Decision Making Under Uncertainty for a Digital Thread Enabled Design Process

Victor Singh*
Massachusetts Institute of Technology
77 Massachusetts Avenue 37-312
Cambridge, MA 02138
Email: victorsi@mit.edu

Karen E. Willcox
Professor of Aerospace Engineering and Engineering Mechanics
University of Texas at Austin
201 E 24th Street, POB 4.102
Austin, TX 78712
Email: kwillcox@oden.utexas.edu

ABSTRACT

Digital Thread is a data-driven architecture that links together information from all stages of the product lifecycle. Although it is finding increasing application in manufacturing, maintenance/operations, and design related tasks, a principled formulation of analyzing the decision making problem under uncertainty for the Digital Thread remains absent. The contribution of this paper is to present a formulation using Bayesian statistics and decision theory. First, we address how uncertainty propagates in the product lifecycle and how the Digital Thread evolves based on the decisions we make and the data we collect. Using these mechanics, we explore designing over multiple product generations or iterations and provide an algorithm to solve the underlying multistage decision problem. We illustrate our method on an example structural design problem where our method can quantify and optimize different types and sequences of decisions, ranging from experimentation, manufacturing, and sensor placement/selection, in order to minimize total accrued costs.

1 Introduction

Digital Thread is a data-driven architecture that links together information from all stages of the product lifecycle (e.g., early concept, design, manufacturing, operation, post-life, and retirement) for real-time querying and long-term decision making [1, 2]. Information contained in the Digital Thread may include sufficient representation (e.g., through numerical/categorical parameters, data structures, textual forms, etc.) of available resources, tools, methods, processes, as well as data collected across the product lifecycle. A desired target of Digital Thread is its use as the primary source from which other downstream information, such as that used for design, analysis, and maintenance/operations related tasks, can be derived [2, 3].

Though a significant challenge of the Digital Thread involves development of an efficient architecture and its associated processing of information, a relatively unexplored aspect of the Digital Thread is how to represent and understand the propagation of the uncertainty within the product lifecycle itself. High levels of uncertainty can lead to designs that are over-conservative or designs that may require expensive damage tolerance policies, redesigns, or retrofits if analysis cannot show sufficient component integrity. One way to reduce this uncertainty is to incorporate data-driven decisions that are informed from data collected throughout the design process.

To assess the benefits data-driven decisions can provide, incorporating the value of future information into the decision making process becomes critical. Performing this type of analysis opens up the possibility to assess not just the current product generation or iteration but also the ones to follow. From this, new ways of thinking about design emerge, such as how can

*Address all correspondence to this author.

strategic collection of data from the current generation be used to improve the design of the next? For example, we may start asking *what* data should be collected, *where* should the data be collected, and *when* should the data be collected to minimize overall accrued costs? This problem involves analyzing uncertainty and the data collected over sequential stages of decision making. In this paper, we showcase how our Digital Thread analysis methodology introduced in [4], and further developed in [5], analyzes this problem using Bayesian inference and decision theory, and solves it numerically using approximate dynamic programming.

To set the stage for discussion, we give a brief overview of the aspects of Digital Thread that are of relevance. Digital Thread can be seen as a synthesis and maturing of ideas from product lifecycle management (PLM), model-based engineering (MBE), and model-based systems engineering (MBSE). PLM is the combination of strategies, methods, tools, and processes to manage and control all aspects of the product lifecycle across multiple products [6]. These aspects might include integrating and communicating processes, data, and systems to various groups across the product lifecycle. A key enabler of efficient PLM has been the development and implementation of MBE where data models or domain models communicate design intent in order to avoid document-based exchange of information [7, 8], the latter which can result in lossy transfer of the original sources, as well as to eliminate redundant and inconsistent forms of data representation and transfer. Examples of MBE data models include use of mechanical/electronic computer aided design tools and modeling languages such as system modeling language (SysML), unified modeling language (UML) and extensible markup language (XML). MBSE applies the principles of MBE to support systems engineering requirements related to formalization of methods, tools, modeling languages, and best practices used in the design, analysis, communication, verification, and validation of large-scale and complex interdisciplinary systems throughout their lifecycles [8–11]. Many recent assessments and applications of these ideas in the context of the Digital Thread can be found in additive manufacturing [12], 3D printing and scanning [13], CNC machining [14], as well as detailed design representation, lifecycle evaluation, and maintenance/operations related tasks [15, 16].

With a functional Digital Thread in place, characterizing uncertainty and optimizing under it can be performed with statistical methods and techniques. For instance, relevant sources of uncertainty within the product lifecycle, such as design parameters, modeling error, and measurement noise, can be identified, characterized, and ultimately reduced using tools and methods from uncertainty quantification [17–19]. With a means of assessing uncertainty, optimization under uncertainty can be performed with stochastic-based design and optimization methods. For instance, minimizing probabilistically formulated cost metrics subject to constraints that will arise for the Digital Thread decision problem may involve utilizing either stochastic programming or robust optimization. In stochastic programming, uncertainty is represented with probabilistic models and optimization is performed on an objective statement with constraints involving some mean, variance, or other probabilistic criteria [20]. Alternatively, in robust optimization, the stochastic optimization problem is cast into a deterministic one through determining the maximum/minimum bounds of the sources of uncertainty and performing an optimization over the range of these bounds [21]. Additionally, if reliability and robustness at the system level is required, uncertainty-based multidisciplinary design optimization can be employed [22–24].

Of course, decision making using the Digital Thread is not a one time occurrence. Understanding the sequential nature of the multistage decision problem of the Digital Thread where one decision affects the next is critical for producing effective data-driven decisions. These decisions will have to be guided through some appropriate metric of assessing costs and benefits. This problem is explored in optimal experimental design where the objective is to determine experimental designs (in our case decisions) that are optimal with respect to some statistical criteria or utility function [25]. To assess the sequential nature of decision making for the Digital Thread in particular, sequential optimal experimental design can be employed where experiments (again, decisions in our case) are conducted in sequence, and the results of one experiment may affect the design of subsequent experiments [26].

Despite the range of development and growth of Digital Thread and its application to manufacturing, maintenance/operations, and design related tasks in various multidisciplinary settings, a principled formulation that considers the propagation of uncertainty in the product lifecycle in the context of the Digital Thread is sparse. Furthermore, a mathematically precise way of analyzing and optimizing sequential data-informed decisions as new information is added to the Digital Thread remains absent. To address these gaps, in this paper we show that 1) the Digital Thread can be considered as a state that can dynamically change based on the decisions we make and the data we collect. We then show that 2) the evolution of uncertainty within the product lifecycle can be described with a Bayesian filter that can be represented in the Digital Thread itself. After expressing the Digital Thread in this way, we show how 3) the evolution of the Digital Thread can be modeled as a dynamical system that can be controlled using a stochastic optimal control formulation expressed as a dynamic program where the objective is to minimize total accrued costs over multiple stages of decision making. Finally, we provide a 4) numerical algorithm to solve this dynamic program using approximate dynamic programming.

We illustrate our methodology through an example composite fiber-steered component design problem where the objective is to minimize total accrued costs over two design generations or iterations. In addition to evaluating design choices such as performing coupon level experiments to reduce uncertainty in materials as well as manufacturing and deploying a component to obtain operational data, effectiveness of data collection can be further tailored by determining where to place sensors (sensor placement) or selecting which sensors to use (sensor selection). The novelty in our approach is that these choices will be guided by the objective directly without the need for additional metrics or criteria.

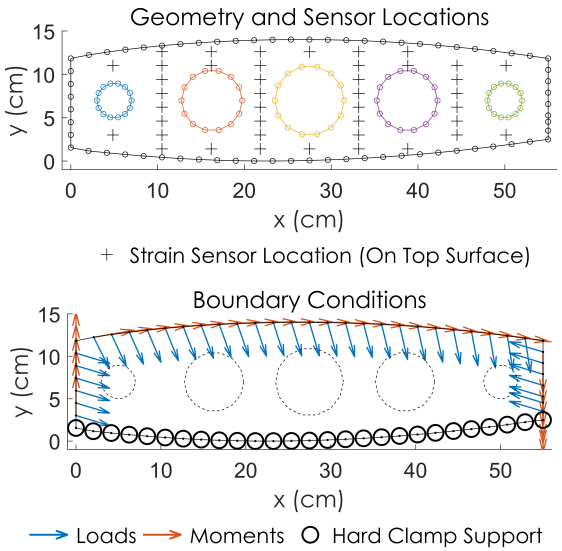

Fig. 1. Geometry, initial sensor locations, and boundary conditions for the design problem for one loading condition. Transverse shear is directed out of the page and is not shown for clarity.

The rest of this paper is organized as follows, Section 2 sets up the illustrative design problem through which our methodology will be described as well as lays out the mathematical machinery that describes the dynamical process underlying the Digital Thread. Section 3 details the decision problem for the Digital Thread enabled design process and presents the numerical algorithm to solve the decision problem. Section 4 presents results for the example design problem. And finally, Section 5 gives concluding remarks.

2 Design Problem Formulation

In this section, we formulate the design problem to be solved using our methodology. Subsection 2.1 describes the example design scenario through which we convey our methodology, Subsection 2.2 lays down the mathematical description of the problem, and Subsection 2.3 describes the underlying dynamical models that will be used for the overall decision making process.

2.1 Scenario Description

The design problem involves finding the optimal fiber angle and component thickness for a composite tow-steered (fiber-steered) planar (2-D) component subject to cost and constraint metrics. We consider specifically the design of a chord-wise rib within a wingbox section of a small fixed wing aircraft of wingspan around 15 meters, as shown in Fig. 1. The overall geometry has five holes of various radii with curved top and bottom edges.

A challenge to our design task is the presence of uncertain inputs that directly influence the design of the component. In this problem, the uncertain inputs are the loading the component will experience in operation, the material properties of the component, and the specific manufacturing timestamps. Situations where these variables have most relevance occur during the early stage of design when testing and experimentation have not yet taken place or when a brand new product is brought to market where only partial information can be used from other sources due to its novelty.

Large uncertainties in these inputs can lead to conservative designs that can be costly both to manufacture and operate. Thus, the goal is to collect data to reduce these uncertainties to the degree necessary to minimize overall costs. Data can be collected through three different lifecycle paths: material properties can be learned through collection of measurements from coupon level experiments; manufacturing timestamps can be learned from a combination of a bill of materials, timestamps of individual processes, and other related documentation when a prototype or product is manufactured; and operating loads can be learned from strain sensors placed on the component in operation.

Although the task of learning the uncertain input variables through measurements can be addressed with methods from machine learning, and more classically from solution methods for inverse problems, this task in the context of the overall design problem is made complicated by the fact that collecting data comes at a cost. To see this, we illustrate the Digital Thread for this design problem in Fig. 2. Here we see that collecting relevant data can require both time and financial resources. Though material properties data can be obtained fairly readily and quickly during the design phase through coupon level

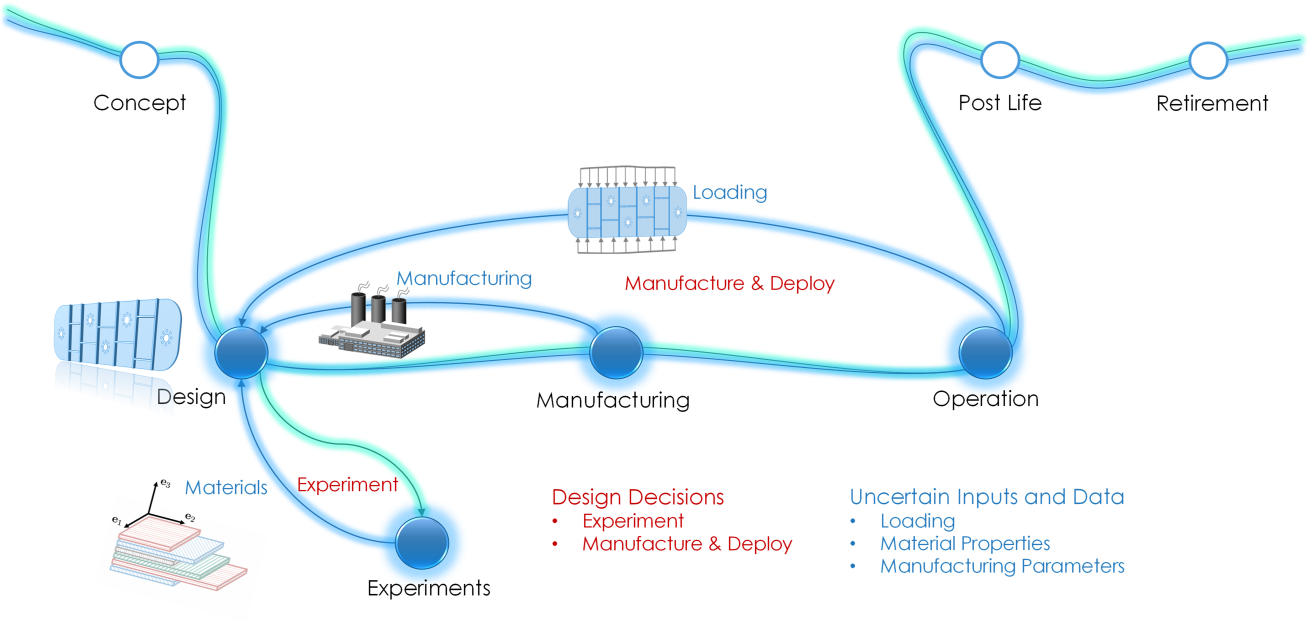

Fig. 2. Illustration of Digital Thread for the example design problem. The product lifecycle stages of interest here are between design and operation.

experiments, manufacturing data can only be obtained once a prototype is built. Additionally, operational data can only be obtained once a prototype or a full component is built, equipped with sensors, and put into operation. Depending on the scale of the component, the manufacturing process can take weeks or months, and putting a full component into operation with proper functionality of all its parts and sensor instrumentation may take much longer. Thus, making cost effective decisions that reduce uncertainty is critical for product reliability, reducing design process flow time, and minimizing total product expenses across its lifecycle.

2.2 Mathematical Formulation

The key elements of the design problem are broken down into five items: a notion of time or stage, the uncertain input variables (what we would like to learn), the measurement data (what we learn from), the Digital Thread itself (how to represent what we know), and the decision variables (the decisions and design choices we can make).

Time or Stage Time is modeled using non-dimensional increments that enumerate the sequence of decisions made or to be made up to some finite horizon $T \in \mathbb{N}$. It is expressed as $t \in \mathcal{T} = \{0, ..., T\}$. Physical time is allowed to vary, and will be the case when different decisions take shorter physical times to execute (e.g., performing coupon experiments) or longer physical times to execute (e.g., manufacturing and deploying a component).

Inputs The N_y inputs (the uncertain quantities to be learned) are described at each stage t as $y_t \in \mathbb{R}^{N_y}$. The variable y_t is composed of parameters of a finite discretization of the five integrated through thickness traction terms (in-plane loads per unit length, in-plane moments per unit length, and transverse shear per unit length) on the component boundary for a particular operating condition, material properties (material strengths), and parameters of a manufacturing process model to compute process times consisting of N_m total steps. The composite structural model is based on the small displacement Mindlin-Reissner plate formulation [27, 28] specialized for composites. For the manufacturing process model, we employ the manufacturing process and associated parameters detailed in the Advanced Composite Cost Estimating Manual (ACCEM) cost model [29, 30] that consists of $N_m = 56$ total steps for our problem.

Measurements The N_z measurements are described at each stage t as $z_t \in \mathbb{R}^{N_z}$. The variable z_t is composed of three strain sensor components for N_s sensor locations located on the top surface of the component, material properties data determined from coupon level experiments, and timestamps for the N_m total steps of the manufacturing process. Coupon level experiments here involve the static failure of composite test specimens of appropriate loading and geometry in order to acquire data about material properties and failure strengths used for structural analysis. Note, as illustrated in Fig. 2 the components of z_t are taken at different points along the product lifecycle (corresponding to coupon tests, manufacturing, and operation) and may not be fully populated at every stage t.

Decisions Decision making will encompass different strategies related to performing coupon tests and manufacturing and deploying a new design to reduce uncertainty while minimizing costs. Associated with manufacturing and deployment are additional specifications of fiber angle, component thickness, and sensor placement/selection. We designate a high-level decision as $u_t^d = \{E, D\}$ where $u_t^d = E$ will correspond to performing coupon tests and $u_t^d = D$ will correspond to manufacturing and deployment. For $u_t^d = D$, we designate the additional design specifications as $u_t = [u_t^p, u_t^s]^\top \in \mathbb{R}^{N_u}$, where $u_t^p \in \mathbb{R}^{N_p}$ specifies the geometrical parametrization of the component and $u_t^s \in [0,1]^{N_s}$ specifies the parametrization for sensor selection.

For this problem, u_t^p is composed of the coefficients of a finite dimensional parametrization of fiber angle and through thickness of the component body, as well as the spatial locations for all N_s sensors. For simplicity, we model ply angle as a continuous function of the component body and not model more detailed specifications such as individual ply and matrix compositions, additional layers consisting of different ply types, ply thicknesses, number of plies, and ply cutoffs at boundaries.

For sensor selection, we use a soft approach based on activation probabilities [31] to avoid the combinatorial problem associated with an exact binary (on/off) representation. Specifically, u_t^s gives probabilities where the i^{th} sensor is selected (active or on) with probability $(u_t^s)_i$ and not-selected (inactive or off) with probability $1-(u_t^s)_i$ during operation. To quantify the effective number of active sensors for this particular parametrization, we use an effective sensor utilization quantity defined as

$$e_t^s = \frac{1}{N_s}\sum_{i=1}^{N_s}(u_t^s)_i \tag{1}$$

Here, $e_t^s = 0$ when all sensors are always off/inactive and $e_t^s = 1$ when all sensors are always on/active.

Digital Thread The Digital Thread $\mathcal{D}_t \in \mathscr{I}$ at stage t reconciles the uncertain parts of the product lifecycle with its certain (or deterministically known) parts as

$$\mathcal{D}_t = (Q_t, R_t) \tag{2}$$

where R_t is the representation of the deterministically known parts of the product lifecycle at stage t that we will collectively call resources, Q_t is the representation of the uncertainty within the product lifecycle itself at stage t, and $\mathscr{I}$ is an information space over the product lifecycle encapsulating all possible uncertain and certain elements across all stages $t \in \mathcal{T}$. Provided a criterion of sufficiency is maintained [5], the Digital Thread can be represented in a number of equivalent ways. In this paper, the uncertainty within the product lifecycle is represented using $Q_t = p(y_t \,|\, I_t, u_t)$ that specifies the probability distribution of the uncertain inputs y_t given the history of collected data $I_t = \{R_0, u_0, ..., u_{t-1}, z_0, ..., z_{t-1}\}$ and the current decision to be made u_t. The resources R_t are represented using a multi-data type set that contains numerical, categorical, and/or character-like specifications of:

1. *Methods, Tools and Processes:* Enterprise level information and protocols of available methods, tools, and processes across the product lifecycle.
2. *Products:* Product specific design geometry, manufacturing process details, operational and data collection protocols, operation/maintenance/repair history, and lifecycle status.

2.3 Dynamical Process of the Digital Thread

With the design problem modeled, the dynamics of the Digital Thread can be described using the transition model

$$\mathcal{D}_{t+1} = \Phi_t(\mathcal{D}_t, u_t, z_t) \tag{3}$$

where $\Phi_t : \mathscr{I} \times \mathbb{R}^{N_u} \times \mathbb{R}^{N_z} \mapsto \mathscr{I}$ evolves the Digital Thread from stage t to $t+1$ given the decision u_t and measurements z_t at stage t. Within this transition model, $Q_t = p(y_t \,|\, I_t, u_t)$ is updated using the Bayesian filter

$$p(y_{t+1} \,|\, I_{t+1}, u_{t+1}) = \frac{1}{p(z_t \,|\, I_t, u_t)} \int_{\mathbb{R}^{N_y}} p(y_{t+1} \,|\, y_t, R_{t+1}, u_{t+1}) p(z_t \,|\, y_t, R_t, u_t) p(y_t \,|\, I_t, u_t)\, \mathrm{d}\nu \tag{4}$$

while the resources are updated according to

$$R_{t+1} = \Psi_t(R_t, u_t) \tag{5}$$

Here, the integration is performed over the uncertain inputs y_t and ν is the measure (or volume) over the uncertain inputs y_t. The function Ψ_t allows for changing resources (adding new elements, updating existing elements, or removing existing elements) at stage t. The Bayesian filter in Eqn. (4) models the process of data assimilation from stage t to stage $t+1$. First, the likelihood term $p(z_t \,|\, y_t, R_t, u_t)$ inside the integral represents the collection of new measurements after a decision is performed, followed by the updating of our knowledge of y_t after incorporating those measurements. Next, the term $p(y_{t+1} \,|\, y_t, R_{t+1}, u_{t+1})$ represents inheritance and modification of information carried over from a previous design into the next design (e.g., loads from a past airplane reused and modified for a new airplane with a longer fuselage). Details of the derivation of this Bayesian filter can be found in [5].

The Bayesian filter in Eqn. (4) can be computed using sequential Monte Carlo methods [32] or other linear/non-linear filtering methods where appropriate. For linear models with Gaussian uncertainty, for example, the Bayesian filter is a variant of the Kalman filter (with the prediction and analyze steps reversed) and can be computed analytically. The resources can be managed and updated through MBE, MBSE, and PLM related tools or software as well as through other data management techniques.

3 Decision Making Using the Digital Thread

In this section, we describe the decision making problem for the design problem. Subsection 3.1 describes the specific decisions of interest, Subsection 3.2 describes the mathematical optimization problem associated to those decisions, Subsection 3.3 describes the approximate dynamic programming technique we employ to solve the mathematical optimization, and finally Subsection 3.4 provides a numerical algorithm to implement the approximate dynamic programming technique.

3.1 Decisions for the Problem Scenario

For this problem scenario, we will produce two generations of a component where we are allowed to perform one set of coupon experiments. This will correspond to a three stage problem where $t \in \mathcal{T} = \{0,1,2\}$. In particular, we will be interested in the high-level decision sequences $\{u_0^d = E, u_1^d = D, u_2^d = D\}$ that we will denote as EDD and $\{u_0^d = D, u_1^d = E, u_2^d = D\}$ that we will denote as DED. For example, the sequence DED means to manufacture and deploy a new design first, followed by performing coupon level experiments second, and finally manufacturing and deploying another new design. The second design benefits from data collected from both coupon experiments and operational measurements of the previous design. These two sequences are distinguished by whether coupon experiments should be performed before any design is ever manufactured (and subsequently deployed) via the sequence EDD or right after the first design is manufactured and deployed via the sequence DED.

For each of these two sequences, we are interested in how subsequent data assimilation influences designs and costs of the component over the two generations. This will be explored through a greedy scenario that makes no use of future information, a sensor placement scenario, and a sensor selection scenario. Sensor placement and selection are not combined together for this problem setup in order to assess the performance of each (location vs. activity) independently.

3.2 Decision Statement for the Digital Thread

The decision statement for the Digital Thread enabled design process is given by the following Bellman equation:

$$\begin{aligned} V_t^*(\mathcal{D}_t) = \min_{u_t \in \mathbb{R}^{N_u}} \; & \mathbb{E}\big[r_t(\mathcal{D}_t, u_t, y_t) + \gamma V_{t+1}^*(\Phi_t(\mathcal{D}_t, u_t, z_t)) \,|\, \mathcal{D}_t, u_t\big] \\ \text{s.t. } & \mathbb{E}\big[g_t(\mathcal{D}_t, u_t, y_t) \,|\, \mathcal{D}_t, u_t\big] \leq 0, \quad t \in \mathcal{T}, \quad V_{T+1}^* = 0 \end{aligned} \tag{6}$$

Here, $V_t^* \colon \mathscr{I} \mapsto [0,\infty)$ is the optimal value function or cost-to-go at stage t, the parameter $\gamma \in [0,1]$ is a discount factor, and the symbol * denotes optimal quantities or functions. The solution to this Bellman equation yields an optimal policy $\pi_t^* = \{\mu_t^*, ..., \mu_T^*\}$ that defines a sequence of functions $\mu_t^*(\cdot) = u_t$ specifying new designs and changes to the Digital Thread for each stage t up to the horizon T. Each function μ_t^* of the optimal policy is a function of $\mathcal{D}_t$ (a feedback policy), i.e., $\pi_t^* = \{\mu_t^*(\mathcal{D}_t), ..., \mu_T^*(\mathcal{D}_T)\}$. The expectation is taken over the uncertain inputs $\{y_t, ..., y_T\}$ and measurements $\{z_t, ..., z_T\}$.

The functions $r_t : \mathscr{I} \times \mathbb{R}^{N_u} \times \mathbb{R}^{N_y} \mapsto [0,\infty)$ and $g_t : \mathscr{I} \times \mathbb{R}^{N_u} \times \mathbb{R}^{N_y} \mapsto \mathbb{R}^{N_g}$ denote the stage-wise cost and constraint functions, respectively, for the problem with N_g total constraints. For the stage-wise cost model, we employ a linear combination of the manufacturing cost model with cost functions that penalize aggressive fiber angle variation, aggressive component thickness variation, operational costs including strain sensor usage, as well as costs associated to coupon tests [5]. We do not penalize the placement of sensors for this particular setup. For the stage-wise constraint function during design stages, we use the Tsai-Wu failure criterion [33].

In some cases, the distribution of g_t may be heavily-tailed in which case the expected value of the constraint given in Eqn. (6) may not produce robust enough designs. In those cases, the expected value of the constraint can be replaced with an

appropriate measure of probabilistic risk, e.g., $\mathbb{P}[g_t(\mathcal{D}_t, u_t, y_t) \leq 0 \,|\, \mathcal{D}_t, u_t] > 1 - \varepsilon$ for some small $\varepsilon > 0$, or other criteria that can also take into consideration the severity of failure associated to the value of g_t [34].

For the greedy scenario, the Bellman equation is solved by setting $\gamma = 0$. Although data assimilation still occurs through evolution of the Digital Thread via the transition model from stage t to $t+1$, decisions determined from the Bellman equation at stage t with $\gamma = 0$ do not take into consideration the benefits nor costs of future data assimilation because the value of future information that comes through V_{t+1} as stage t is canceled out. For the sensor placement scenario, $\gamma = 1$ and the sensor locations are allowed to vary while the sensor selection probabilities are all fixed at one. For sensor selection, $\gamma = 1$ and the sensor selection probabilities are allowed to vary while the sensor locations are fixed. Note that structural tailoring for the sensor selection and sensor placement setups also takes place by design of the Bellman equation because the future value function V_{t+1} is a function of u_t. This dependency enables fiber angle and component thickness to also control the effectiveness of subsequent data collection in addition to sensor placement or sensor selection.

In total, there are six policies to compare: the greedy, sensor placement, and sensor selection scenarios for both the *EDD* and *DED* high-level decision sequences.

3.3 Solving the Decision Problem Using Approximate Dynamic Programming

We solve the decision problem by first rewriting the Bellman equation given in Eqn. (6) to produce the following equivalent, but notationally simpler statement:

$$\begin{aligned} V_t^*(\mathcal{D}_t) = \min_{u_t \in \mathbb{R}^{N_u}} \; & O_t(\mathcal{D}_t, u_t) + \gamma S_t^*(\mathcal{D}_t, u_t) \\ \text{s.t. } & G_t(\mathcal{D}_t, u_t) \leq 0, \quad t \in \mathcal{T}, \quad S_T^* = 0 \end{aligned} \tag{7}$$

where

$$\begin{aligned} O_t(\mathcal{D}_t, u_t) &= \mathbb{E}\big[r_t(\mathcal{D}_t, u_t, y_t) \,|\, \mathcal{D}_t, u_t\big] \\ G_t(\mathcal{D}_t, u_t) &= \mathbb{E}\big[g_t(\mathcal{D}_t, u_t, y_t) \,|\, \mathcal{D}_t, u_t\big] \\ S_t^*(\mathcal{D}_t, u_t) &= \mathbb{E}\big[V_{t+1}^*(\Phi_t(\mathcal{D}_t, u_t, z_t)) \,|\, \mathcal{D}_t, u_t\big] \end{aligned} \tag{8}$$

The function S_t^* is the expected value of the forward $t+1$ optimal value function, O_t is the expected value of the stage-wise cost function at stage t, and G_t is the expected value of the stage-wise constraint function at stage t. Next, we use a combination of Monte Carlo sampling with policy and function approximation [35] to solve the optimization problem numerically. The functions O_t and G_t can be computed directly using Monte Carlo sampling methods. The remaining terms that need to be determined, namely V_t^*, μ_t^*, and S_t^*, will be approximated and updated using policy and function approximation. However, in order to apply the methods of function approximation to our problem, V_t^*, μ_t^*, and S_t^* need to first have explicit parametrized forms. The parametrized forms we use are:

$$\begin{aligned} \mu_t(\mathcal{D}_t) &= A_t \phi_t(\mathcal{D}_t) + a_t \\ V_t(\mathcal{D}_t) &= \exp\{B_t \phi_t(\mathcal{D}_t) + b_t\} \\ S_t(\mathcal{D}_t, u_t) &= \exp\{C_t \varphi_t(\mathcal{D}_t, u_t) + c_t\} \end{aligned} \tag{9}$$

where $\phi_t : \mathscr{I} \mapsto \mathbb{R}^{M_p}$ is a vector of M_p basis functions at stage t, $A_t \in \mathbb{R}^{N_u \times M_p}$ is a matrix of basis coefficients for the policy function at stage t, $B_t \in \mathbb{R}^{1 \times M_p}$ is a matrix of basis coefficients for the value function at stage t, $\varphi_t : \mathscr{I} \times U \mapsto \mathbb{R}^{M_v}$ is a vector of M_v basis functions at stage t, and $C_t \in \mathbb{R}^{1 \times M_v}$ is a matrix of basis coefficients for the expected value of the forward value function at stage t. The variables $a_t \in \mathbb{R}^{N_u}$ and $b_t, c_t \in \mathbb{R}$ are used for setting initial conditions (e.g., initial component fiber angle, component thickness, sensor locations, etc.).

The symbol * is dropped because we are now approximating the optimal functions with imposed structure, which may lose optimality. In addition, the expression for V_t and S_t are exponentiated to ensure non-negativity of the value function. For the implementation, the basis functions ϕ_t and φ_t are constructed using radial basis functions (Gaussian radial basis functions in particular) with inputs appropriately scaled to lie within the bi-unit hypercube of appropriate dimension. These basis functions are not direct functions of the Digital Thread, but functions of numerical features of the Digital Thread such as mean, variances and/or or other statistical metrics of Q_t as well as other relevant numerical quantities from R_t

Next, A_t, B_t, and C_t are trained using a combination of least squares and approximate solutions of the Bellman equation. However, a direct application of least squares is challenging because we need to have a means of generating samples to train A_t, B_t, and C_t in the first place. We also don't know in advance how many samples are sufficient to yield good estimates of the policy and value functions so we would like to have flexibility of incorporating new samples without much re-calculation of the least squares formulas. Furthermore, performing the inverses in these least squares formulas can become computationally

expensive for large dimensions of A_t, B_t, and C_t. Finally, we would like to have an incremental update rule to the approximation of the policy where new samples are obtained through exploration using the latest approximation of the policy.

These issues can be addressed by utilizing a recursive update of the least squares formulas, known as Recursive Least Squares (RLS) [36]. Associated to the RLS formulation is a class of recursive approximate dynamic programming techniques given in [37] and [38] that we parallel. Details of the derivations for the RLS equations used for this paper can be found in [5]. The RLS equations take the form:

$$\begin{aligned} A_t^j &= A_t^{j-1} + \left(u_t^j - A_t^{j-1}\phi_t^j - a_t\right)[\phi_t^j]^\top [H_t^j]^{-1} \\ B_t^j &= B_t^{j-1} + \left(\log Q_t^j - B_t^{j-1}\phi_t^j - b_t\right)[\phi_t^j]^\top [H_t^j]^{-1} \\ C_t^j &= C_t^{j-1} + \left(\log V_{t+1}^j - C_t^{j-1}\varphi_t^j - c_t\right)[\varphi_t^j]^\top [J_t^j]^{-1} \end{aligned} \tag{10}$$

Here, j is the update index that increases by one when a new data point is added, $\phi_t^j = \phi_t(\mathcal{D}_t^j)$, and $\varphi_t^j = \varphi_t(\mathcal{D}_t^j, \hat{u}_t^j)$ at some Digital Thread $\mathcal{D}_t^j \in \mathscr{I}$ and decision $\hat{u}_t^j \in \mathbb{R}^{N_u}$. The terms $Q_t^j \in (0,\infty)$ and $u_t^j \in \mathbb{R}^{N_u}$ are determined from the minimization

$$\begin{aligned} Q_t^j = \min_{u_t^j \in \mathbb{R}^{Nu}} \; & O_t(\mathcal{D}_t^j, u_t^j) + \gamma \exp\left\{C_t^j \varphi_t^j + c_t\right\} \\ \text{s.t. } & G_t(\mathcal{D}_t^j, u_t^j) \le 0, \quad t \in \mathcal{T}, \quad S_T = 0 \end{aligned} \tag{11}$$

while V_{t+1}^j is given by

$$\begin{aligned} V_{t+1}^j &= \exp\left\{B_{t+1}^j \phi_{t+1}(\Phi_t(\mathcal{D}_t^j, \hat{u}_t^j, z_t^j)) + b_{t+1}\right\} \\ z_t^j &\sim p(z_t \,|\, I_t^j, \hat{u}_t^j) \end{aligned} \tag{12}$$

The decision $\hat{u}_t^j$ is generated by sampling in the neighborhood of the current iteration of the policy, or decision iterates during optimization of Eqn. (11). Note, this decision is different than u_t^j to allow additional flexibility on when and where to update S_t. This is because unlike μ_t and V_t, S_t is a function of *both* the Digital Thread and decision, and thus requires a different sampling approach to capture different values of the decision at a given state of the Digital Thread.

The symmetric matrix $H_t^j \in \mathbb{R}^{M_p \times M_p}$ at $j = 0$ is the regularization term in the original least squares formulas for A_t and B_t. It is updated using the Sherman–Morrison matrix identity:

$$[H_t^j]^{-1} = [H_t^{j-1}]^{-1} - \frac{[H_t^{j-1}]^{-1}\phi_t^j [\phi_t^j]^\top [H_t^{j-1}]^{-\top}}{1 + [\phi_t^j]^\top [H_t^{j-1}]^{-1}\phi_t^j} \tag{13}$$

An identical formula holds for the symmetric matrix $J_t^j \in \mathbb{R}^{M_v \times M_v}$ by swapping out ϕ_t^j with φ_t^j and H_t^{j-1} with J_t^{j-1} in the above equation.

The Digital Thread $\mathcal{D}_{t+1}^j$ at stage $t+1$ is determined through forward simulation with the latest iteration of the policy $\mu_t^j = A_t^j \phi_t^j + a_t$:

$$\begin{aligned} \mathcal{D}_{t+1}^j &= \Phi_t(\mathcal{D}_t^j, \mu_t^j, z_t^j) \\ z_t^j &\sim p(z_t \,|\, I_t^j, \mu_t^j) \end{aligned} \tag{14}$$

while the Digital Thread $\mathcal{D}_0^j$ at stage $t = 0$ is sampled from $\mathscr{I}$. Sampling the Digital Thread $\mathcal{D}_0 = (Q_0, R_0)$ at stage $t = 0$ involves sampling different distributions for $Q_0 = p(y_0 \,|\, I_0, u_0)$, which can be performed through using a suitable hyperprior distribution or through direct sampling of the parameters of the parametrization of Q_0, if applicable. Sampling R_0 involves direct sampling from the set of allowable values (discrete and/or continuous) its elements can take.

3.4 Numerical Implementation to Solve the Multistage Decision Problem

The numerical implementation for the algorithm presented in Subsection 3.3 is divided between Algorithms 1, 2 and 3. To initialize and train a policy, first INITIALIZEPOLICY is called and then TRAINPOLICY is called however many times is necessary until a convergence threshold on the policy or value function is achieved [38]. Details of the subroutines are given in the following.

Algorithm 1 Simulate and Initialize Policy

1: **procedure** SIMULATEPOLICY($\mathcal{D}_s, \pi_0$)
2: **for** $t = s : 1 : T-1$ **do**
3: $u_t \leftarrow \mu_t(\mathcal{D}_t)$
▷ *Obtain measurement through forward simulation or from physical system*
4: $z_t \sim p(z_t \,|\, I_t, u_t)$
5: $\mathcal{D}_{t+1} \leftarrow \Phi_t(\mathcal{D}_t, u_t, z_t)$
6: **return** $\{\mathcal{D}_t\}_{t=s}^{T}$

1: **procedure** INITIALIZEPOLICY($\{(u_t^d, a_t, b_t, c_t)\}_{t=0}^{T}$)
2: **for** $t = 0 : 1 : T$ **do**
▷ *Initialize values at each stage t*
3: $A_t \leftarrow 0_{N_u \times M_p}$
4: $B_t \leftarrow 0_{1 \times M_p}$
5: $C_t \leftarrow 0_{1 \times M_v}$
6: $H_t^{-1} \leftarrow (1/\eta_t) I_{M_p}$ with $\eta_t > 0$
7: $J_t^{-1} \leftarrow (1/\kappa_t) I_{M_v}$ with $\kappa_t > 0$
▷ *Update parameters of μ_t with initialized values*
8: $\mu_t \leftarrow \{A_t, B_t, C_t, a_t, b_t, c_t, H_t^{-1}, J_t^{-1}, u_t^d\}$
9: $\pi_0 \leftarrow \{\mu_0, ..., \mu_T\}$
10: **return** π_0

Simulate and Initialize Policy Simulating a policy is described in SIMULATEPOLICY. Here, a given policy π_0 along with a Digital Thread $\mathcal{D}_s$ at stage $s \in \mathcal{T}$ are used to generate the future evolution of $\mathcal{D}_s$ from stage s onwards. The output is the trajectory (i.e., states) of the Digital Thread for $t \in \{s, ..., T\}$. Note, the transition model outputs the Digital Thread from stage $t+1$ from stage t so stage T of the Digital Thread is computed from stage $T-1$, thus the for loop is truncated to stage $T-1$.

Information about product lifecycle elements (statistics of inputs, resources, products in operation and their Digital Twins, etc.) at some $t \in \{s, ..., T\}$ within this trajectory is extracted through post-processing of the appropriate $\mathcal{D}_t$. Measurements are synthetically generated if no physical measurements are available during offline training. Online, measurements come directly from test data or the actual physical systems.

To initialize a policy from scratch, INITIALIZEPOLICY is called taking in as input the high-level decision sequence and initial condition parameters. Here, the parameters $\eta_t, \kappa_t > 0$ represent the scaling factors of the initial least squares regularization term (in this implementation, the identity matrix of appropriate size).

Train Policy In TRAINPOLICY, a Digital Thread state $\mathcal{D}_0$ is first sampled from $\mathscr{I}$ followed by generation of a Digital Thread trajectory using input π_0. Using this trajectory as a skeleton, optimization is performed backwards from each Digital Thread state in the trajectory. A call to a deterministic optimizer is made in OPTIMIZER and takes as arguments an initial condition, a cost function, and a constraint function. The deterministic optimizer can be any appropriate off-the-shelf optimizer. For this implementation, we use the trust-region method in MATLAB's FMINUNC and impose inequality constraints using penalty functions.

The optimizer can be run for a fixed number of iterations per call or until a suitable level of convergence is achieved. During or after running of the optimizer, the parameters of μ_t, $\{A_t, B_t, C_t, H_t^{-1}, J_t^{-1}\}$, are updated before moving to stage $t-1$. For implementation, the update index j has been omitted since we only need to keep track of the current values of all relevant objects at any particular point in the routines. The procedure TRAINPOLICY updates all parameters of the policy *once* over all stages $t \in \mathcal{T}$, allowing flexibility for sequential updating in the future when necessary.

At lines 2-14 in COSTFUNCTION of Algorithm 3, M_s samples of the decision near the optimization iteration point are generated and used to update the estimate of S_t. Updating S_t adaptively at the start of the cost function is implemented so that new samples are taken near every evaluation point of the optimization. The terms G_t and O_t are computed using Monte Carlo sampling; the same samples $\{y_t^i\}_{i=1}^{N}$ can be used for both G_t and O_t to save on computation, or for subsequent iterations per optimization call if an expectation-maximization related strategy is used. The policy π_0 in the call to the cost function is passed "by reference" so that updates to any part of π_0 is made immediately available to all levels and Algorithms.

Algorithm 2 Train Policy

procedure TRAINPOLICY(π_0)
  ▷ *Build a skeleton trajectory to perform updates*
  Sample $\mathcal{D}_0 \in \mathscr{I}$
  $\{\mathcal{D}_t\}_{t=0}^{T} \leftarrow$ SIMULATEPOLICY($\mathcal{D}_0, \pi_0$)
  ▷ *Update terms going backwards from $t = T$*
  for $t = T : -1 : 0$ **do**
    $\phi_t \leftarrow \phi_t(\mathcal{D}_t)$
    ▷ *Assign cost and constraint functions for deterministic optimizer at stage t*
    $Q'_t(u) \leftarrow$ COSTFUNCTION($u, \mathcal{D}_t, \pi_0$)
    $G'_t(u) \leftarrow$ CONSTRAINTFUNCTION($u, \mathcal{D}_t$)
    ▷ *Run deterministic optimizer*
    $\{u_t, Q_t\} \leftarrow$ OPTIMIZER($A_t\phi_t + a_t, Q'_t(\cdot), G'_t(\cdot)$)
    ▷ *Update A_t and B_t using RLS update rule*
    $l_t \leftarrow H_t^{-1}\phi_t$
    $H_t^{-1} \leftarrow H_t^{-1} - (l_t l_t^\top)/(1 + \phi_t^\top l_t)$
    $l_t \leftarrow H_t^{-1}\phi_t$
    $A_t \leftarrow A_t + (u_t - A_t\phi_t - a_t)l_t^\top$
    $B_t \leftarrow B_t + (\log Q_t - B_t\phi_t - b_t)l_t^\top$
  return π_0

4 Results

In this section, we provide computational results for the example problem using the numerical algorithm given in Subsection 3.3. Data assimilation and uncertainty reduction trends are given in Subsection 4.1. Select component design, sensor placement, and sensor selection results are given in Subsection 4.2. Comparison of total costs across all policies are given in Subsection 4.3. Finally, a discussion on computational cost and complexity are given in Subsection 4.4.

4.1 Data Assimilation and Uncertainty Reduction

Typical data assimilation and uncertainty reduction as a result of collecting measurements throughout the various lifecycle paths are shown for uncertain loads in Fig. 3, material properties in Fig. 4, and manufacturing process times in Fig. 5.

In Fig. 3, the loads used for the first design are compared to the loads estimated from operational data of the first design after manufacturing and deployment. These estimated loads are then used for the final design. The mean and two standard deviations of the variance for the initial estimate of loads (before any data assimilation) are shown with the red dashed-dotted line and red shading, respectively. Similarly, the mean and two standard deviations of the variance for the estimate of the loads after a design is deployed are shown with the blue dashed line and blue shading, respectively. The actual loads to be learned are shown with the thick magenta line. In this figure, we see that the large shifts in the mean for all loading components and variance reduction of the moments and shear after data assimilation indicate that the design of the next generation can be built lighter (and therefore at a lower cost) than the previous generation. This is because the loads for this particular scenario are learned to be of lower magnitude than what was used for the design of the previous generation. However, in order to have obtained this knowledge first, we had to manufacture and deploy first, foregoing any benefits provided by performing coupon experiments sooner.

In Fig. 4, the estimates of material strength properties known initially are compared to the estimates after learning from coupon level experiments. The probability density function for the initial estimate of strength properties is given by the red shaded curve. Similarly, the probability density function for the strength properties after performing coupon level experiments is given by the blue shaded curve. The actual strength properties to be learned are shown with the thick magenta vertical line. Here we see that the large shifts in the mean and variance reduction of the strength properties after performing the coupon experiments indicate that the next design to be deployed will benefit from higher and more confident material strength property estimates and therefore be lighter and of lower cost. Of course, to have obtained this knowledge first, we had to forgo deploying a design earlier and the potential benefits it could have provided.

In Fig. 5, the estimates of manufacturing timestamps known initially are compared to the estimates after learning from data collected from the manufacturing of a component. The mean and two standard deviations of the variance for the initial estimate and final estimate of the timestamps are shown with the blue-shaded bars and yellow-shaded bars, respectively. The actual timestamps to be learned are shown with the red-shaded bars. From this figure, we see that in addition to achieving better estimates of process times, we also see that only a few number of process steps contribute significantly to the total

Algorithm 3 Cost and Constraint Functions

procedure COSTFUNCTION($u_t, \mathcal{D}_t, \pi_0$)
  if $t < T$ **and** $\gamma > 0$ **then**
    for $i = 1:1:M_s$ **do**
      $\hat{u}_t \leftarrow$ Sample in a neighborhood of u_t
      ▷ *Obtain measurement through forward simulation*
      $z_t \sim p(z_t \,|\, I_t, \hat{u}_t)$
      ▷ *Sample forward value function*
      $V_{t+1} \leftarrow \exp\{B_{t+1}\phi_{t+1}(\Phi_t(\mathcal{D}_t, \hat{u}_t, z_t)) + b_{t+1}\}$
      ▷ *Update C_t using RLS update rule*
      $\varphi_t \leftarrow \varphi_t(\mathcal{D}_t, \hat{u}_t)$
      $q_t \leftarrow J_t^{-1}\varphi_t$
      $J_t^{-1} \leftarrow J_t^{-1} - (q_t q_t^\top)/(1 + \varphi_t^\top q_t)$
      $q_t \leftarrow J_t^{-1}\varphi_t$
      $C_t \leftarrow C_t + (\log V_{t+1} - C_t\varphi_t - c_t)q_t^\top$
    ▷ *Construct forward value function*
    $S_t \leftarrow \exp\{C_t\varphi_t(\mathcal{D}_t, u_t) + c_t\}$
  else
    ▷ *V_{T+1} (and hence S_T) is zero*
    $S_t \leftarrow 0$
  ▷ *Evaluate O_t using Monte Carlo*
  $O_t \leftarrow \frac{1}{N}\sum_{i=1}^{N} r_t(\mathcal{D}_t, u_t, y_t^i)$ where $y_t^i \sim p(y_t \,|\, I_t, u_t)$
  ▷ *Construct Bellman equation*
  $Q_t \leftarrow O_t + \gamma S_t$
  return Q_t

procedure CONSTRAINTFUNCTION($u_t, \mathcal{D}_t$)
  ▷ *Evaluate G_t using Monte Carlo*
  $G_t \leftarrow \frac{1}{N}\sum_{i=1}^{N} g_t(\mathcal{D}_t, u_t, y_t^i)$ where $y_t^i \sim p(y_t \,|\, I_t, u_t)$
  return G_t

manufacturing time.

4.2 Component Design and Sensor Placement/Selection

We highlight the first and final designs produced through the greedy, sensor placement, and sensor selection policies for the *EDD* decision sequence in Fig. 6 for component thickness and sensor location. For this example problem, optimized design geometries tend to be thicker around the holes and near the left and right of the component, and regions directly below the holes tend to thin out. Optimized geometries also tend to favor modifying thickness over modifying fiber angle to minimize costs. As a result, fiber angle tends to be similar across all policies for the final design. Typical fiber directions for the final design are shown in Fig. 7. Here, fiber steering tends to be more prominent near the surface of the component as a result of the structure being heavily driven by out-of-plane loading for this problem.

4.3 Comparison of Total Costs

Comparison of mean total costs for all policies is shown in Fig. 8. Even though the final design produced from the *EDD* and *DED* policies benefit from both operational and coupon level experimental data, the costs for each policy are accumulated differently. As a result, we see that the *EDD* policies achieve lower total costs than the *DED* policies. The best strategy from the given initial state of the Digital Thread is to first perform experiments to drive down the uncertainty of the material strength properties, and second to manufacture and deploy that design to learn about the uncertain loading conditions from data collected through operation. Interestingly, we see that manufacturing and deploying first leads to higher overall costs as a result of designing heavier and more conservative designs from the lack of data about the material strength properties earlier. Additionally, the corresponding operational costs are higher and accrued over a longer time frame. The results overall show that material strength properties have a larger impact on the overall costs than do the input loads, despite the fact that the means of the material strength properties were only 10% away from the true values compared to 50% for the input loads.

From Fig. 8 we see that there is no strong benefit of sensor placement in this example problem setting, even when the

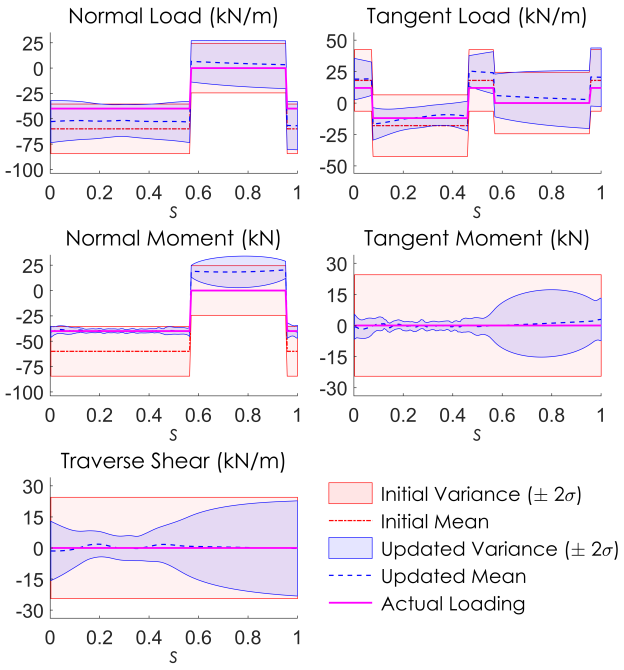

Fig. 3. Typical data assimilation and uncertainty reduction for the uncertain input loads after collecting strain sensor measurements during operation. Here, σ stands for standard deviation. Loading components are given as a function of a parameter $s \in [0,1]$ that wraps around the outer boundary of the component starting at the center of the far right edge of the component.

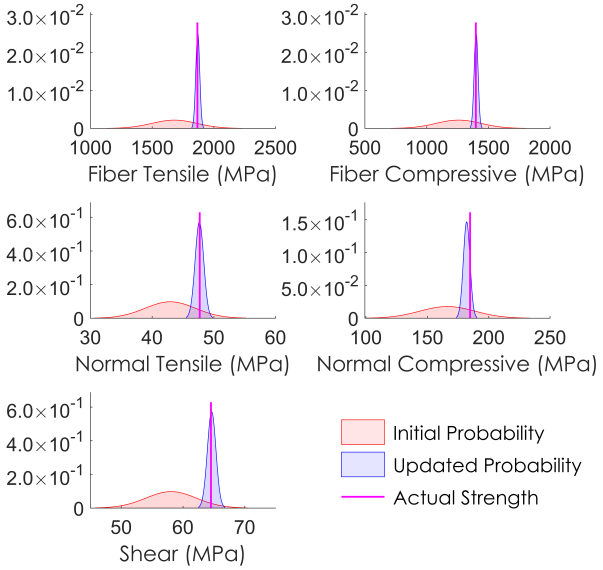

Fig. 4. Typical data assimilation and uncertainty reduction for the uncertain material properties (material strengths) after collecting data from coupon failure test.

location of sensors are not penalized. As long as sensors are initially well dispersed on the top surface of the component, the cost improvement from moving the sensors around is small ($< 1\%$). However, in our studies we observed that sensor placement is typically more aggressive in the *DED* case than the *EDD* case. This is because in the *DED* case, not learning the material properties before manufacturing the first design means that the first design will be thicker (and thus of higher costs) compared to the *EDD* case. Consequently, more effort is put into placement of the sensors to recover the total cost.

Though the changes in total accrued cost are low (largely due to our particular choice for the sensor selection costs in relation to total design costs), Fig. 8 shows that the optimized sensor selection policies have effective sensor utilization of less than 40%. That is, only 40% of all sensors need to be effectively active to recover sufficient data to minimize costs.

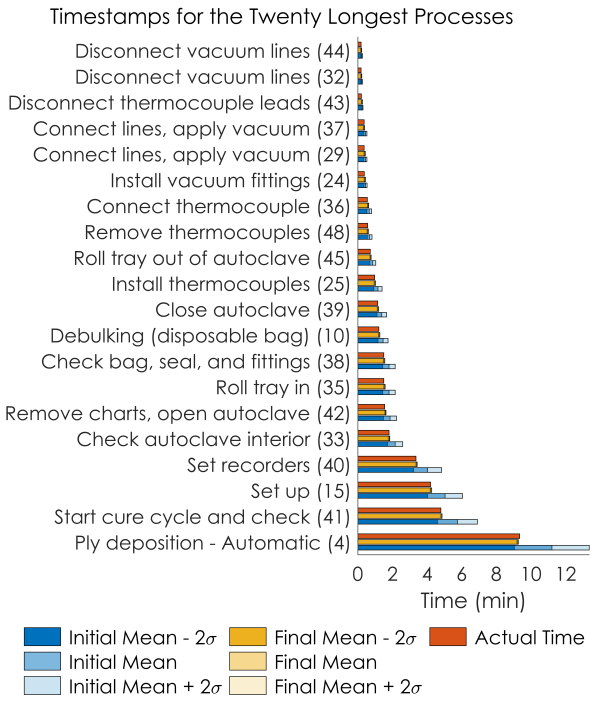

Fig. 5. Typical data assimilation and uncertainty reduction for manufacturing times after collecting timestamps during manufacturing, ordered by decreasing step times (bottom to top) of the twenty longest processes. Here, σ stands for standard deviation. Numbers in parenthesis correspond to the step number in the manufacturing process.

To understand why this is the case, we compare different load components in Fig. 3. For this example setup, the moments are the best resolved (low variance and better mean estimates) loading components while the normal and tangent loads are not resolved as effectively. From the standpoint of structural analysis, this means that generated designs are robustly designed to variations of normal and tangent loading while dominant structural sizing will depend on the resolved moments and possibly shears. Based on the low effective sensor utilization values, these moments can be resolved effectively with out large utilization of the available sensors. Interestingly, because sensor utilization is penalized, the optimized sensor selection policy for the *DED* case found it more advantageous to drive the sensor utilization to less than 20%, forfeiting some of the structural efficiency of the next design, in order to preserve lower overall costs.

4.4 Computational Cost and Complexity

Computational cost depends on the number of the design decision variables, number of uncertain input variables, number of measurements, and mesh discretization for the finite element model and cost calculation. For this example problem, each of the two design-based stages consisted of 3,105 design variables (parameters for fiber steering, thickness, and sensor location). The high number of design variables arises as a result of using a direct parametrization of the finite element model. The number of uncertain input variables is 1,466 per stage (uncertain loads, material properties, and manufacturing parameters). The number of measurements is 187 per stage (strain measurements, material properties, and manufacturing time stamps). The policy and function approximation uses 2000 basis functions with input dimensions on the order of the number of uncertain inputs for μ_t and V_t, and with input dimensions on the order of the number of uncertain inputs and design decision variables for S_t. These approximations were updated one data point at a time using the rank-1 update of the RLS formulations (i.e., no matrix inversions). The process models for this problem for the uncertain input variables per stage are taken to be linear with Gaussian noise, thus the Bayesian filter in Eqn. (4) is calculated *analytically* using a variant of the Kalman filter (with the prediction and analyze steps reversed). This requires computing matrix inverses of dimension equal to the number of measurements per stage. As a result, samples for Monte Carlo estimation could be easily drawn using appropriately scaled normal distributions at each stage. It was determined that 50-100 Monte Carlo samples can be used to achieve estimates of R_t and G_t with less than 1% error (the point of reference being taken at 10,000 samples). This was a result of these terms having small variances for this problem. Optimization was accelerated through the use of gradient information computed using adjoint solves of the finite element model and analytical derivatives of the policy and function approximation forms. Detailed description of all modeling of terms can be found in [5]. In total, 100 policy updates for all six policies took on the order of 8 hrs on a Windows 10 64-bit machine with 16 GB of RAM where the vast majority of the time was spent on solving

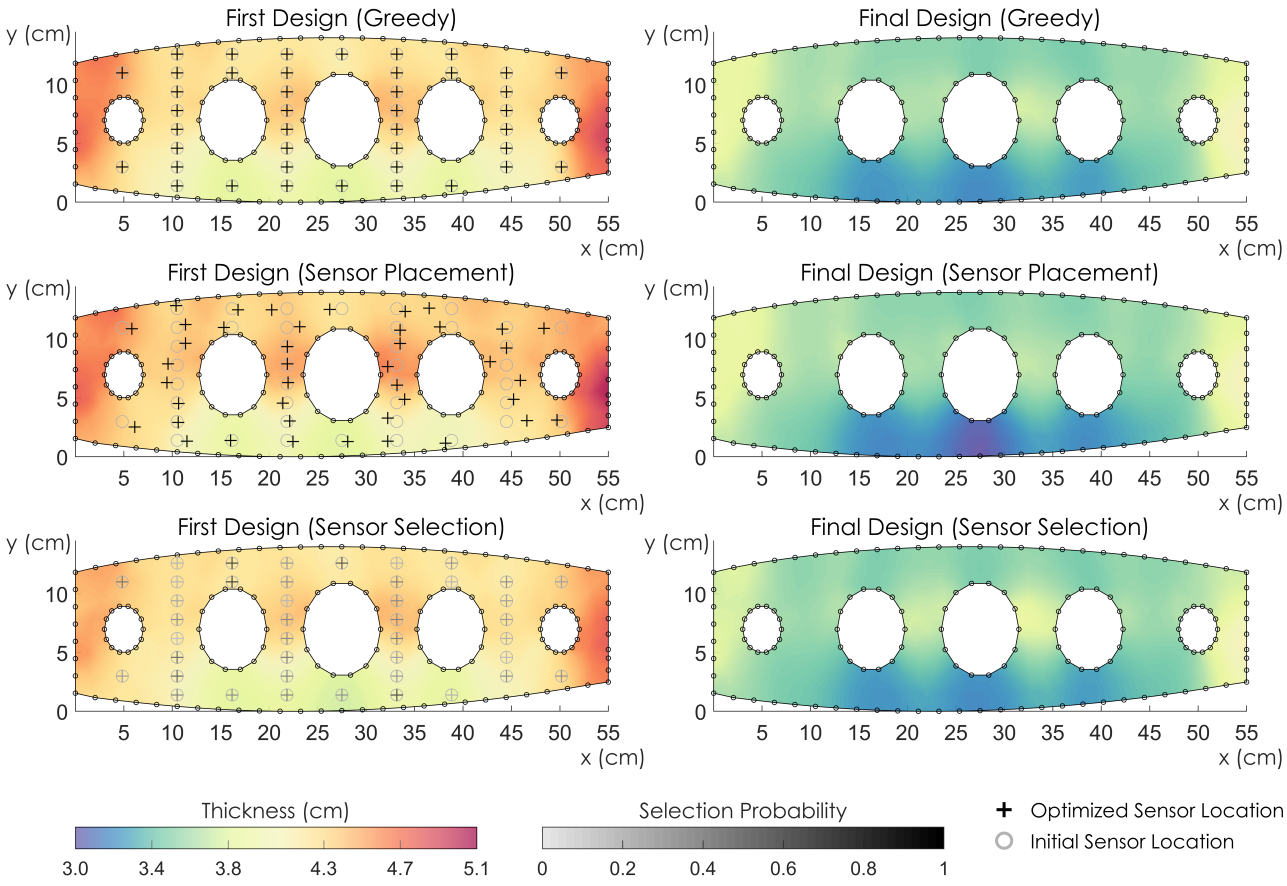

Fig. 6. Thicknesses and sensor placement/selection for the greedy, sensor placement, and sensor selection policies for the EDD decision sequence. Sensor locations are given by "+" markers while their initial locations (before optimization) are shown in the grayed out circle markers. Sensor activation probabilities are represented using a gray shading of the "+" markers, lighter for values near zero and darker for values near one. Strain sensor data is only collected for the first design.

the finite element model. Results were run for 2000 policy updates, although convergence of the policy to within 2% of final values was achievable within 200 updates.

As with all methods involving quantification of uncertainty, efficiency of the approaches proposed here may become challenging for higher dimensional problems. For instance, the main hurdle for scalability of the stage-wise optimization (line 8 in TRAINPOLICY of Algorithm 2) is the number of design variables. A way to reduce the number of design variables is through a reduced geometrical representation, i.e., not using a direct parametrization of a detailed finite element model but using instead simplified geometry or another low-dimensional representation of the component. Reducing the number of design variables for a given component then allows scaling up to multiple components more readily. Supplying derivative information for the stage-wise optimization is also beneficial. In addition, rather than run the detailed finite element model during filter updates (line 5 in SIMULATEPOLICY of Algorithm 1), measurement generation via forward simulation (line 4 in SIMULATEPOLICY of Algorithm 1 and line 5 in COSTFUNCTION of Algorithm 3), and stage-wise optimization, a projection-based reduced order model can be employed instead [39,40]. For filter updates themselves, exploiting independence of the uncertain input variables can alleviate high dimensionality allowing one to work with smaller transition models to propagate uncertain quantities. For instance, the loads on the component during operation are physically independent of the cost of manufacturing that component, when given the design. Therefore filtering for loads and manufacturing parameters can be done independently. For non-linear models, sequential importance sampling involved for the filter updates as well as sampling for R_t and G_t can be accelerated through the use of multifidelity Monte Carlo sampling techniques where inexpensive (but less accurate) models are used in conjunction with expensive (but more accurate) models to reduce the total number of expensive evaluations for sampling [41,42]. In addition, further acceleration for sampling can be achieved through the use of parallel computations.

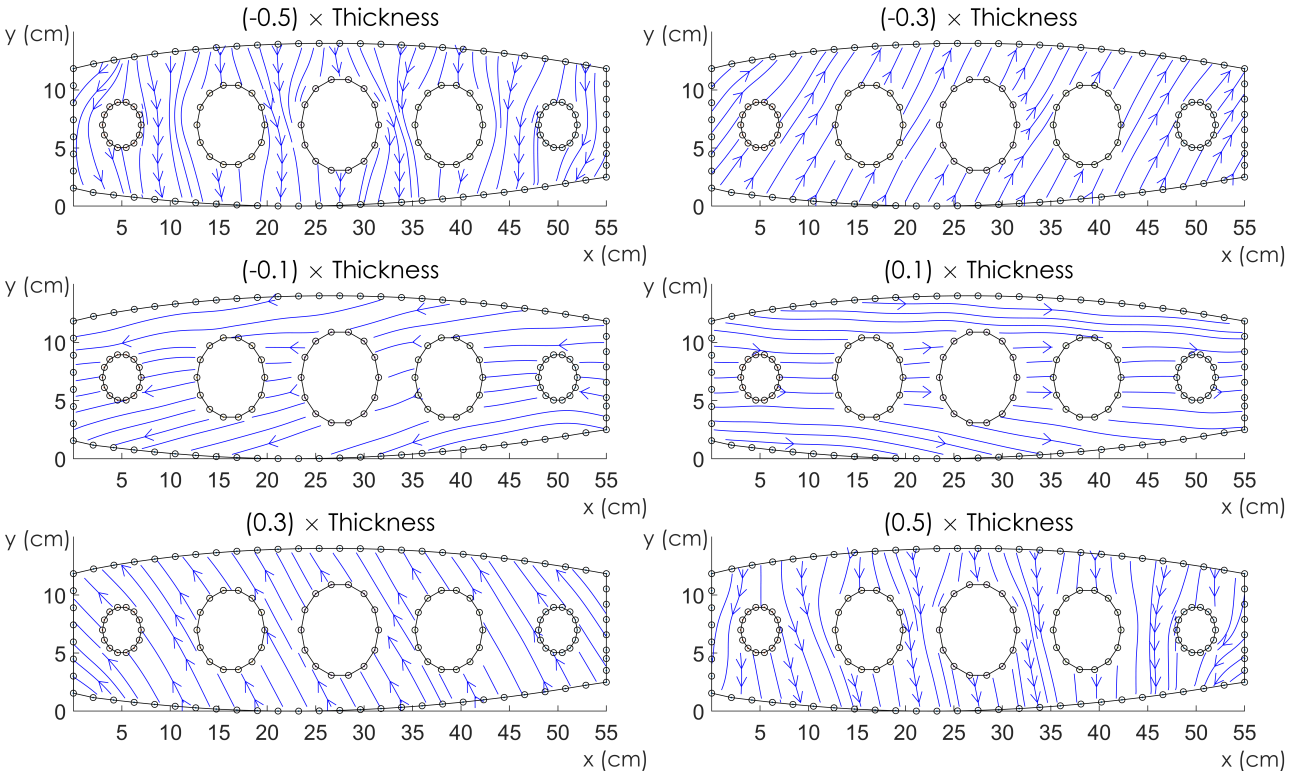

Fig. 7. Typical optimized fiber direction for the final design across all policies. Fiber direction is shown in a scaled vertical coordinate where a factor of 0.5 of the thickness corresponds to the top surface, a factor of 0 to the mid-plane, and a factor of -0.5 to the bottom surface. Arrows designate the local fiber zeroth direction.

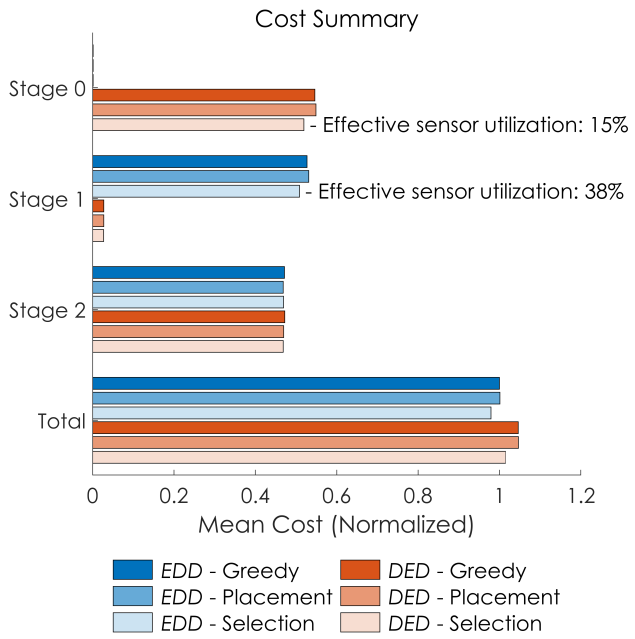

Fig. 8. Comparison of mean costs for all policies. Mean costs are normalized with respect to the total mean cost of the EDD - Greedy policy. Effective sensor utilization is reported for policies with sensor selection.

5 Conclusions

In this paper, we presented a methodology to enable decision making under uncertainty for a Digital Thread enabled design process and applied it to a relevant structural composites design problem. This methodology enabled assessing a variety of decision making strategies involving sensor placement, sensor selection, and structural tailoring as well as high-level decisions involving experimentation or manufacturing and deploying. Implementation of an approximate dynamic programming algorithm that utilizes a combination of function and policy approximation coupled with recursive least squares was also detailed.

In addition to learning of sensor limitations in resolving various uncertain loading inputs in the example, our method recognized that designs can be made robust to normal and tangent loadings where major sizing changes were driven predominately by moment or transverse shear loading. Simultaneously, our method found that it could significantly reduce the effective number of sensors that are active to be able to sense the dominant loading components efficiently while forfeiting learning precisely the other loading components. This translated to reduced costs since fewer effective sensors are needed to make cost efficient design decisions. In addition, our method was also able to show that sensor placement has only a small impact on the overall costs for this example problem setting.

Overall, our design methodology showcases how data-driven design decisions change based on the sources of uncertainty and the sequence in which we attempt to reduce them. Furthermore, limitations and advantages of resources can both be exploited to drive costs down. Our methodology is able to identity the order in which uncertainty must be reduced to achieve lowest costs. Resulting policies output realizable design geometries that can be assessed for further detailed analysis. The novelty in our method is that sensor placement and selection can be determined directly (and to the degree necessary) from total accrued cost without requiring specification of additional metrics.

Note that our solution method yields a policy, i.e., a function of the Digital Thread. For the example problem, we tested this policy on just *one* set of inputs unknown to the policy. However, this same policy can be evaluated for other input scenarios, provided the inputs and initial conditions of these other scenarios are within some reasonable neighborhood of where the policy was trained. Furthermore, the computational effort to train a policy is divided between an offline step (initialization and training of the policy) and inexpensive online evaluations for prediction or subsequent updates.

In our cost modeling for the example design problem, experimental coupon failure test costs are small with respect to manufacturing costs. This may not be the case for larger scale static/fatigue testing of assemblies or systems. Nevertheless, our method is adaptable through appropriate modification of the stage-wise costs and constraints. In addition, input loading variances are relatively high with respect to the mean for this example, so designs generated by the optimized policies reflect robustness to a wide range of possible uncertain inputs. Reducing input variance can lead to more specialized designs (more specific tailoring of fiber angles and thickness) through cost savings obtained by limiting uncertain inputs that are less likely to occur.

Future work will look into multiple loading/operating conditions and failure modes, other recurrent design applications, expanding lifecycle costs to include inspections, maintenance, and repair, as well as applying the methodology to assemblies or larger systems consisting of multiple components and/or assemblies. In the latter, further development may likely need multilevel and sparse representations, reduced parametrization of individual component details, as well as employing reduced-order modeling of physics-based simulations, in order to manage complexity and retain computational performance.

Acknowledgements

The work was supported in part by AFOSR grant FA9550-16-1-0108 under the Dynamic Data Driven Application System Program (Program Manager Dr. E. Blasch); The MIT-SUTD International Design Center; and the United States Department of Energy Office of Advanced Scientific Computing Research (ASCR) grants DE-FG02-08ER2585 and DE-SC0009297, as part of the DiaMonD Multifaceted Mathematics Integrated Capability Center (program manager Dr. S. Lee).

Obtaining Analysis Code

Code to generate all analysis data and figures is available online at `https://github.com/victornsi/DT-AVT`. All code is written in MATLAB R2018b on a Windows 10 64-bit machine.

Nomenclature

Superscripts/Subscripts

t Time or stage designating sequence of decisions

j Iteration index of terms in numerical algorithm

$*$ Designation for optimal quantities or functions

Statistical Operators

$\mathbb{E}$ Expected value

$\mathbb{P}$ Probability measure
p Probability distribution
Time or Stage
$\mathcal{T}$ Set of time or stage indices in consideration
T Final time index or horizon length from $t = 0$
Uncertain Inputs
y_t Uncertain inputs at stage t
N_y Total number of uncertain inputs
ν Measure or volume over uncertain inputs
Measurements
z_t Measurements across product lifecycle at stage t
N_z Total number of measurements
N_m Total number of steps in manufacturing process
N_s Total number of strain sensors on component
Decisions
u_t Decisions at stage t
u_t^d High-level decision between performing coupon testing or manufacturing and deploying a design at stage t
u_t^p Fiber steering, component thickness, and sensor placement parameters for design at stage t
u_t^s Sensor selection probabilities for design at stage t
e_t^s Effective sensor utilization during sensor selection for design at stage t
N_u Total number of decision variables
N_p Total number of parameters to define fiber direction, component thickness, and sensor locations
Digital Thread
$\mathcal{D}_t$ Digital Thread at stage t
Q_t Representation of uncertainty in the product lifecycle at stage t
R_t Resources related to tools, methods, and processes in the product lifecycle at stage t
I_t History of collected data at stage t
$\mathcal{I}$ Information space over product lifecycle
Φ_t Digital Thread transition model at stage t
Ψ_t Resource transition model at stage t
Multistage Decision Statement
V_t^* Optimal value function at stage t
π_t^* Optimal policy at stage t
μ_t^* Optimal policy stage function at stage t
r_t Stage-wise cost function at stage t
g_t Stage-wise constraint function at stage t
γ Discount factor
N_g Total number of stage-wise constraints for design
Numerical Algorithm
O_t Expected value of the stage-wise cost function at stage t
G_t Expected value of the stage-wise constraint function at stage t
S_t Expected value of the forward value function at stage $t+1$
A_t Matrix of basis coefficients for the policy at stage t
B_t Matrix of basis coefficients for the value function at stage t
C_t Matrix of basis coefficients for S_t at stage t
a_t Vector used for setting initial conditions for the policy function at stage t
b_t Scalar used for setting initial conditions for the value function at stage t
c_t Scalar used for setting initial conditions for S_t at stage t
ϕ_t Vector of basis functions for the value function and policy at stage t
φ_t Vector of basis functions for S_t at stage t
H_t Incremental regularization matrix used in the recursive least squares update for the policy and value function at stage t
J_t Incremental regularization matrix used in the recursive least squares update for S_t at stage t
η_t Least squares regularization scaling term for the policy and value function at stage t
κ_t Least squares regularization scaling term for S_t at stage t
M_p Total number of basis functions used for the parametrization of the policy and value function
M_v Total number of basis functions used for the parametrization of S_t at stage t

References

[1] US Airforce, 2013. *Global Horizons Final Report: United States Air Force Global Science and Technology Vision - AF/ST TR 13-01*. United States Air Force.

[2] Kraft, E., 2015. "Hpcmp create-av and the air force digital thread". *in AIAA SciTech 2015, 53rd AIAA Aerospace Sciences Meeting, Kissimmee, Florida*, pp. 1–13. doi:10.2514/6.2015-0042.

[3] West, T., and Pyster, A., 2015. "Untangling the digital thread: The challenge and promise of model-based engineering in defense acquisition". *INSIGHT,* **18**(2), pp. 45–55. doi:10.1002/inst.12022.

[4] Singh, V., and Willcox, K., 2018. "Engineering design with digital thread". *AIAA Journal,* **56**(11), pp. 4515–4528. doi:10.2514/1.J057255.

[5] Singh, V., 2019. *Towards a Feedback Design Process Using Digital Thread*. PhD Thesis, MIT.

[6] Stark, J., 2015. *Product Lifecycle Management*, 3rd ed., Vol. 1. Springer International. doi:10.1007/978-3-319-17440-2.

[7] Wymore, A., 1993. *Model-Based Systems Engineering*. CRC press.

[8] Ramos, A., Ferreira, J., and Barceló, J., 2012. "Model-based systems engineering: An emerging approach for modern systems". *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* **42**(1), pp. 101–111. doi:10.1109/TSMCC.2011.2106495.

[9] Estefan, J., 2009. "Mbse methodology survey". *INSIGHT,* **12**(4), pp. 16–18. doi:10.1002/inst.200912416.

[10] Cloutier, R., 2009. "Introduction to this special edition on model-based systems engineering". *INSIGHT,* **12**(4), pp. 7–8. doi:10.1002/inst.20091247.

[11] Loper, M., ed., 2015. *Modeling and Simulation in the Systems Engineering Life Cycle: Core Concepts and Accompanying Lectures*. Springer-Verlag London. doi:10.1007/978-1-4471-5634-5.

[12] Mies, D., Marsden, W., and Warde, S., 2016. "Overview of additive manufacturing informatics: "a digital thread"". *Integrating Materials and Manufacturing Innovation,* **5**, pp. 114–142. doi:10.1186/s40192-016-0050-7.

[13] Mahan, T., Meisel, N., McComb, C., and Menold, J., 2019. "Pulling at the digital thread: Exploring the tolerance stack up between automatic procedures and expert strategies in scan to print processes". *ASME Journal of Mechanical Design,* **141**(2), pp. 021701–1 – 021701–12. doi:10.1115/1.4041927.

[14] Lee, Y., and Fong, Z., 2020. "Study on building digital-twin of face-milled hypoid gear from measured tooth surface topographical data". *ASME Journal of Mechanical Design,* **142**(11), pp. 113401–1 – 113401–13. doi:10.1115/1.4046915.

[15] Gharbi, A., Sarojini, D., Kallou, E., Harper, D., Petitgenet, V., Rancourt, D., Briceno, S., and Mavris, D., 2017. "Standd: A single digital thread approach to detailed design". *in AIAA SciTech 2017, 55th AIAA Aerospace Sciences Meeting, Grapevine, Texas*, pp. 1–13. doi:10.2514/6.2017-0693.

[16] Thomsen, B., Kokkolaras, M., Månsson, T., and Isaksson, O., 2017. "Quantitative assessment of the impact of alternative manufacturing methods on aeroengine component lifing decisions". *ASME Journal of Mechanical Design,* **139**(2), pp. 021401–1–021401–10. doi:10.1115/1.4034883.

[17] Smith, R., 2013. *Uncertainty Quantification: Theory, Implementation, and Applications*. SIAM.

[18] Zaman, K., McDonald, M., and Mahadevan, S., 2011. "Probabilistic framework for uncertainty propagation with both probabilistic and interval variables". *ASME Journal of Mechanical Design,* **133**(2), pp. 021010–1 – 021010–14. doi:10.1115/1.4002720.

[19] Xi, Z., 2019. "Model-based reliability analysis with both model uncertainty and parameter uncertainty". *ASME Journal of Mechanical Design,* **141**(5), pp. 051404–1 – 051404–11. doi:10.1115/1.4041946.

[20] Kall, P., Wallace, S., and Kall, P., 1994. *Stochastic Programming*. John Wiley & Sons, Chichester.

[21] Bertsimas, D., Brown, D., and Caramanis, C., 2011. "Theory and applications of robust optimization". *SIAM Review,* **53**(3), pp. 464–501. doi:10.1137/080734510.

[22] Yao, W., Chen, X., Luo, W., van Tooren, M., and Guo, J., 2011. "Review of uncertainty-based multidisciplinary design optimization methods for aerospace vehicles". *Progress in Aerospace Sciences,* **47**(6), pp. 450–479. doi:10.1016/j.paerosci.2011.05.001.

[23] Du, X., and Chen, W., 2002. "Efficient uncertainty analysis methods for multidisciplinary robust design". *AIAA Journal,* **40**(3), pp. 545–552. doi:10.2514/2.1681.

[24] Kokkolaras, M., Mourelatos, Z., and Papalambros, P., 2004. "Design optimization of hierarchically decomposed multilevel systems under uncertainty". *ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Volume 1: 30th Design Automation Conference*, pp. 613–624. doi:10.1115/DETC2004-57357.

[25] Atkinson, A., Donev, A., and Tobias, R., 2007. *Optimum Experimental Designs, with SAS*, Vol. 34 of *Oxford Statistical Science Series*. Oxford University Press.

[26] Huan, X., and Marzouk, Y., 2013. "Simulation-based optimal bayesian experimental design for nonlinear systems". *Journal of Computational Physics,* **232**(1), pp. 288–317. doi:10.1016/j.jcp.2012.08.013.

[27] Mindlin, R., 1951. "Influence of rotatory inertia and shear on flexural motions of isotropic, elastic plates". *ASME Journal of Applied Mechanics,* **18**, pp. 31–38.

[28] Reissner, E., 1945. "The effect of transverse shear deformation on the bending of elastic plates". *ASME Journal of*

Applied Mechanics, **12**, pp. A69–A77.
[29] Gutowski, T., Hoult, D., Dillon, G., Neoh, E., Muter, S., Kim, E., and Tse, M., 1994. "Development of a theoretical cost model for advanced composite fabrication". *Composites Manufacturing,* **5**(4), pp. 231–239. doi:10.1016/0956-7143(94)90138-4.
[30] Northrop Corporation, 1976. "Advanced composites cost estimating manual (accem)". *AFFDL-TR-76-87,* **1**, pp. 1–88.
[31] Joshi, S., and Boyd, S., 2009. "Sensor selection via convex optimization". *IEEE Transactions on Signal Processing,* **57**(2), pp. 451–462. doi:10.1109/TSP.2008.2007095.
[32] Doucet, A., Freitas, N., and Gordon, N., eds., 2001. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag New York. doi:10.1007/978-1-4757-3437-9.
[33] Tsai, S., and Wu, E., 1971. "A general theory of strength for anisotropic materials". *Journal of Composite Materials,* **5**(1), pp. 58–80. doi:10.1177/002199837100500106.
[34] Chaudhuri, A., Norton, M., and Kramer, B., 2020. "Risk-based design optimization via probability of failure, conditional value-at-risk, and buffered probability of failure". *in AIAA SciTech 2020, Managing Multiple Information Sources of Multi-Physics Systems, Orlando, FL*, pp. 1–18. doi:10.2514/6.2020-2130.
[35] Busoniu, L., Babuska, R., Schutter, B., and Ernst, D., 2010. *Reinforcement Learning and Dynamic Programming Using Function Approximators*, Vol. 39 of *in Automation and Control Engineering*. CRC Press.
[36] Björck, A., 1996. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics.
[37] Powell, W., 2011. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed. Wiley Series in Probability and Statistics. Wiley.
[38] Bertsekas, D., 2012. *Dynamic Programming and Optimal Control - Approximate Dynamic Programming*, 4th ed., Vol. II. Athena Scientific.
[39] Benner, P., Gugercin, S., and Willcox, K., 2015. "A survey of projection-based model reduction methods for parameteric dynamical systems". *SIAM Review,* **57**(4), pp. 483 – 531. doi:10.1137/130932715.
[40] Quarteroni, A., and Rozza, G., eds., 2014. *Reduced Order Methods for Modeling and Computational Reduction*, Vol. 9 of *MS&A*. Springer.
[41] Peherstorfer, B., Willcox, K., and Gunzburger, M., 2016. "Optimal model management for multifidelity monte carlo estimation". *SIAM Journal of Scientific Computing,* **38**(5), pp. A3163–A3194. doi:10.1137/15M1046472.
[42] Kramer, B., Marques, A., Peherstorfer, B., Villa, U., and Willcox, K., 2019. "Multifidelity probability estimation via fusion of estimators". *Journal of Computational Physics,* **392**, pp. 385–402. doi:10.1016/j.jcp.2019.04.071.

