub-problem's optimum. We use the high-fidelity optimality and feasibility to measure overall convergence. We need the values of the Lagrange multipliers to be able to compute optimality. If the multipliers are not provided by the sub-problem optimizer, as is the case for many off-the-shelf optimizers, we estimate them by solving a least-squares problem. As we know that the norm of the gradient of the sub-problem Lagrangian will be close to zero at the sub-problem optimum, we can estimate the values of the Lagrange multipliers $\boldsymbol{\pi}^{(k)}$ at the k-th iteration by solving

$$
\begin{aligned}
\boldsymbol{\pi}^{(k)} = \arg\min_{\boldsymbol{\pi}} &= \left\| \nabla_x \hat{\mathcal{L}}(\boldsymbol{x}^*, \boldsymbol{\pi}) \right\|_2^2 \\
&= \left\| \nabla \hat{f}(\boldsymbol{x}^*) - \hat{\mathsf{A}}^{\mathrm{T}} \boldsymbol{\pi} \right\|_2^2,
\end{aligned}
\tag{13}
$$

where $\boldsymbol{x}^*$ is the optimal solution to the k-th sub-problem, and $\hat{\mathsf{A}}$ is the Jacobian of the sub-problem's active constraints. Once we have the (estimated) Lagrange multipliers, we compute the high-fidelity optimality as follows:

$$
O = \left\| \nabla_x \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\pi}) \right\|_\infty = \left\| \nabla f(\boldsymbol{x}^*) - \mathsf{A}^{\mathrm{T}} \boldsymbol{\pi} \right\|_\infty. \tag{14}
$$

We compute the high-fidelity feasibility as follows:

$$
F = \left\| \boldsymbol{m}(\boldsymbol{x}^*) \right\|_\infty, \tag{15}
$$

where again, $\boldsymbol{m}(\boldsymbol{x}^*)$ computes the vector of constraint violations. The iterations terminate when the optimality and feasibility are below their user-specified tolerances. The optimization procedure is illustrated graphically in the flow chart in Fig. 3.

4 Optimization examples

This section presents the results from numerical experiments that serve to validate the E^{2}M^2 framework. The framework is first demonstrated on a one-dimensional analytical optimization problem that illustrates how the algorithm works and how the quality of the low-fidelity model affects the

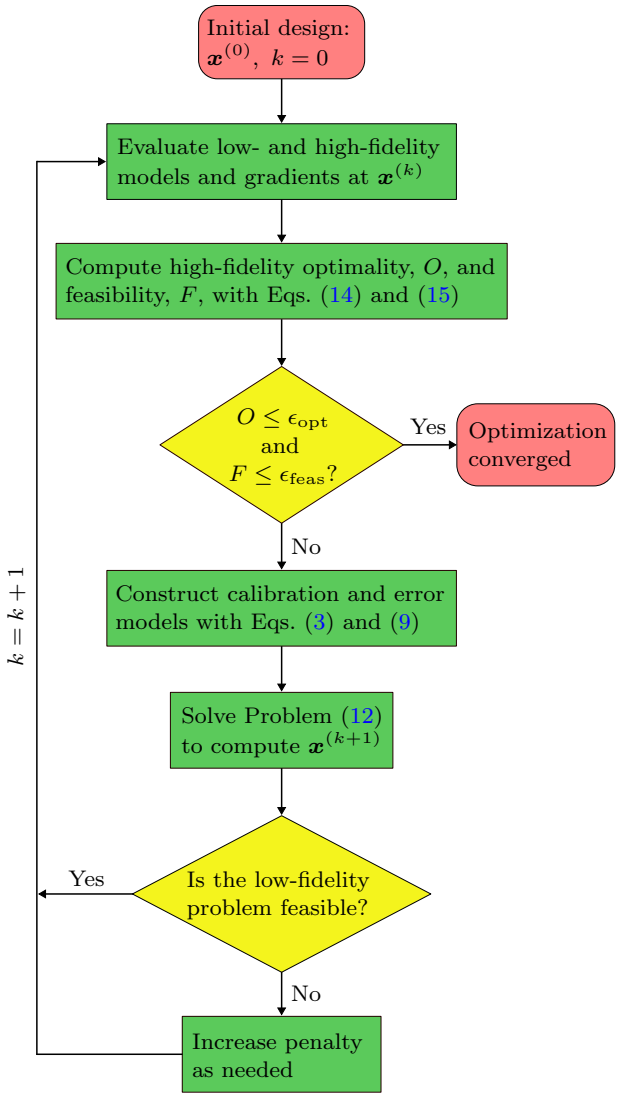


Fig. 3 This flow chart illustrates the major components of the E^2M^2 optimization framework at a high level

optimization. We then compare the E^2M^2 algorithm against state-of-the-art multi-fidelity optimization methods on a series of common analytical benchmark problems. Finally, we demonstrate it on a realistic electric-motor optimization problem.

We converge the multi-fidelity optimizations to a high-fidelity optimality tolerance of 10^{-4} and a feasibility tolerance of 10^{-6} for all problems. We use SNOPT (Gill et al. 2002, 2005) version 7.7.1 with an optimality tolerance of 10^{-6} and a feasibility tolerance of 10^{-6} to solve each calibrated low-fidelity sub-optimization. We also use SNOPT with an optimality tolerance of 10^{-4} and a feasibility tolerance of 10^{-6} for the direct high-fidelity optimizations used for comparison. We interface with SNOPT using OpenMDAO (Gray et al. 2019) with the PyOptSparse (Wu et al. 2020) optimization driver.

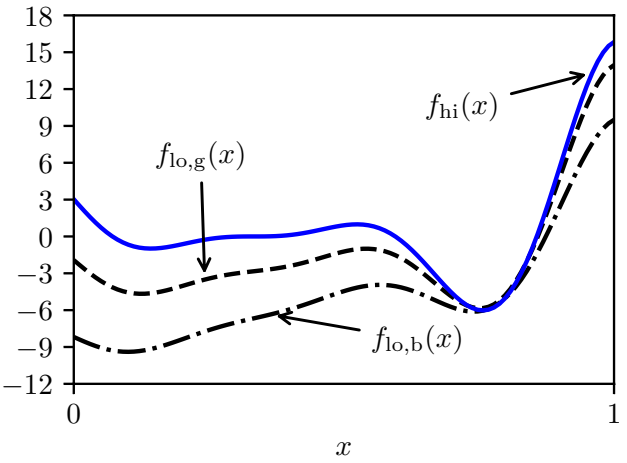


Fig. 4 The plot of the 1D models shows how the different low-fidelity models capture the trends of the high-fidelity model over the domain

4.1 Forrester problem

We first present an application of the E^2M^2 algorithm on a simple 1D analytical problem that demonstrates how the efficacy of the framework depends on the quality of the low-fidelity model.

4.1.1 Problem setup

We consider the simple 1D problem described by Forrester et al. (2007). Thus, the high-fidelity model is

$$f_{\text{hi}}(x) = (6x - 2)^2 \sin(12x - 4), \quad x \in [0, 1]. \tag{16}$$

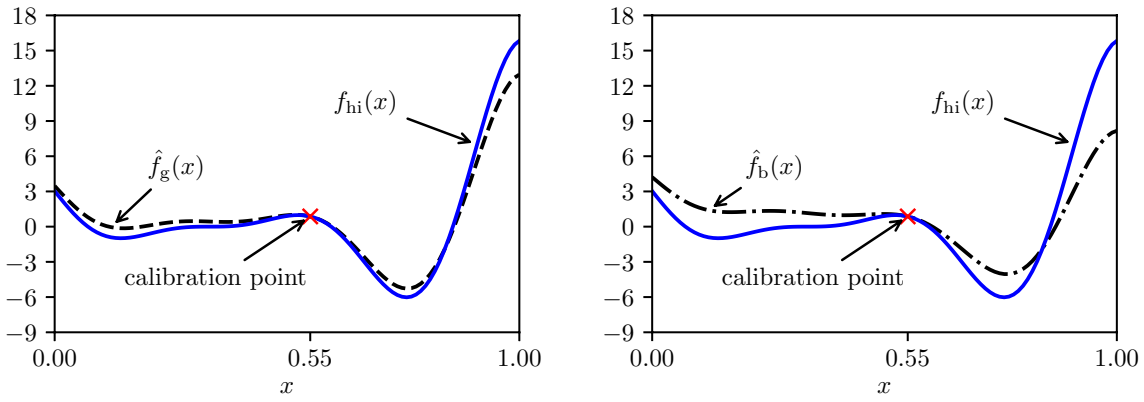
We consider two different low-fidelity models to demonstrate how the efficacy of the multi-fidelity optimization framework depends on the correlation between the low- and high-fidelity models. The first model, considered the "good" model, is given as follows:

$$f_{\text{lo,g}}(x) = 0.85 f_{\text{hi}}(x) + 5(x - 0.5) - 2, \quad x \in [0, 1], \tag{17}$$

while the "bad" low-fidelity model is given as follows:

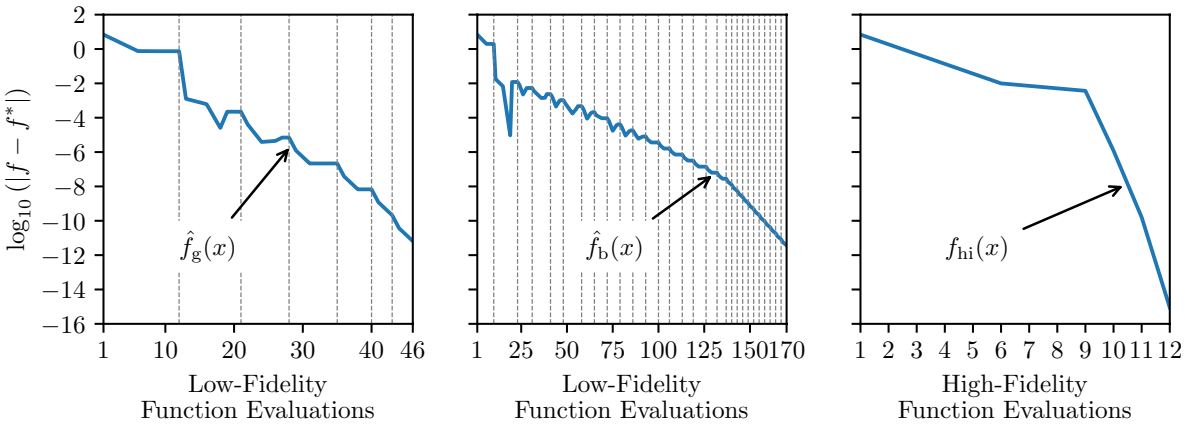
$$f_{\text{lo,b}}(x) = 0.6 f_{\text{hi}}(x) + 10(x - 0.5) - 5, \quad x \in [0, 1]. \tag{18}$$

The 1D low- and high-fidelity models are plotted in Fig. 4.



(a) Good correlation (b) Poor correlation

Fig. 5 The calibrated 1D models illustrate the effect of calibrating the gradient in addition to the function value



(a) "Good" multi-fidelity optimization (b) "Bad" multi-fidelity optimization (c) Direct high-fidelity optimization

Fig. 6 The log difference between the true optimum and the objective function history clearly illustrates the convergence of the 1D multi-fidelity and direct high-fidelity optimizations. The vertical dashed lines in the multi-fidelity plots indicate when the low-fidelity model was re-calibrated

We wish to solve the bound-constrained minimization of Eq. (16), stated as follows:

$$\min_x \quad f_{\text{hi}}(x), \quad \text{s.t.} \quad 0 \le x \le 1, \tag{19}$$

with objective error bounds of $\tau_{\text{abs},f} = \infty$ and $\tau_{\text{rel},f} = 0.1$ using the multi-fidelity optimization algorithm described in Sect. 3. Guidance for selecting values for τ_{abs} and τ_{rel} will be discussed shortly. We approximate the Hessian difference, needed to build the error estimates, using finite differences of the gradients. For a simple 1D example problem like this, we do not expect the multi-fidelity approach to significantly outperform a conventional optimizer such as SNOPT. Nevertheless, this problem is useful to illustrate how the E^{2}M^2 algorithm progresses during an optimization and to highlight potential issues.

4.1.2 Multi-fidelity optimization

We first consider the "good" low-fidelity model, given by Eq. (17), initialized at $x^{(0)} = 0.55$. The high-fidelity model f_{hi} and the low-fidelity model calibrated about $x^{(0)} = 0.55$ are plotted in Fig. 5a. The objective function history is plotted against each low-fidelity model evaluation used in the subproblem optimizations in Fig. 6a.

The optimization converged to the calibrated low-fidelity optimum given by $x^* = 0.7572$ and $f_{\text{hi}}(x^*) = -6.0207$. The optimization evaluated the calibrated low-fidelity objective function and gradient 46 times. Additionally, it required 8 high-fidelity function and gradient evaluations to calibrate the low-fidelity model, and 8 additional gradient evaluations to approximate the Hessian difference needed by the error estimates.

Next, we consider the "bad" low-fidelity model given in Eq. (18), again started from $x^{(0)} = 0.55$. The high-fidelity model f_{hi} and the low-fidelity model calibrated about $x^{(0)} = 0.55$ are plotted in Fig. 5b. The sub-optimization objective function history is plotted for each low-fidelity model evaluation in Fig. 6b.

This optimization converged to the calibrated low-fidelity optimum given by $x^* = 0.7572$ and $f_{\text{hi}}(x^*) = -6.0207$. The optimization evaluated the calibrated low-fidelity objective function and gradient 170 times. Additionally, it required 30 high-fidelity function and gradient evaluations to calibrate the low-fidelity model, and 30 additional gradient evaluations to approximate the Hessian difference needed by the error estimates.

4.1.3 Direct high-fidelity optimization

For comparison, we perform a direct high-fidelity optimization of Eq. (16). Again, starting from $x^{(0)} = 0.55$, the objective function history is plotted in Fig. 6c. This optimization used 12 high-fidelity function and gradient evaluations, and converged to $x^* = 0.7572$ and $f_{\text{hi}}(x^*) = -6.0207$.

As illustrated by Fig. 6a and b, the behavior and efficacy of the multi-fidelity optimization largely depends on the correlation between the low- and high-fidelity models. While the "good" low-fidelity model only took 8 high-fidelity function and gradient evaluations, the "bad" model took 30 high-fidelity function and gradient evaluations, more than the stand-alone high-fidelity optimization. Thus, the true efficiency improvements realizable with the E^2M^2 framework are problem specific, depending on the cost of the low-fidelity model relative to the high-fidelity model, and the quality with which the low-fidelity model approximates the high-fidelity model. However, the results presented in this test case suggest that the framework can produce optimized designs quite efficiently, provided that the low-fidelity model is relatively inexpensive and captures the high-fidelity model well.

4.2 Benchmark problems

In this section, we investigate the impact of the chosen values for τ_{abs} and τ_{rel}. In addition, we compare the E^2M^2 framework to state-of-the-art multi-fidelity optimization approaches on a series of common analytical benchmark problems.

The first analytical problem we consider is the 1D "Double-well Potential" model described by Foumani et al. (2023). For this problem, the high-fidelity model is

Table 1 Borehole design variables

Variable			Description
$0.05 \leq$	r_{w}	≤ 0.15	Radius of borehole (m)
$100 \leq$	r	$\leq 10,000$	Radius of influence (m)
$100 \leq$	T_{u}	$\leq 12,000$	Transmissivity of upper aquifer (m^2/year)
$990 \leq$	H_{u}	≤ 1110	Potentiometric head of upper aquifer (m)
$10 \leq$	T_{l}	≤ 500	Transmissivity of lower aquifer (m^2/year)
$700 \leq$	H_{l}	≤ 820	Potentiometric head of lower aquifer (m)
$1000 \leq$	L	≤ 1000	Length of borehole (m)
$6000 \leq$	K_{w}	$\leq 12,000$	Hydraulic conductivity of borehole (m/year)

$$f_{\text{hi}}(x) = 0.6x^4 - 0.3x^3 - 3x^2 + 2x, \; x \in [-2.5, 3], \tag{20}$$

and the low-fidelity model is given as follows:

$$f_{\text{lo}}(x) = 0.6x^4 - 0.3x^3 - 3x^2 - 1.2x, \; x \in [-2.5, 3]. \tag{21}$$

The next problem we consider is the 8-D "Borehole" model described by Morris et al. (1993) that characterizes the flow of water through a borehole drilled between two aquifers. The high-fidelity model is given as

$$f_{\text{hi}}(r_{\text{w}}, r, T_{\text{u}}, T_{\text{l}}, H_{\text{u}}, H_{\text{l}}, L, K_{\text{w}}) = \frac{2\pi T_{\text{u}}(H_{\text{u}} - H_{\text{l}})}{\log\left(\frac{r}{r_{\text{w}}}\right)\left(1 + \frac{2LT_{\text{u}}}{\log\left(\frac{r}{r_{\text{w}}}\right) r_{\text{w}}^2 K_{\text{w}}} + \frac{T_{\text{u}}}{T_{\text{l}}}\right)}. \tag{22}$$

We use the low-fidelity model from Foumani et al. (2023), which for this problem is

$$f_{\text{lo}}(r_{\text{w}}, r, T_{\text{u}}, T_{\text{l}}, H_{\text{u}}, H_{\text{l}}, L, K_{\text{w}}) = \frac{2\pi T_{\text{u}}(1.05\, H_{\text{u}} - H_{\text{l}})}{\log\left(\frac{2r}{r_{\text{w}}}\right)\left(1 + \frac{3LT_{\text{u}}}{\log\left(\frac{r}{r_{\text{w}}}\right) r_{\text{w}}^2 K_{\text{w}}} + \frac{T_{\text{u}}}{T_{\text{l}}}\right)}. \tag{23}$$

The design variable descriptions and bounds are given in Table 1.

Finally, we consider the 10D "Wing" model described by Forrester et al. (2008) that computes a conceptual-level estimate for the weight of a small aircraft wing. The high-fidelity model is given as follows:

$$\begin{aligned} f_{\text{hi}}(S_{\text{w}}, W_{\text{fw}}, A, \Lambda, q, \lambda, t_{\text{c}}, N_{\text{z}}, W_{\text{dg}}, W_{\text{p}}) = \\ 0.036 S_{\text{w}}^{0.758} W_{\text{fw}} \left(\frac{A}{\cos^2 \Lambda}\right)^{0.6} q^{0.006} \lambda^{0.04} \\ \left(\frac{100 t_{\text{c}}}{\cos \Lambda}\right)^{-0.3} (N_{\text{z}} W_{\text{dg}})^{0.49} + S_{\text{w}} W_{\text{p}}. \end{aligned} \tag{24}$$

Again, we use the low-fidelity model from Foumani et al. (2023), which for this problem is

Table 2 Wing design variables

Variable			Description
$150 \leq$	S_w	≤ 200	Wing area (ft^2)
$220 \leq$	W_{fw}	≤ 300	Weight of fuel in the wing (lb)
$6 \leq$	A	≤ 10	Aspect ratio
$-10 \leq$	Λ	≤ 10	Quarter-chord sweep (deg)
$16 \leq$	q	≤ 45	Dynamic pressure at cruise (lb/ft^2)
$0.5 \leq$	λ	≤ 1	Taper ratio
$0.08 \leq$	t_c	≤ 0.18	Aerofoil thickness to chord ratio
$2.5 \leq$	N_z	≤ 6	Ultimate load factor
$1700 \leq$	W_{dg}	≤ 2500	Flight design gross weight (lb)
$0.025 \leq$	W_p	≤ 0.08	Paint weight (lb/ft^2)

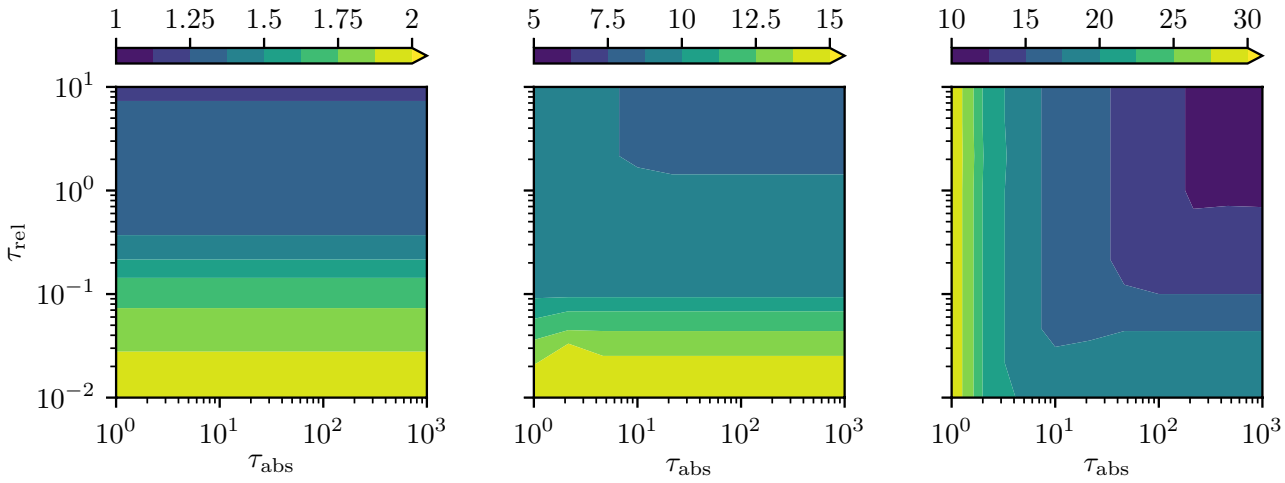
$$f_{lo}(S_w, W_{fw}, A, \Lambda, q, \lambda, t_c, N_z, W_{dg}, W_p) = 0.036 S_w^{0.9} W_{fw} \left(\frac{A}{\cos^2 \Lambda}\right)^{0.6} q^{0.006} \lambda^{0.04} \left(\frac{100 t_c}{\cos \Lambda}\right)^{-0.3} (N_z W_{dg})^{0.49}. \quad (25)$$

The design variable descriptions and bounds are given in Table 2.

4.2.1 Impact of τ_{abs} and τ_{rel}

We perform a series of optimizations of each of the analytical benchmark problems with the E^2M^2 algorithm, sweeping over different values of τ_{abs} and τ_{rel} to assess their impact on the performance of the optimization. We perform a "full-factorial" sweep, using uniformly log-spaced values of τ_{abs} between 10^0 and 10^3 and of τ_{rel} between 10^{-2} and 10^1. For each combination of τ_{abs} and τ_{rel}, we perform 10 optimizations, each started from a random initial design, and average the number of high-fidelity model evaluations required to obtain the optimum. The resulting heatmaps of average high-fidelity model evaluations are plotted in Fig. 7a–c for the Double-well Potential, Borehole, and Wing cases, respectively.

For the Double-well Potential model, we see that the optimizations took between one and two high-fidelity iterations to converge. This can be explained by observing that the difference between the low- and high-fidelity models [Eqs. (21) and (20)] is a linear term. Once the low-fidelity model has been calibrated, this linear term is corrected, and the calibrated model is identical to the high-fidelity model. Thus, the optimization performance depends solely on the size of the sub-optimizations' feasible space. We see that τ_{abs} has little impact, and that performance is determined solely by τ_{rel}, with larger values being more performant. While the Borehole and Wing models are not so trivially calibrated, we do observe a similar trend that performance degrades at lower values of τ_{abs}. In all cases, we observe the general trend that larger values of τ_{abs} and τ_{rel} tend to result in the fewest number of high-fidelity model evaluations needed.



(a) Double-well Potential (b) Borehole (c) Wing

Fig. 7 Heatmaps of the average number of high-fidelity model evaluations required during a multi-fidelity optimization with the E^2M^2 framework over a wide range of values for the parameters τ_{abs} and τ_{rel}

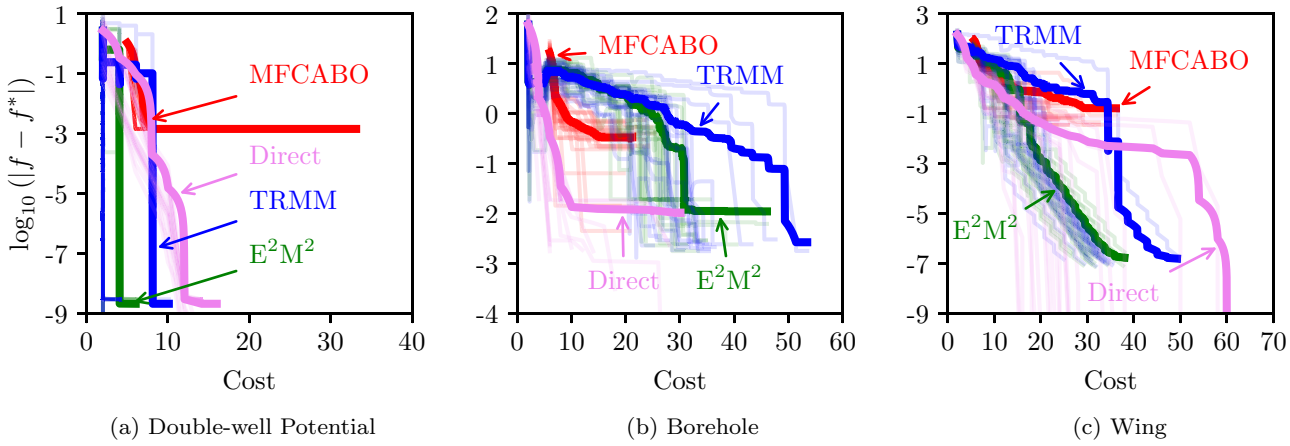


Fig. 8 The log difference between the true optimum and the objective function history for the three benchmark models studied. The thick lines indicate the average behavior for each algorithm over the 20 runs. Note that the units for cost are the number of equivalent high-fidelity model evaluations

4.2.2 Comparison to the state-of-the-art

We now compare the performance of the E²M² algorithm against a direct high-fidelity optimization and against existing state-of-the-art multi-fidelity optimization algorithms: the Multi-Fidelity Cost-Aware Bayesian Optimization (MFCABO) algorithm described by Foumani et al. (2023) and a TRMM implementation[1]. We use values of $\tau_{\text{abs}} = \infty$ and $\tau_{\text{rel}} = 1.0$ for all E²M² results.

For each benchmark problem, we run 20 optimizations, each started from a random initial design. We plot the objective function convergence history versus the cost of the optimization for each of the 20 runs in Fig. 8a–c for the Double-well Potential, Borehole, and Wing cases, respectively. For consistency, we measure optimization cost in the same manner as Foumani et al. (2023); we treat the high-fidelity model as 1000 times more expensive than the low-fidelity model. We make the assumption that both the low- and high-fidelity models use differentiated forward analyses based on either the reverse mode of algorithmic differentiation or the adjoint method. Consequently, the cost of a gradient evaluation is on the order of the forward model evaluation (Griewank and Walther 2008). Thus, we treat the cost of a gradient evaluation as the same as a model evaluation for both the low- and high-fidelity models.

Across all of the models, we see that the gradient-based optimization methods obtain a significantly more accurate optimal value than the MFCABO method. For the Double-well Potential model specifically, all three of the gradient-based methods converge more quickly than the MFCABO method, in addition to converging to a more accurate optimal value. The E²M² algorithm is the most efficient, followed by TRMM, and finally the direct high-fidelity optimization.

For the Borehole model, the direct high-fidelity optimization is the most efficient method. This implies that the low-fidelity model is particularly poor and is not worth using. This explains why the MFCABO method performs next-best (converging with less cost than both TRMM and E²M²), as its acquisition function safeguards against biased low-fidelity data (Foumani et al. 2023). However, despite MFCABO converging with less cost than TRMM and E²M², the latter converge much more tightly, with the E²M² algorithm again beating TRMM.

Finally, for the Wing model, the E²M² algorithm is again the most efficient method, followed by TRMM and the direct high-fidelity optimization. We argue that the speedup observed with E²M² compared to TRMM is due to the anisotropy in the feasible space defined by the error-estimate constraints; by allowing larger design steps in directions where the low-fidelity is estimated to be accurate, and by restricting the step size in directions where it is estimated to be inaccurate the E²M² algorithm is able to outperform the TRMM approach that uses an isotropic trust-region.

4.3 Electric-motor problem

This section presents the application of the E²M² framework on a realistic electric-motor optimization problem. We first describe the models used in the optimization and then present the results of the optimization studies.

[1] For all TRMM results, we use the trust-region update parameters $c_1 = 0.5$, $c_2 = 2.0$, $r_1 = 0.1$, $r_2 = 0.75$, $\Delta_0 = 10$, and $\Delta_\infty = 10^3\Delta_0$, as recommended by Alexandrov et al. (2001).

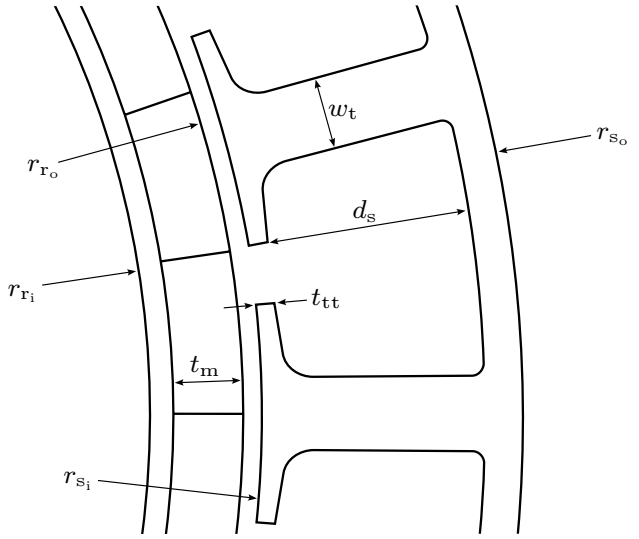


Fig. 9 Diagram showing how geometric design parameters define the geometry for the PMSM of interest

Table 3 Continuous motor geometric design parameters and their physical descriptions

Parameter	Description
l_s	Stack length
d_s	Slot depth
t_m	Magnet thickness
r_{r_i}	Rotor inner radius
r_{r_o}	Rotor outer radius
r_{s_i}	Stator inner radius
r_{s_o}	Stator outer radius
w_t	Tooth width
t_{tt}	Tooth tip thickness

Table 4 Remaining motor design parameters and their physical descriptions

Parameter	Description	Classification
r_s	Strand radius	Continuous
i	RMS current	Continuous
S	Rotor RPM	Continuous
n_t	Number of winding turns	Discrete
n_p	Number of magnetic poles	Discrete
n_s	Number of stator slots	Discrete

4.3.1 Motor parameterization

We demonstrate our multi-fidelity optimization framework by studying the commonly used three-phase radial-flux inrunner permanent magnet synchronous motor (PMSM). We characterize the geometry of the PMSM with the continuous parameters listed in Table 3 and illustrated in Fig. 9. Note that the stack length measures the "out-of-the-page" axial depth of the motor and is thus not shown in Fig. 9.

We define the PMSM by the set of parameters listed in Table 4 in addition to the geometric parameters listed in Table 3 and briefly describe them here. In a PMSM, a round wire with radius r_s is wrapped around each stator tooth n_t times for each of the motor's electrical phases. Each of these wires has an alternating current (AC) with root-mean-squared (RMS) value i flowing through it. The speed of the rotor rotation, given in rotations per minute (RPM), is directly related to the motor's electrical frequency f_e as $S = \frac{60}{n_p} f_e$, where n_p is the number of magnetic poles on the rotor.

Moreover, the selection of several discrete parameters are required to close the design of a PMSM, which are also listed in Table 4. The number of magnetic poles and stator slots are two discrete parameters that can dramatically influence the optimal PMSM design. Further, material choices for each component can significantly impact the performance of a PMSM. As we are targeting gradient-based optimization for our multi-fidelity optimization framework, we cannot directly consider these discrete parameters in an optimization. This is not a tremendous issue in practice, however, since electric-motor design theory guides such parameter selection (Hanselman 2003).

4.3.2 Computational model

The following section describes the details of the computational model used for the electric-motor analysis. In particular, we explain the geometry representation, the equations governing the electromagnetic analysis, and the methodology used to compute the outputs of interest. The section concludes with a brief discussion of the adjoint-based sensitivity analysis.

Geometry representation We use the open-source Engineering Sketch Pad (ESP) (Haimes and Dannenhoffer 2013) parametric CAD system to computationally represent the motor geometry in our model using the design parameters listed in Table 3. We use the EGADS Tessellator (Haimes and Drela 2012) through the CAPS (Haimes et al. 2016) interface to generate the finite-element mesh on the ESP CAD model needed by the electromagnetic analysis. Finally, we use the EGADS tesselation APIs (Haimes and Drela 2012) to explicitly map the geometric design parameters to the mesh node coordinates of the a priori generated finite-element mesh. We use $\boldsymbol{x}^h$ to denote the mesh node coordinates.

Electromagnetic field model We use the magnetostatic approximation of Maxwell's equations to model the

electromagnetic field inside the PMSM, given in differential form as

$$\nabla \times \boldsymbol{H} = \boldsymbol{J}_{\text{src}}, \qquad \forall\, \boldsymbol{x} \in \Omega_E, \tag{26}$$

$$\nabla \cdot \boldsymbol{B} = 0, \qquad \forall\, \boldsymbol{x} \in \Omega_E, \tag{27}$$

where $\boldsymbol{H}$ is the magnetic field intensity, $\boldsymbol{J}_{\text{src}}$ is the applied current density, $\boldsymbol{B}$ is the magnetic flux density, and Ω_E is the computational domain of the electromagnetic analysis. Here we take Ω_E to be a two-dimensional cross-section of the motor. Equations (26) and (27) are known as Ampère's circuital law, and Gauss's law for magnetism, respectively. Boundary conditions are required for Eqs. (26) and (27) to define a well-posed boundary value problem; these will be discussed shortly.

The magnetic field intensity, $\boldsymbol{H}$, and the magnetic flux density, $\boldsymbol{B}$, are related through the following constitutive equation:

$$\boldsymbol{H} = \nu(\boldsymbol{B})(\boldsymbol{B} - \boldsymbol{M}), \tag{28}$$

where $\nu(\boldsymbol{B})$ is the reluctivity, and $\boldsymbol{M}$ is the magnetic source created by permanent magnets. In general, $\nu(\boldsymbol{B})$ is a material-dependant nonlinear function of the magnetic flux density. We discuss the implementation details of our reluctivity model in Appendix.

We use the magnetic vector potential $\boldsymbol{A}$, which satisfies

$$\boldsymbol{B} = \nabla \times \boldsymbol{A}, \tag{29}$$

to ensure that $\nabla \cdot \boldsymbol{B} = \nabla \cdot \nabla \times \boldsymbol{A} = 0$ is satisfied by construction. Equation (29) is insufficient to define $\boldsymbol{A}$ uniquely, as the gradient of any scalar function may be added to $\boldsymbol{A}$ without changing $\boldsymbol{B}$. To address this, we impose the Coulomb gauge condition $\nabla \cdot \boldsymbol{A} = 0$ on $\boldsymbol{A}$.

Using this gauge condition, the magnetic vector potential from Eq. (29), the constitutive relationship Eq. (28), and by restricting the $\boldsymbol{B}$ field to be two-dimensional, Eq. (26) can be re-written as the following nonlinear scalar diffusion equation for the z-component of $\boldsymbol{A}$:

$$-\nabla \cdot \left(\nu \nabla A_z\right) - \left[\nabla \times (\nu \boldsymbol{M})\right]_z - J_{\text{src}_z} = 0, \forall\, \boldsymbol{x} \in \Omega_E. \tag{30}$$

Here, J_{src_z} is a piecewise-continuous source holding the current density in each of the phases of the motor. We implement the Dirichlet condition $A_z = 0$ along the entire boundary of Ω_E to make Eq. (30) well posed. This is equivalent to enforcing $\boldsymbol{B} \cdot \boldsymbol{n} = 0$ on the entire boundary of Ω_E; that is, there is no flux fringing along the boundary.

We discretize Eq. (30) with the finite-element method by leveraging the Modular Finite Element Methods (MFEM) (Kolev 2020; Anderson et al. 2021) library. This results in the following algebraic form for the analysis:

$$\boldsymbol{R}_A = \boldsymbol{R}_A(\boldsymbol{A}^h, \boldsymbol{x}^h, \boldsymbol{J}) = \boldsymbol{0}, \tag{31}$$

where $\boldsymbol{A}^h$ is the vector of finite-element degrees of freedom and $\boldsymbol{x}^h$ is the vector of mesh node coordinates. The vector $\boldsymbol{J} \in \mathbb{R}^p$ holds the z-axis-aligned current density J_{src_z} for each of the p phases in the motor. To capture the behavior of the motor at different points in time, we solve Eq. (31) multiple times at different rotor positions. This will be discussed in more detail shortly.

We solve Eq. (31) using Newton's method with absolute and relative convergence tolerances of 10^{-6}. We use a backtracking line search during Newton iterations that minimizes an interpolated quadratic or cubic approximation to $\|\boldsymbol{R}_A\|_2$ to ensure that $\|\boldsymbol{R}_A\|_2$ decreases with each step [see, for example, Chapter 4.3.3 of Martins and Ning (2021)]. Each Newton update is computed using the preconditioned conjugate gradient (PCG) method with an algebraic multigrid (AMG) preconditioner from the *hypre* library (Falgout and Yang 2002; Henson and Yang 2002). We use absolute and relative tolerances of 10^{-12} to measure convergence while solving the linear Newton updates and use default settings for the AMG preconditioner in *hypre* version 2.25.0.

Electromagnetic outputs Once Eq. (31) has been solved, we can compute the torque created by the motor and the various loss terms that result in reduced motor efficiency. We compute the torque on the rotor created by the magnetic field using Coulomb's virtual work method (Coulomb 1983; Coulomb and Meunier 1984). We calculate losses caused by direct-current (DC) and alternating-current (AC) flowing in the motor's windings, which are known as copper losses, and losses caused by hysteresis and eddy-current effects in the motor's magnetic steel, which are known as core losses.

To calculate the DC losses, we first compute the length of the conductor windings l_{w} in the motor as

$$l_{\text{w}} = 2n_{\text{s}}n_{\text{t}}\left(l_{\text{s}} + \pi\left(\frac{w_{\text{t}}}{2} + \frac{\pi\left(2r_{\text{s}_{\text{i}}} + d_{\text{s}} + t_{\text{tt}}\right)}{4n_{\text{s}}}\right)\right) + \frac{\pi\left(2r_{\text{s}_{\text{i}}} + d_{\text{s}} + t_{\text{tt}}\right)}{2}, \tag{32}$$

where n_{s} is the number of stator slots, w_{t} is the width of a stator tooth, $r_{\text{s}_{\text{i}}}$ is the stator inner radius, d_{s} is the slot depth, and t_{tt} is the tooth tip thickness. The first term of the length calculation accounts for wrapping a conductor around each tooth n_{t} times, while the second accounts for the length of the end windings connecting each group of teeth. Then, we calculate the DC resistance R_{DC} of the windings as

$$R_{\text{DC}} = \frac{l_{\text{w}}}{\sigma \pi r_{\text{s}}^2}, \tag{33}$$

where $\sigma = 58.14 \times 10^6 \frac{1}{\Omega \text{m}}$ is the electrical conductivity of the copper windings, and r_{s} is the radius of the conductor

winding. With the DC resistance computed, we calculate the DC power loss as

$$P_{\mathrm{DC}} = i^2 R_{\mathrm{DC}}, \tag{34}$$

where i is the RMS value of the AC current in the conductor.

The remaining loss terms that we incorporate in the electric-motor model are the result of time-dependent phenomena. As our underlying physical model is based on a static approximation of Maxwell's equations, we cannot model these terms directly. Instead, we rely on a combination of analytical and empirical relations to model these losses.

We use a hybrid method to calculate the AC losses that is based on the method presented by Fatemi et al. (2019). This hybrid approach uses the magnetic field computed from the finite-element analysis as part of an analytical equation for the AC loss in a single conductor. The analytical AC losses induced in a single round conductor in an externally oscillating magnetic field can be estimated as (Sullivan 2001)

$$P_{\mathrm{AC}_{\mathrm{strand}}} = l \frac{\pi r^4 \sigma \left(\omega B_{\mathrm{pk}}\right)^2}{8}, \tag{35}$$

where l is the conductor length exposed to the oscillating magnetic field, r is the conductor radius, σ is the electrical conductivity, ω is the frequency of oscillation, and B_{pk} is the peak value of the magnitude of the oscillating magnetic flux density. When applying Eq. (35) to a motor, we take l to be the stack length l_{s}, r to be the strand radius r_{s}, $\sigma = 58.14 \times 10^6 \frac{1}{\Omega \mathrm{m}}$ to be the electrical conductivity of copper, and ω to be the angular electrical frequency, related to the motor's RPM S as $\omega = \frac{\pi}{30} n_{\mathrm{p}} S$.

Equation (35) requires the peak (maximum in time) value of the oscillating magnetic flux density field. Therefore, we use multiple solutions of Eq. (31) at different rotor positions to capture the behavior of the magnetic field in time. Using these multiple field solutions, we use the discrete induced-exponential smooth max function from Kennedy and Hicken (2015) to calculate an estimate for the peak (in time) magnetic flux density field at each point in space.[2] Finally, with this peak magnetic flux density field, we integrate Eq. (35) over the winding area and scale by the total number of wire strands to calculate the final AC loss estimate.

We use the empirically derived Steinmetz equation (1892) to compute the core losses in the motor's components. For each component in the motor, the core losses are given as

$$P_{\mathrm{C}} = K_{\mathrm{s}} f_{\mathrm{e}}^{\alpha} B_{\mathrm{pk}}^{\beta} m, \tag{36}$$

where f_{e} is the electrical excitation frequency, B_{pk} is the peak value of the magnitude of the magnetic flux density in the component, m is the mass of the component, and the coefficient K_{s} and exponents α and β are empirically fit material-dependent parameters. For the results presented in this work, we use the values $K_{\mathrm{s}} = 0.0044$, $\alpha = 1.286$, and $\beta = 1.76835$. We use the same procedure described in the AC loss calculation to calculate the B_{pk} field in the stator and rotor. Once the B_{pk} field is obtained, we estimate its spatial maximum value across a component using the induced-exponential smooth max function presented in Kennedy and Hicken (2015).

Analytical derivatives We supply the optimizer with analytical derivatives, where possible, to improve the computational efficiency of the optimization. We use algorithmic differentiation to compute the partial derivatives of all of the electromagnetic outputs. Then, we use a combination of algorithmic differentiation and the adjoint method to compute derivatives of the implicit state calculation. Unfortunately, we are unable to compute exact analytical derivatives of the geometry representation, so we rely on forward finite differences with step size $\delta = 10^{-6}$ to compute partial derivatives through the ESP CAD system. Once we have computed the partial derivatives for each component of the analysis, we rely on OpenMDAO (Gray et al. 2019) to solve for the required total derivatives using the unified derivatives equations (UDE) (Martins and Hwang 2013; Hwang and Martins 2018).

4.3.3 Problem setup

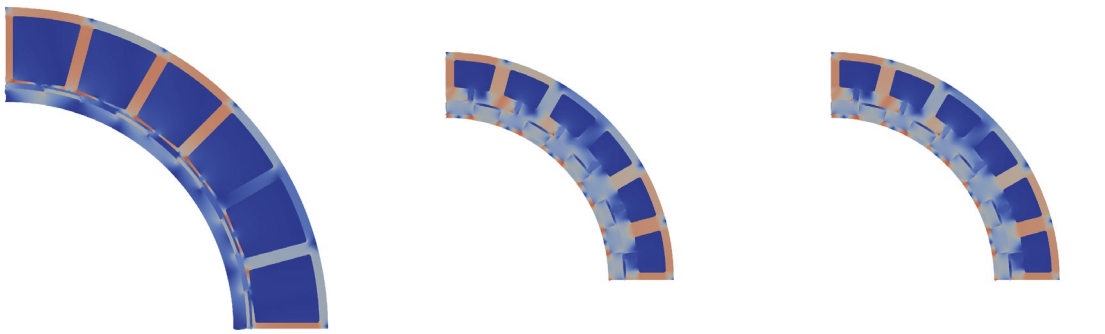
In this section, we describe the problem we will use to demonstrate the $\mathrm{E}^2\mathrm{M}^2$ framework on a realistic electric-motor optimization problem. The objective of the optimization is to maximize the efficiency of an electric motor by varying the motor geometry, input current, and winding strand radius, subject to output power and geometric constraints. Table 5 provides a summary of the optimization problem statement. We use the Symmetric Rank 1 quasi-Newton update formula [see, for example, Chapter 6.2 of Nocedal and Wright (1999)] to approximate the Hessian differences needed by the error estimates. The quasi-Newton updates are computed during the model calibration process.

We use a coarse mesh finite element analysis with linear Lagrange basis functions for the low-fidelity model, and a fine mesh model with quadratic Lagrange basis functions for the high-fidelity model. The low-fidelity model has a total of 17,716 finite-element degrees of freedom and takes approximately 15 s to evaluate the model and 20 s to compute its gradient. The high-fidelity model has a total of 1,193,920 finite-element degrees of freedom and takes approximately 15 min to evaluate and an additional 5 min to compute its gradient. Note that for the low-fidelity model, the gradient computation time is dominated by the cost of

[2] Specifically, at each finite-element degree of freedom.

Table 5 Electric-motor optimization problem statement

	Function/variable			Description
Maximize		η		Motor efficiency
With respect to	0 mm <	l_s		Stack length
	1 mm ≤	d_s		Slot depth
	1 mm ≤	t_m	≤ 5 mm	Magnet thickness
		r_{r_i}		Rotor inner radius
		r_{r_o}		Rotor outer radius
		r_{s_i}		Stator inner radius
	100.0 mm ≤	r_{s_o}		Stator outer radius
	2.5 mm ≤	w_t		Tooth width
	0.75 mm ≤	t_{tt}		Tooth tip thickness
	0.1 mm ≤	r_s	≤ 0.32 mm	Strand radius
	0 A <	i		Strand RMS current
Subject to	$P_{out} = 13$ kW			Output power
	$\frac{i}{\pi r_s^2} \geq 10 \frac{\text{A}}{\text{mm}^2}$			Strand current density
	$t_{ry} \equiv r_{r_o} - r_{r_i} \geq 1$ mm			Rotor yoke thickness
	$t_{sy} \equiv r_{s_o} - r_{s_i} - d_s \geq 1$ mm			Stator yoke thickness
	$t_g \equiv r_{s_i} - r_{r_o} - t_m = 1$ mm			Air-gap thickness



(a) Feasible initial design (b) Multi-fidelity optimized design (c) Direct high-fidelity optimized design

Fig. 10 The magnitude of the magnetic flux density in the different motor geometries. Note that while only a quarter of the geometry is shown, the full motor was simulated

finite-differencing the ESP CAD system, while for the high-fidelity analysis, the adjoint solves dominate. We compute the B_{pk} field needed for the AC and core loss computations by evaluating Eq. (31) at two different rotor positions for the low-fidelity analysis, and four rotor positions for the high-fidelity analysis, corresponding to $\theta_e = \left(0, \frac{\pi}{2}\right)^T$, and $\theta_e = \left(0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right)^T$ for the low- and high-fidelity analyses respectively.

We start each electric-motor optimization from a feasible but non-optimal design, with geometry and magnetic flux density field illustrated in Fig. 10a. The initial design variables and outputs are listed in Table 6, and the remaining fixed parameters are listed in Table 7.

4.3.4 Multi-fidelity optimization

We first consider the multi-fidelity electric-motor optimization where the output power and efficiency are calibrated. We start the optimization from the feasible design with design variables given in Table 6, and use error bounds of $\tau_{abs,\eta} = \infty$, $\tau_{rel,\eta} = 0.1$, $\tau_{abs,P_{out}} = \infty$, and $\tau_{rel,P_{out}} = 0.1$ for the efficiency and output power, respectively. The optimization procedure raised the motor's efficiency from the initial 91.45% to the optimized value of 98.26%. The history of the motor efficiency at each low-fidelity model evaluation is plotted in Fig. 11a.

The multi-fidelity optimization converged to the optimal design vector given in Table 6, with the optimized geometry

Table 6 Electric-motor design vectors and outputs for the initial design, as well as the multi-fidelity and direct high-fidelity optimized designs

	Initial value		Multi-fidelity		High-fidelity	
η	91.4511	%	98.2624	%	98.2624	%
l_s	12.2862	mm	12.9020	mm	12.9086	mm
d_s	28.5715	mm	12.1054	mm	12.0896	mm
t_m	1.5006	mm	5.0	mm	5.0	mm
r_{r_i}	99.5549	mm	71.1470	mm	71.1746	mm
r_{r_o}	106.8851	mm	77.9643	mm	77.9934	mm
r_{s_i}	107.8851	mm	78.9643	mm	78.9934	mm
r_{s_o}	141.0950	mm	100.0	mm	100.0	mm
w_t	5.5316	mm	6.5331	mm	6.5397	mm
t_{tt}	0.8212	mm	5.5133	mm	5.4966	mm
r_s	0.2425	mm	0.1	mm	0.1	mm
i	2.0755	A	0.3142	A	0.3142	A
P_{out}	13.0000	kW	13.0000	kW	13.0000	kW
$\frac{i}{\pi r_s^2}$	11.2323	$\frac{A}{mm^2}$	10.0	$\frac{A}{mm^2}$	10.0	$\frac{A}{mm^2}$
t_{ry}	5.8296	mm	1.8174	mm	1.8189	mm
t_{sy}	3.8172	mm	3.4169	mm	3.4204	mm
t_g	1.0	mm	1.0	mm	1.0	mm

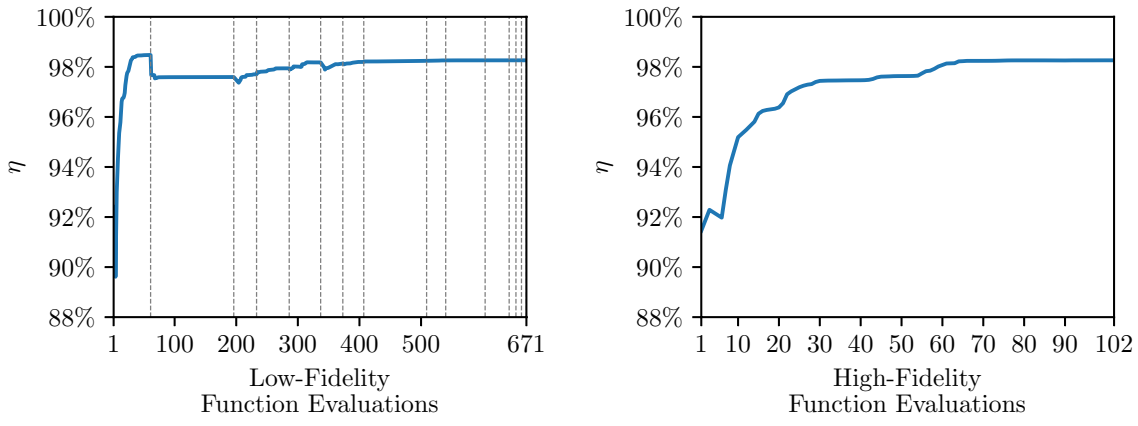
Table 7 Electric-motor optimization fixed parameters

Parameter	Value
n_t	840
S	6000 RPM
n_p	20
n_s	24

shown in Fig. 10b. Note that Fig. 10 shows each motor geometry at the same scale, illustrating that the optimized motors are physically smaller than the initial design. The procedure evaluated the calibrated low-fidelity model and its gradient 617 times. Further, it required 14 high-fidelity model and gradient evaluations to calibrate the low-fidelity model. The SR1 Hessian difference updates were computed with the already computed gradients, requiring no additional cost. In total, the multi-fidelity electric-motor optimization took 12 h and 21 min.

4.3.5 Direct high-fidelity optimization

Finally, for comparison, we perform a direct optimization of the high-fidelity model. Starting from the initial design shown in Fig. 10a, the optimization raised the motor's efficiency from the initial value of 91.45% to the optimized



(a) Multi-fidelity optimization (b) Direct high-fidelity optimization

Fig. 11 The objective function history illustrated the convergence of the multi-fidelity and direct high-fidelity electric-motor optimizations. The vertical dashed lines in the multi-fidelity optimization plot indicate when the low-fidelity model was re-calibrated

value of 98.26%. The objective function history is plotted in Fig. 11b. This direct high-fidelity optimization used 102 high-fidelity model and gradient evaluations and took 28 h and 4 min to complete. The final optimized geometry is illustrated in Fig. 10c, and the optimized design variables are given in Table 6.

The results from the multi-fidelity electric-motor optimization show that the proposed method can be significantly more efficient than a stand-alone high-fidelity optimization. The multi-fidelity optimization was able to compute the same high-fidelity optimized design in less than half the time compared to the direct high-fidelity optimization. This gain in optimization efficiency is largely due to the ability of the first-order calibrated low-fidelity model to accurately capture the physics of the high-fidelity model, at a fraction of its cost.

Table 8 Control points and knot vector for the cubic B-spline fit used in the Hiperco 50 reluctivity calculation

Control points	Knots
5.5286	0.0
5.4645	0.0
4.5597	0.0
4.2891	0.0
3.8445	0.1479
4.2880	0.5757
4.9505	0.9924
11.9364	1.4090
11.9738	1.8257
12.6554	2.2424
12.8097	2.6590
13.3347	3.0757
13.5871	3.4924
13.5871	3.9114
13.5871	8.0039
	10.0
	10.0
	10.0
	10.0

5 Conclusions

This paper has presented a novel multi-fidelity model-management framework based on error estimates between the calibrated low- and high-fidelity models. This framework uses a specified error tolerance between the low- and high-fidelity models to globalize the optimization, avoiding the need for a practitioner to specify non-intuitive parameters as needed by the commonly employed multi-fidelity trust-region methods. Additionally, due to the anisotropy introduced by defining the trust region in terms of the estimated low-fidelity error, the framework is able to take larger design steps in directions where the calibrated model is estimated to be accurate, and smaller steps in directions where it is estimated to be less accurate, ultimately leading to a speedup compared to classical TRMM-based methods.

We have compared our proposed error-estimate-based multi-fidelity optimization framework to state-of-the-art algorithms and found it to perform favorably on a series of benchmark problems. The results presented here show that the proposed E^2M^2 framework can quite efficiently produce high-fidelity optima provided the low-fidelity model accurately correlates with the high-fidelity model. However, should the low-fidelity model not accurately capture the trends of the high-fidelity model, the presented framework can be less efficient than a direct high-fidelity optimization. A limitation of our proposed method is that it cannot directly optimize over discrete inputs; a limitation shared by all gradient-based optimization algorithms. However, for problems with mixed continuous-discrete variables, the efficiency afforded by the E^2M^2 algorithm when optimizing over the continuous variables should enable an efficient optimization over the discrete parameters, using e.g., a "Branch and Bound"-type algorithm [see, for example, Chapter 8 of Martins and Ning (2021)].

There is further potential to improve the E^2M^2 algorithm and make it even more performant. Linearizing the error-estimate constraints would reduce the computational cost associated with solving the low-fidelity sub-optimizations, and further enable the application to large-scale problems. Additionally, an extension to the framework to recursively apply the E^2M^2 algorithm to solve the low-fidelity sub-optimizations would allow the consideration of arbitrary levels of fidelity and likely provide further acceleration. We plan to investigate these avenues in future work.

Appendix: Reluctivity model

We model the reluctivity, $\nu(\boldsymbol{B})$, as a piecewise-continuous function where each sub-function of ν is based on the material it is in. We use constant values for the reluctivity in the motor's air-gap, magnets and windings. For the air-gap and motor windings, this function takes the value of the reluctivity of free space $\nu_0 = \frac{1}{\mu_0} = \frac{1}{4\pi\times10^{-7}}$. For the magnets, we use the constant value $\nu_{\text{mag}} = \frac{1}{\mu_r\mu_0}$, where μ_r is the magnet's relative permeability, a material-dependent value listed in a material data-sheets. We take $\mu_r = 1.04$ for the $Nd_2Fe_{14}B$ magnets considered in this work.

The reluctivity of the magnetic steel used in the motor's stator and rotor is a nonlinear function of the magnetic flux density. We use the model given as follows:

$$\nu_{\text{Fe}} = \exp\left(f(\|\boldsymbol{B}\|)\right),$$

where $f(\|\boldsymbol{B}\|)$ is a cubic B-spline that represents the log-transformed reluctivity as a function of the magnitude of the magnetic flux density. The B-spline knot vector and control points are found by minimizing the least-squares error between the spline and discrete B-ν data points. The control points and knot vector for the Hiperco 50 magnetic steel used for the results presented in this work are listed in Table 8.

Acknowledgements The authors gratefully acknowledge support for the reported research from the National Aeronautics and Space Administration under the Fellowship Award 20-0091. We graciously thank the authors of Foumani et al. (2023) for providing a data file containing their results to allow easy comparison.

Funding This study was supported by Glenn Research Center (Grant No. 20-Fellows'20-0091).

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Replication of results The code used to implement the E^2M^2 algorithm and produce the results presented in Sect. 4.1 can be found on GitHub https://github.com/tuckerbabcock/E2M2.

References

Alexandrov NM, Dennis JE, Lewis R, Torczon V (1998) A trust-region framework for managing the use of approximation models in optimization. Struct Optim 15(1):16–23

Alexandrov NM, Lewis R, Gumbert CR, Green LL, Newman PA (2001) Approximation and model management in aerodynamic optimization with variable-fidelity models. J Aircr 38(6):1093–1101

Anderson R, Andrej J, Barker A, Bramwell J, Camier J-S, Cerveny J, Dobrev V, Dudouit Y, Fisher A, Kolev T, Pazner W, Stowell M, Tomov V, Akkerman I, Dahm J, Medina D, Zampini S (2021) MFEM: a modular finite element methods library. Comput Math Appl 81:42–74. https://doi.org/10.1016/j.camwa.2020.06.009

Bryson DE, Rumpfkeil MP (2018) Multifidelity quasi-newton method for design optimization. AIAA J 56(10):4074–4086

Conn Andrew R, Gould Nicholas IM, Toint Philippe L (2000) Trust region methods. SIAM, Philadelphia

Coulomb J (1983) A methodology for the determination of global electromechanical quantities from a finite element analysis and its application to the evaluation of magnetic forces, torques and stiffness. IEEE Trans Magn 19(6):2514–2519. https://doi.org/10.1109/TMAG.1983.1062812. ISSN 1941-0069.

Coulomb J, Meunier G (1984) Finite element implementation of virtual work principle for magnetic or electric force and torque computation. IEEE Trans Magn 20(5):1894–1896. https://doi.org/10.1109/TMAG.1984.1063232. ISSN 1941-0069

Elham A, van Tooren MJL (2017) Multi-fidelity wing aerostructural optimization using a trust region filter-SQP algorithm. Struct Multidisc Optim 55(5):1773–1786

Eriksson D, Jankowiak M (2021) High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In: Uncertainty in artificial intelligence. PMLR, pp 493–503

Eweis-Labolle JT, Oune N, Bostanabad R (2022) Data fusion with latent map Gaussian processes. J Mech Des 144(9):091703

Falgout RD, Yang UM (2002) *hypre*: a library of high performance preconditioners. In: Sloot PMA, Hoekstra AG, Kenneth Tan CJ, Dongarra JJ (eds) Computational science—ICCS 2002. Springer, Berlin/Heidelberg, pp 632–641. ISBN 978-3-540-47789-1

Fatemi A, Ionel DM, Demerdash NAO, Staton DA, Wrobel R, Chong YC (2019) Computationally efficient strand eddy current loss calculation in electric machines. IEEE Trans Ind Appl 55(4):3479–3489. https://doi.org/10.1109/TIA.2019.2903406. ISSN 1939-9367

Fernández-Godino MG, Park C, Kim N-H, Haftka RT (2016) Review of multi-fidelity models. arXiv Preprint. https://doi.org/10.48550/arXiv.1609.07196

Forrester A, Sóbester A, Keane A (2007) Multi-fidelity optimization via surrogate modelling. Proc R Soc A Math Phys Eng Sci 463(2088):3251–3269

Forrester A, Sóbester A, Keane A (2008) Engineering design via surrogate modelling: a practical guide. Wiley, Hoboken

Foumani ZZ, Shishehbor M, Yousefpour A, Bostanabad R (2023) Multi-fidelity cost-aware Bayesian optimization. Comput Methods Appl Mech Eng 407:115937

Gill PE, Murray W, Saunders MA (2002) SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM J Optim 12(4):979–1006. https://doi.org/10.1137/S1052623499350013

Gill PE, Murray W, Saunders MA (2005) SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM Rev 47(1):99–131. https://doi.org/10.1137/S0036144504446096

Gould NIM, Orban D, Sartenaer A, Toint PL (2005) Sensitivity of trust-region algorithms to their parameters. 4OR 3:227–241

Gratton S, Sartenaer A, Toint PL (2008) Recursive trust-region methods for multiscale nonlinear optimization. SIAM J Optim 19(1):414–444

Gray JS, Hwang JT, Martins JRRA, Moore KT, Naylor BA (2019) OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization. Struct Multidisc Optim 59(4):1075–1104. https://doi.org/10.1007/s00158-019-02211-z

Griewank A, Walther A (2008) Evaluating derivatives: principles and techniques of algorithmic differentiation. SIAM, Philadelphia

Haimes R, Dannenhoffer J (2013) The engineering sketch pad: a solid-modeling, feature-based, web-enabled system for building parametric geometry. In: 21st AIAA computational fluid dynamics conference. p 3073. https://doi.org/10.2514/6.2013-3073

Haimes R, Drela M (2012) On the construction of aircraft conceptual geometry for high-fidelity analysis and design. In: 50th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition. p 683. https://doi.org/10.2514/6.2012-683

Haimes R, Dannenhoffer J, Bhagat ND, Allison DL (2016) Multi-fidelity geometry-centric multi-disciplinary analysis for design. In: AIAA modeling and simulation technologies conference. p 4007. https://doi.org/10.2514/6.2016-4007

Hanselman D (2003) Brushless permanent magnet motor design. Magna Physics Publishing. ISBN 1881855155

Henson VE, Yang UM (2002) BoomerAMG: a parallel algebraic multigrid solver and preconditioner. Appl Numer Math 41(1):155–177. https://doi.org/10.1016/S0168-9274(01)00115-5. ISSN 0168-9274

Hwang JT, Martins JRRA (2018) A computational architecture for coupling heterogeneous numerical models and computing coupled

derivatives. ACM Trans Math Softw. https://doi.org/10.1145/3182393. ISSN 0098-3500
Jo Y, Choi S (2014) Variable-fidelity aerodynamic design using gradient-enhanced kriging surrogate model with regression. In: 52nd aerospace sciences meeting. p 0900
Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Glob Optim 13:455–492
Joseph H, van Bloemen Waanders B (2023) Hyper-differential sensitivity analysis with respect to model discrepancy: optimal solution updating. Comput Methods Appl Mech Eng 412:116082
Keane A (2003) Wing optimization using design of experiment, response surface, and data fusion methods. J Aircr 40(4):741–750
Kennedy GJ, Hicken JE (2015) Improved constraint-aggregation methods. Comput Methods Appl Mech Eng 289:332–354. https://doi.org/10.1016/j.cma.2015.02.017. ISSN 0045-7825
Kennedy MC, O'Hagan A (2001) Bayesian calibration of computer models. J R Stat Soc Ser B (Stat Methodol) 63(3):425–464. https://doi.org/10.1111/1467-9868.00294
Kolev TV (2020) Modular finite element methods. Computer Software. mfem.org
Lewis R (1996) A trust region framework for managing approximation models in engineering optimization. In: 6th symposium on multidisciplinary analysis and optimization. p 4101
March A, Willcox K (2012a) Constrained multifidelity optimization using model calibration. Struct Multidisc Optim 46(1):93–109
March A, Willcox K (2012b) Provably convergent multifidelity optimization algorithm not requiring high-fidelity derivatives. AIAA J 50(5):1079–1089
Martins JRRA, Hwang JT (2013) Review and unification of methods for computing derivatives of multidisciplinary computational models. AIAA J 51(11):2582–2599. https://doi.org/10.2514/1.J052184
Martins JRRA, Ning A (2021) Engineering design optimization. Cambridge University Press, Cambridge
Morris MD, Mitchell TJ, Ylvisaker D (1993) Bayesian design and analysis of computer experiments: use of derivatives in surface prediction. Technometrics 35(3):243–255
Nagawkar J, Ren J, Xiaosong D, Leifsson L, Koziel S (2021) Single- and multipoint aerodynamic shape optimization using multifidelity models and manifold mapping. J Aircr 58(3):591–608
Nocedal J, Wright SJ (1999) Numerical optimization. Springer, New York
Olivanti R, Gallard F, Brézillon J, Gourdain N (2019) Comparison of generic multi-fidelity approaches for bound-constrained nonlinear optimization applied to adjoint-based CFD applications. In: AIAA aviation 2019 forum. p 3102
Peherstorfer B, Willcox K, Gunzburger M (2018) Survey of multifidelity methods in uncertainty propagation, inference, and optimization. SIAM Rev 60(3):550–591
Réthoré P-E, Fuglsang P, Larsen GC, Buhl T, Larsen TJ, Madsen HA (2014) TOPFARM: multi-fidelity optimization of wind farms. Wind Energy 17(12):1797–1816
Shan S, Wang GG (2010) Metamodeling for high dimensional simulation-based design problems. J Mech Des 132(5):051009. https://doi.org/10.1115/1.4001597. ISSN 1050-0472
Shi R, Long T, Ye N, Yufei W, Wei Z, Liu Z (2021) Metamodel-based multidisciplinary design optimization methods for aerospace system. Astrodynamics 5:185–215
Simpson TW, Martins JRRA (2011) Multidisciplinary design optimization for complex engineered systems design: report from an NSF workshop. J Mech Des 133(10):101002. https://doi.org/10.1115/1.4004465
Steinmetz CP (1892) On the law of hysteresis (part II.) and other phenomena of the magnetic circuit. Trans Am Inst Electr Eng IX(1):619–758. https://doi.org/10.1109/T-AIEE.1892.5570469. ISSN 2330-9431
Sullivan CR (2001) Computationally efficient winding loss calculation with multiple windings, arbitrary waveforms, and two-dimensional or three-dimensional field geometry. IEEE Trans Power Electron 16(1):142–150
Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Special section on multidisciplinary design optimization: metamodeling in multidisciplinary design optimization: how far have we really come? AIAA J 52(4):670–690. https://doi.org/10.2514/1.J052375
Wu N, Kenway G, Mader CA, Jasa J, Martins JRRA (2020) pyOptSparse: a Python framework for large-scale constrained nonlinear optimization of sparse systems. J Open Source Softw 5(54):2564. https://doi.org/10.21105/joss.02564
Wu N, Mader CA, Martins JRRA (2022a) Large-scale multifidelity aerostructural optimization of a transport aircraft. In: 33rd congress of the international council of the aeronautical sciences
Wu N, Mader CA, Martins JRRA (2022b) A gradient-based sequential multifidelity approach to multidisciplinary design optimization. Struct Multidisc Optim 65:131–151. https://doi.org/10.1007/s00158-022-03204-1